Version 31.0: Summer '14

SOAP API Developer's Guide

Last updated: June 27, 2014

© Copyright 2000–2014 salesforce.com, inc. All rights reserved. Salesforce.com is a registered trademark of salesforce.com, inc., as are other names and marks. Other marks appearing herein may be trademarks of their respective owners.

Table of Contents

GETTING STARTED....................1

Chapter 1: Introducing SOAP API....................1

When to Use the SOAP API....................2
Customize, Integrate, and Extend Your Salesforce Solutions....................3
Supported Salesforce Editions....................3
Standards Compliance....................3
Development Platforms....................4
SOAP API Support Policy....................4
Choosing a WSDL....................5
Related Resources 5
What's New in Version 31.0....................5
Quick Start....................6
Step 1: Obtain a Salesforce Developer Edition Organization....................6
Step 2: Generate or Obtain the Web Service WSDL....................6
Step 3: Import the WSDL File Into Your Development Platform....................7
Step 4: Walk Through the Sample Code....................8

Chapter 2: Standard and Custom Object Basics....................26

Primitive Data Types....................27
Field Types....................29
Compound Fields—Beta....................33
Address Compound Fields....................34
Geolocation Compound Fields....................36
Compound Field Considerations and Limitations....................36
API Data Types and Salesforce Field Types....................37
Core Data Types Used in API Calls....................37
sObject....................38
API Fault Element....................38
ExceptionCode....................39
Error....................44
StatusCode....................44
System Fields....................55
Required Fields....................56
Frequently-Occurring Fields....................56
API Field Properties....................57
Relationships Among Objects....................58
Relabeling Fields and Tabs and the API....................59
Force.com AppExchange Object Prefixes and the API....................59
Custom Objects....................59

Chapter 3: Call Basics....................62

Characteristics of API Calls....................63

Factors that Affect Data Access....63
Package Version Settings....65

Chapter 4: Error Handling....67
Error Handling for Session Expiration....68
More About Error Handling....68

Chapter 5: Security....69
User Authentication....70
User Profile and Permission Sets Configuration....70
Security Token....70
Sharing....71
Implicit Restrictions for Objects and Fields....72
API Access in Force.com AppExchange Packages....72
Outbound Port Restrictions....74

Chapter 6: Using the Partner WSDL....75
Obtaining the Partner WSDL File....76
Calls and the Partner WSDL....76
Objects, Fields, and Field Data and the Partner WSDL....77
Queries and the Partner WSDL....77
Namespaces in the Partner WSDL....78
Package Versions and the Partner WSDL....78
User Interface Themes....79
Examples Using the Partner WSDL....79
Sample query and queryMore Calls....82
Sample search Call....84
Sample create Call....86
Sample update Call....88

REFERENCE....91

Chapter 7: Data Model....91
Sales Objects....92
Task and Event Objects....93
Support Objects....94
Document, Note, and Attachment Objects....95
User, Sharing, and Permission Objects....96
Profile and Permission Objects....97
Record Type Objects....98
Product and Schedule Objects....99
Sharing and Team Selling Objects....100
Customizable Forecasting Objects....101
Forecasts Objects....101
Territory Management....103
Process Objects....104

Content Objects....105
Chatter Objects....105
Chatter Feed Objects....107
Salesforce Knowledge Objects....108

Chapter 8: Standard Objects....109
AcceptedEventRelation....128
Account....130
AccountCleanInfo 151
AccountContactRole....173
AccountFeed....174
AccountHistory....181
AccountOwnerSharingRule....183
AccountPartner....186
AccountShare....188
AccountTag....191
AccountTeamMember....192
AccountTerritoryAssignmentRule....194
AccountTerritoryAssignmentRuleItem....196
AccountTerritorySharingRule....197
ActivityHistory....200
AdditionalNumber....208
AllowedEmailDomain....210
Announcement....210
ApexClass....211
ApexComponent....214
ApexLog....217
ApexPage....219
ApexTestQueueItem....222
ApexTestResult....224
ApexTrigger....226
AppMenuItem....231
Approval....235
Article Type__DataCategorySelection....237
Article Type__Feed....239
Asset....245
AssetFeed....248
AssetTag....254
AssignmentRule....256
AsyncApexJob....257
AttachedContentDocument....259
Attachment....261
AuthSession....265
Bookmark....267
BrandTemplate....268

BusinessHours....270
BusinessProcess....274
CallCenter....276
Campaign....278
CampaignFeed....286
CampaignMember....293
CampaignMemberStatus....296
CampaignOwnerSharingRule....298
CampaignShare....300
CampaignTag....301
Case....303
CaseArticle....312
CaseComment....313
CaseContactRole 316
CaseFeed....317
CaseHistory....324
CaseMilestone....326
CaseOwnerSharingRule....329
CaseShare....331
CaseSolution....333
CaseStatus....334
CaseTag....336
CaseTeamMember....337
CaseTeamRole....338
CaseTeamTemplate....339
CaseTeamTemplateMember....339
CaseTeamTemplateRecord....340
CategoryData....341
CategoryNode....342
CategoryNodeLocalization....344
ChatterActivity....347
ChatterAnswersActivity....349
ChatterAnswersReputationLevel....354
ChatterConversation....355
ChatterConversationMember....355
ChatterMessage....356
ClientBrowser....357
CollaborationGroup....359
CollaborationGroupFeed....364
CollaborationGroupMember....370
CollaborationGroupMemberRequest....372
CollaborationInvitation....374
CombinedAttachment....377
Community (Zone)....379

Contact..380
ContactCleanInfo 392
ContactFeed..402
ContactHistory..409
ContactOwnerSharingRule...410
ContactShare...412
ContactTag..414
ContentDocument..415
ContentDocumentFeed..421
ContentDocumentHistory...427
ContentDocumentLink...429
ContentVersion..431
ContentVersionHistory...440
ContentWorkspace..442
ContentWorkspaceDoc..444
Contract..446
ContractContactRole 455
ContractFeed...456
ContractHistory..463
ContractLineItem...465
ContractLineItemHistory..468
ContractStatus...470
ContractTag...471
CronJobDetail..473
CronTrigger...473
CurrencyType..476
CustomBrand 478
CustomBrandAsset 479
Custom Object__Feed..481
CustomPermission...487
DandBCompany...490
Dashboard...509
DashboardComponent...515
DashboardComponentFeed..515
DashboardFeed...522
DashboardTag...528
DatacloudCompany...530
DatacloudContact..536
DatacloudDandBCompany..541
DatacloudOwnedEntity...561
DatacloudPurchaseUsage..563
DatacloudSocialHandle..565
DatedConversionRate..566
DeclinedEventRelation...567

Division....569
DivisionLocalization....570
Document....572
DocumentAttachmentMap....577
DocumentTag....578
Domain....579
DomainSite....580
EmailMessage....581
EmailServicesAddress....585
EmailServicesFunction....587
EmailStatus....592
EmailTemplate....594
Entitlement....598
EntitlementContact....602
EntitlementFeed....603
EntitlementHistory....610
EntitlementTemplate....611
EntityHistory....613
EntitySubscription....615
EnvironmentHubMember....617
Event....620
EventFeed....632
EventRelation....639
EventTag....643
EventWhoRelation....645
ExternalDataSource....646
ExternalDataUserAuth....648
FeedComment....649
FeedItem....652
FeedLike....660
FeedPollChoice....661
FeedPollVote....662
FeedPost....663
FeedTrackedChange....667
FieldHistoryArchive....670
FieldPermissions....671
FiscalYearSettings....674
Folder....677
ForecastingAdjustment....680
ForecastingFact....683
ForecastingItem....686
ForecastingQuota....691
ForecastingType....694
ForecastShare....695

GoogleDoc....697
Group....698
GroupMember....701
HashtagDefinition....702
Holiday....703
Idea....707
IdeaComment....714
IdeaReputation....716
IdeaReputationLevel....719
IdeaTheme....720
KnowledgeableUser....722
KnowledgeArticle....723
KnowledgeArticleVersion....725
KnowledgeArticleVersionHistory....732
KnowledgeArticleViewStat....735
KnowledgeArticleVoteStat....737
Lead....738
LeadCleanInfo 755
LeadFeed....769
LeadHistory....776
LeadOwnerSharingRule....778
LeadShare....780
LeadStatus....781
LeadTag....783
LimitAllocationPerApp....784
LineitemOverride....786
LiveAgentSession....789
LiveAgentSessionHistory....793
LiveAgentSessionOwnerSharingRule....794
LiveAgentSessionShare....795
LiveChatButton....797
LiveChatButtonDeployment....805
LiveChatButtonSkill....806
LiveChatDeployment....807
LiveChatTranscript....809
LiveChatTranscriptEvent....817
LiveChatTranscriptHistory....819
LiveChatTranscriptOwnerSharingRule....820
LiveChatTranscriptShare....821
LiveChatTranscriptSkill....823
LiveChatUserConfig....824
LiveChatUserConfigProfile....829
LiveChatUserConfigUser....830
LiveChatVisitor....831

LoginIp....832
LoginHistory....833
MailmergeTemplate....836
MetricsDataFile....838
MilestoneType....841
Name....842
Network....846
NetworkActivityAudit....852
NetworkMember....854
NetworkMemberGroup....860
NetworkModeration....861
NewsFeed....862
Note....869
NoteAndAttachment....871
NoteTag....873
ObjectPermissions....874
ObjectTerritory2AssignmentRule....877
ObjectTerritory2AssignmentRuleItem....879
ObjectTerritory2Association....881
OpenActivity....882
Opportunity....889
OpportunityCompetitor....899
OpportunityContactRole....901
OpportunityFeed....902
OpportunityFieldHistory....909
OpportunityHistory....910
OpportunityLineItem....913
OpportunityLineItemSchedule....919
OpportunityOverride....923
OpportunityOwnerSharingRule....926
OpportunityPartner....928
OpportunityShare....929
OpportunitySplit....931
OpportunitySplitType....933
OpportunityStage....935
OpportunityTag....938
OpportunityTeamMember....939
Order....941
OrderFeed....952
OrderHistory....957
OrderItem....959
OrderItemFeed....962
OrderItemHistory....967
Organization....969

OrgWideEmailAddress....988
OwnedContentDocument....989
PackageLicense....991
Partner....993
PartnerNetworkConnection....995
PartnerNetworkRecordConnection....997
PartnerRole....1001
Period....1002
PermissionSet....1004
PermissionSetAssignment....1009
PermissionSetLicense....1011
PermissionSetLicenseAssign....1014
Pricebook2....1015
Pricebook2History....1018
PricebookEntry....1020
ProcessDefinition....1022
ProcessInstance....1024
ProcessInstanceHistory....1027
ProcessInstanceStep....1030
ProcessInstanceNode....1032
ProcessInstanceWorkitem....1035
ProcessNode....1037
Product2....1038
Product2Feed....1043
ProductEntitlementTemplate....1050
Profile....1051
ProfileSkill....1054
ProfileSkillEndorsement....1056
ProfileSkillEndorsementHistory....1057
ProfileSkillFeed....1058
ProfileSkillHistory....1062
ProfileSkillShare....1063
ProfileSkillUser....1065
ProfileSkillUserHistory....1066
PushTopic....1067
QuantityForecast....1069
QuantityForecastHistory....1077
Question....1079
QuestionDataCategorySelection....1084
QuestionReportAbuse....1086
QuestionSubscription....1087
QueueSobject....1089
QuickText....1089
QuickTextHistory....1091

QuickTextOwnerSharingRule....1092
QuickTextShare....1094
Quote....1095
QuoteDocument....1109
QuoteLineItem....1111
RecentlyViewed....1115
RecordType....1119
RecordTypeLocalization....1122
Reply....1124
ReplyReportAbuse....1126
Report....1127
ReportFeed....1130
ReportTag....1137
RevenueForecast....1138
RevenueForecastHistory....1147
RuleTerritory2Association....1150
Scontrol....1151
ScontrolLocalization....1154
SelfServiceUser....1158
ServiceContract....1161
ServiceContractFeed....1170
ServiceContractHistory....1177
ServiceContractOwnerSharingRule....1178
ServiceContractShare....1180
SetupEntityAccess....1182
SignupRequest....1184
Site....1190
SiteDomain....1195
SiteHistory....1196
Skill....1198
SkillProfile....1199
SkillUser....1200
SlaProcess....1200
Solution....1203
SolutionFeed....1207
SolutionHistory....1214
SolutionStatus....1215
SolutionTag....1217
StaticResource....1218
StreamingChannel....1221
TagDefinition....1222
Task....1224
TaskFeed....1233
TaskPriority....1240

TaskRelation....1241
TaskStatus....1243
TaskTag....1245
TaskWhoRelation....1246
Territory....1247
Territory2....1250
Territory2Model....1253
Territory2ModelHistory....1255
Territory2Type....1256
Topic....1258
TopicAssignment....1259
TopicFeed....1260
UndecidedEventRelation....1266
User....1268
UserAccountTeamMember....1306
UserConfigTransferButton....1309
UserConfigTransferSkill....1309
UserFeed....1310
UserLicense....1317
UserLogin....1320
UserMembershipSharingRule....1321
UserPackageLicense....1322
UserPreference....1323
UserProfile....1324
UserProfileFeed....1343
UserRecordAccess....1350
UserRole....1352
UserShare....1356
UserTeamMember....1358
UserTerritory....1359
UserTerritory2Association....1361
Vote....1362
WebLink....1363
WebLinkLocalization....1369
WorkAccess....1373
WorkAccessShare....1374
WorkBadge....1375
WorkBadgeDefinition....1376
WorkBadgeDefinitionHistory....1379
WorkBadgeDefinitionShare....1380
WorkCoaching....1381
WorkCoachingFeed....1383
WorkCoachingHistory....1388
WorkCoachingShare....1389

WorkFeedback 1391
WorkFeedbackQuestion 1392
WorkFeedbackQuestionHistory 1395
WorkFeedbackQuestionSet 1396
WorkFeedbackQuestionSetShare 1398
WorkFeedbackQuestionShare 1399
WorkFeedbackRequest 1400
WorkFeedbackRequestHistory 1404
WorkFeedbackRequestShare 1405
WorkFeedbackShare 1406
WorkGoal 1408
WorkGoalCollaborator 1414
WorkGoalCollaboratorHistory 1415
WorkGoalFeed 1416
WorkGoalHistory 1422
WorkGoalLink 1423
WorkGoalShare 1424
WorkPerformanceCycle 1425
WorkPerformanceCycleFeed 1427
WorkPerformanceCycleHistory 1432
WorkPerformanceCycleShare 1433
WorkReward 1435
WorkRewardFund 1436
WorkRewardFundHistory 1438
WorkRewardFundShare 1439
WorkRewardFundType 1440
WorkRewardFundTypeHistory 1443
WorkRewardFundTypeShare 1444
WorkRewardHistory 1445
WorkRewardShare 1446
WorkThanks 1447
WorkThanksShare 1448

Chapter 9: Core Calls 1450

convertLead() 1451
LeadConvertResult 1456
create() 1456
SaveResult 1464
delete() 1465
DeleteResult 1467
emptyRecycleBin() 1468
EmptyRecycleBinResult 1470
getDeleted() 1470
GetDeletedResult 1474
getUpdated() 1474

GetUpdatedResult....1477
invalidateSessions()....1478
InvalidateSessionsResult....1479
login()....1480
LoginResult....1484
logout()....1484
merge()....1485
MergeResult....1489
performQuickActions()....1490
PerformQuickActionResult....1491
process()....1491
ProcessResult....1494
query()....1494
QueryResult....1498
QueryLocator....1500
queryAll()....1500
queryMore()....1503
QueryResult....1505
QueryLocator....1506
retrieve()....1506
search()....1508
SearchResult....1511
undelete()....1511
UndeleteResult....1514
update()....1514
SaveResult....1520
upsert()....1520
UpsertResult....1525

Chapter 10: Describe Calls....1526
describeAllTabs()....1527
describeAppMenu()....1528
DescribeAppMenuResult....1529
describeApprovalLayout()....1530
DescribeApprovalLayoutResult....1531
describeAvailableQuickActions()....1531
DescribeAvailableQuickActionResult....1532
describeCompactLayouts()....1533
DescribeCompactLayoutsResult....1535
describeDataCategoryGroups()....1536
DescribeDataCategoryGroupResult....1538
describeDataCategoryGroupStructures()....1538
DescribeDataCategoryGroupStructureResult....1541
describeFlexiPages()....1542
DescribeFlexiPageResult....1543

describeGlobal()....1544
DescribeGlobalResult....1545
describeGlobalTheme()....1547
DescribeGlobalTheme....1549
describeLayout()....1549
DescribeLayoutResult....1553
describePrimaryCompactLayouts()....1564
describeQuickActions()....1565
DescribeQuickActionResult....1566
describeSearchScopeOrder()....1570
DescribeSearchScopeOrderResult....1571
describeSearchLayouts()....1571
DescribeSearchLayoutResult....1572
describeSObject()....1573
describeSObjectResult....1575
describeSObjects()....1576
DescribeSObjectResult....1579
describeSoftphoneLayout()....1589
describeTabs()....1592
describeTabSetResult....1595
describeTheme()....1597
DescribeThemeResult....1598
DescribeThemeItem....1598

Chapter 11: Utility Calls....1599
getServerTimestamp()....1599
getServerTimestampResult....1601
getUserInfo()....1601
getUserInfoResult....1602
resetPassword()....1604
sendEmail()....1605
SendEmailResult....1612
sendEmailMessage()....1613
setPassword()....1615

Chapter 12: SOAP Headers....1617
AllOrNoneHeader....1618
AllowFieldTruncationHeader....1619
AssignmentRuleHeader....1620
CallOptions....1621
DisableFeedTrackingHeader....1622
EmailHeader....1624
LimitInfoHeader....1625
LocaleOptions....1626
LoginScopeHeader....1627

MruHeader....1628
OwnerChangeOptions....1629
PackageVersionHeader....1630
QueryOptions....1631
SessionHeader....1632
UserTerritoryDeleteHeader....1632

USING THE API WITH SALESFORCE FEATURES....1633

Chapter 13: Implementation Considerations....1633
Choosing a User for an Integration....1634
Login Server URL....1634
Log in to the Login Server....1635
Typical API Call Sequence....1635
Salesforce Sandbox....1635
Multiple Instances of Salesforce.com Database Servers....1636
Content Type Requirement....1636
Monitoring API Traffic....1636
API Usage Metering....1636
Compression....1638
HTTP Persistent Connections....1639
HTTP Chunking....1639
Internationalization and Character Sets....1639
XML Compliance....1639
.NET, Non-String Fields, and the Enterprise WSDL....1640

Chapter 14: Outbound Messaging....1641
Understanding Outbound Messaging....1642
Understanding Notifications....1643
Setting Up Outbound Messaging....1643
Setting Up User Profiles....1644
Defining Outbound Messaging....1644
Downloading the Salesforce Client Certificate....1645
Viewing Outbound Messages....1645
Tracking Outbound Message Status....1646
Important Security Considerations....1646
Understanding the Outbound Messaging WSDL....1646
Building a Listener....1648

Chapter 15: Data Loading and Integration....1650
Choosing the Right API....1651
Client Application Design....1651
Salesforce Settings....1652
Best Practices with Any Data Loader....1652
Integration and Single Sign-On....1653

Chapter 16: Data Replication....1654
API Calls for Data Replication....1655
Scope of Data Replication....1655
Data Replication Steps....1655
Object-Specific Requirements for Data Replication....1656
Polling for Changes....1656
Checking for Structural Changes in the Object....1657

Chapter 17: Feature-Specific Considerations....1658
Archived Activities....1659
Person Account Record Types....1659
Opportunity Forecast Override Business Rules....1660
OpportunityOverride Lifecycle....1661
LineitemOverride Object Lifecycle....1663
Call Centers and the API....1664
Implementing Salesforce Integrations on Force.com....1665
Articles....1667
Data Categories....1670

Glossary....1673

Index....1689

GETTING STARTED

Chapter 1

Introducing SOAP API

In this chapter ...

- When to Use the SOAP API
- Customize, Integrate, and Extend Your Salesforce Solutions
- Supported Salesforce Editions
- Standards Compliance
- Development Platforms
- SOAP API Support Policy
- Choosing a WSDL
- Related Resources
- What's New in Version 31.0
- Quick Start

Salesforce provides programmatic access to your organization's information using simple, powerful, and secure application programming interfaces. To use this document, you should have a basic familiarity with software development, Web services, and the Salesforce user interface.

Any functionality described in this guide is available if your organization has the API feature enabled. This feature is enabled by default for Performance, Unlimited, Enterprise, and Developer Editions. Some Professional Edition organizations may also have the API enabled. If you cannot access the features you see in this guide, contact salesforce.com.

Note: Salesforce.com Education Services offers a suite of training courses to enable developers to design, create, integrate, and extend applications built on the Force.com platform. Be sure to visit http://www.salesforce.com/training to learn more.

When to Use the SOAP API

The Salesforce prebuilt applications provide powerful CRM functionality. In addition, Salesforce provides the ability to customize the prebuilt applications to fit your organization. However, your organization may have complex business processes that are unsupported by the existing functionality. When this is the case, the Force.com platform includes a number of ways for advanced administrators and developers to implement custom functionality. These include the SOAP API, Apex, and Visualforce.

SOAP API

Use SOAP API to create, retrieve, update or delete records, such as accounts, leads, and custom objects. With more than 20 different calls, SOAP API also allows you to maintain passwords, perform searches, and much more. Use SOAP API in any language that supports Web services.

REST API

REST API provides a powerful, convenient, and simple REST-based Web services interface for interacting with Salesforce. Its advantages include ease of integration and development, and it's an excellent choice of technology for use with mobile applications and Web projects. However, if you have a large number of records to process, you may wish to use Bulk API, which is based on REST principles and optimized for large sets of data.

Bulk API

Bulk API is based on REST principles and is optimized for loading or deleting large sets of data. You can use it to query, insert, update, upsert, or delete a large number of records asynchronously by submitting batches which are processed in the background by Salesforce.

SOAP API, in contrast, is optimized for real-time client applications that update small numbers of records at a time. Although SOAP API can also be used for processing large numbers of records, when the data sets contain hundreds of thousands of records, it becomes less practical. Bulk API is designed to make it simple to process data from a few thousand to millions of records.

Metadata API

Use Metadata API to retrieve, deploy, create, update, or delete customizations for your organization. The most common use is to migrate changes from a sandbox or testing organization to your production environment. Metadata API is intended for managing customizations and for building tools that can manage the metadata model, not the data itself.

The easiest way to access the functionality in Metadata API is to use the Force.com IDE or Force.com Migration Tool. These tools are built on top of Metadata API and use the standard Eclipse and Ant tools respectively to simplify the task of working with Metadata API. Built on the Eclipse platform, the Force.com IDE provides a comfortable environment for programmers familiar with integrated development environments, allowing you to code, compile, test, and deploy all from within the IDE itself. The Force.com Migration Tool is ideal if you want to use a script or a command-line utility for moving metadata between a local directory and a Salesforce organization.

Apex

Use Apex if you want to:

- Create Web services.
- Create email services.
- Perform complex validation over multiple objects.
- Create complex business processes that are not supported by workflow.
- Create custom transactional logic (logic that occurs over the entire transaction, not just with a single record or object).

- Attach custom logic to another operation, such as saving a record, so that it occurs whenever the operation is executed, regardless of whether it originates in the user interface, a Visualforce page, or from SOAP API.

For more information, see the *Force.com Apex Code Developer's Guide*.

Visualforce

Visualforce consists of a tag-based markup language that gives developers a more powerful way of building applications and customizing the Salesforce user interface. With Visualforce you can:

- Build wizards and other multistep processes.
- Create your own custom flow control through an application.
- Define navigation patterns and data-specific rules for optimal, efficient application interaction.

For more information, see the *Visualforce Developer's Guide*.

Customize, Integrate, and Extend Your Salesforce Solutions

The Force.com platform allows you to customize, integrate, and extend your Salesforce organization using the language and platform of your choice:

- **Customize Salesforce** with custom fields, links, objects, page layouts, buttons, record types, s-controls, and tabs to meet specific business requirements.
- **Integrate Salesforce** with your organization's ERP and finance systems, deliver real-time sales and support information to company portals, and populate critical business systems with customer information.
- **Extend Salesforce** in presentation, business logic, and data services with new functionality that reflects the business requirements of your organization.

For more information about Force.com solutions, developer resources, and community resources, go to Developer Force.

Supported Salesforce Editions

To use SOAP API, your organization must use Enterprise Edition, Performance Edition, Unlimited Edition, or Developer Edition. If you are an existing Salesforce customer and want to upgrade to Enterprise, Unlimited, or Performance Edition, contact your account representative.

To develop Web service client applications, it is strongly recommended that you use Developer Sandbox, which is an exact replica of your Salesforce deployment, including all customization and data. For more information, see `http://www.salesforce.com/products/sandbox.jsp`.

Developer Edition provides access to all of the features available with Enterprise Edition. Developer Edition is constrained only by the number of users and the amount of storage space. Developer Edition provides a development context that allows you to build and test your solutions without affecting your organization's live data. Developer Edition accounts are available for free at `https://developer.salesforce.com/page/Getting_Started`.

Standards Compliance

SOAP API is implemented to comply with the following specifications:

Standard Name	Website
Simple Object Access Protocol (SOAP) 1.1	
Web Service Description Language (WSDL) 1.1	`http://www.w3.org/TR/2001/NOTE-wsdl-20010315`

Standard Name	Website
WS-I Basic Profile 1.1	http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-08-24.html

Development Platforms

SOAP API works with current SOAP development environments, including, but not limited to, Visual Studio .NET 2005. In this document, we provide examples in Java and C# (.NET). The Java examples are based on WSC 20.0 (WSC) and JDK 6 (Java Platform Standard Edition Development Kit 6). Additional versions of WSC are available at `https://github.com/forcedotcom/wsc` and `http://mvnrepository.com/artifact/com.force.api/force-wsc`. To see a complete list of compatible development platforms and more sample code, go to developer.salesforce.com.

Note: Development platforms vary in their SOAP implementations. Implementation differences in certain development platforms might prevent access to some or all of the features in the API. If you are using Visual Studio for .NET development, we recommend that you use Visual Studio 2003 or higher.

SOAP API Support Policy

Salesforce.com recommends that your new client applications use the most recent version of the Force.com WSDL file to fully exploit the benefits of richer features and greater efficiency. You can navigate to the most recent WSDL for your organization from Setup by clicking **Develop** > **API**. When a new version is released, use the following steps in Quick Start to update your WSDL:

- Regenerate the WSDL file (see Step 2: Generate or Obtain the Web Service WSDL)
- Import it into your environment (see Step 3: Import the WSDL File Into Your Development Platform)

Backward Compatibility

Salesforce.com strives to make backward compatibility easy when using the Force.com platform.

Each new Salesforce release consists of two components:

- A new release of platform software that resides on salesforce.com systems
- A new version of SOAP API

For example, the Winter '07 release included SOAP API version 9.0 and the Summer '07 release included SOAP API version 10.0.

We maintain support for each SOAP API version across releases of the platform software. SOAP API is backward compatible in that an application created to work with a given SOAP API version will continue to work with that same SOAP API version in future platform software releases.

Salesforce.com does not guarantee that an application written against one SOAP API version will work with future SOAP API versions: Changes in method signatures and data representations are often required as we continue to enhance SOAP API. However, we strive to keep SOAP API consistent from version to version with minimal if any changes required to port applications to newer SOAP API versions.

For example, an application written using SOAP API version 9.0 which shipped with the Winter '07 release will continue to work with SOAP API version 9.0 on the Summer '07 release and on future releases beyond that. However, that same application may not work with SOAP API version 10 without modifications to the application.

SOAP API End-of-Life

Salesforce.com is committed to supporting each SOAP API version for a minimum of three years from the date of first release. In order to improve the quality and performance of SOAP API, versions that are more than three years old may cease to be supported.

When a SOAP API version is scheduled to be unsupported, an advance end-of-life notice will be given at least one year before support for SOAP API version is ended. Salesforce.com will directly notify customers using SOAP API versions scheduled for end of life.

Choosing a WSDL

There are two Force.com Web services for which you can obtain WSDL files for API access:

- **Force.com Enterprise WSDL**—This API is for most enterprise users who are developing client applications for their organization. The enterprise WSDL file is a strongly typed representation of your organization's data. It provides information about your schema, data types, and fields to your development environment, allowing for a tighter integration between it and the Force.com Web service. This WSDL changes if custom fields or custom objects are added to, renamed, or removed from, your organization's Salesforce configuration. If you are downloading an enterprise WSDL and you have managed packages installed in your organization, you need to take an extra step to select the version of each installed package to include in the generated WSDL.

  Note the following when generating the enterprise WSDL:

  ◊ If new custom fields or objects are added to, renamed, or removed from your organization's information, you need to regenerate the WSDL file in order to access them.
  ◊ The generated WSDL contains the objects and fields in your organization, including those available in the selected versions of each installed package. If a field or object is added in a later package version, you must generate the enterprise WSDL with that package version to work with the object or field in your API integration.

- **Force.com Partner WSDL**—This API is for salesforce.com partners who are developing client applications for multiple organizations. As a loosely-typed representation of the Salesforce object model, the partner WSDL can be used to access data within any organization.

Related Resources

The salesforce.com developer website provides a full suite of developer toolkits, sample code, sample SOAP messages, community-based support, and other resources to help you with your development projects. Be sure to visit `https://developer.salesforce.com/page/Getting_Started` for more information, or visit `https://developer.salesforce.com/signup` to sign up for a free Developer Edition account.

You can visit these websites to find out more about Salesforce applications:

- Salesforce.com for information about the Salesforce application.
- Force.com AppExchange for access to apps created for Salesforce.
- Salesforce.com Community for services to ensure Salesforce customer success.

What's New in Version 31.0

New Objects

For more information about new API objects in version 31.0, see New and Changed Objects in the release notes.

Earlier Versions

Information about changes for earlier versions is available on the Earlier Reference Documentation page on `developer.salesforce.com`.

Quick Start

Use this quick start to create a sample application in your development environment.

Note: Before you begin building an integration or other client application:

- Install your development platform according to its product documentation.
- Read through all the steps before beginning this quick start. You may also wish to review the rest of this document to familiarize yourself with terms and concepts.

Step 1: Obtain a Salesforce Developer Edition Organization

If you are not already a member of the Force.com developer community, go to `developer.salesforce.com/signup` and follow the instructions for signing up for a Developer Edition organization. Even if you already have Enterprise Edition, Unlimited Edition, or Performance Edition, use Developer Edition for developing, staging, and testing your solutions against sample data to protect your organization's live data. This is especially true for applications that insert, update, or delete data (as opposed to simply reading data).

If you already have a Developer Edition organization, verify that you have the "API Enabled" permission. This permission is enabled by default, but may have been changed by an administrator. For more information, see the help in the Salesforce user interface.

Step 2: Generate or Obtain the Web Service WSDL

To access the Force.com Web service, you need a Web Service Description Language (WSDL) file. The WSDL file defines the Web service that is available to you. Your development platform uses this WSDL to generate an API to access the Force.com Web service it defines. You can either obtain the WSDL file from your organization's Salesforce administrator or you can generate it yourself if you have access to the WSDL download page in the Salesforce user interface. You can navigate to the most recent WSDL for your organization from Setup by clicking **Develop** > **API**.

For more information about WSDL, see `http://www.w3.org/TR/wsdl`.

Generating the WSDL File for Your Organization

Any user with the "Modify All Data" permission can download the Web Services Description Language (WSDL) file to integrate and extend Salesforce using the API. (The System Administrator profile has this permission.)

The WSDL file is dynamically generated based on which type of WSDL file (enterprise or partner) you download. The generated WSDL defines all of the API calls, objects (including standard and custom objects), and fields that are available for API access for your organization.

To generate the WSDL file for your organization:

1. Log in to your Enterprise, Unlimited, Performance, or Developer Edition Salesforce account. You must log in as an administrator or as a user who has the "Modify All Data" permission. Logins are checked to ensure they are from a known IP address. For more information, see "Setting Login Restrictions" in the Salesforce online help.
2. From Setup, click **Develop** > **API** to display the WSDL download page.
3. Download the appropriate WSDL:

- If you are downloading an enterprise WSDL and you have managed packages installed in your organization, click **Generate Enterprise WSDL**. Salesforce prompts you to select the version of each installed package to include in the generated WSDL.
- Otherwise, right-click the link for the appropriate WSDL document to save it to a local directory. In the right-click menu, Internet Explorer users can choose **Save Target As**, while Mozilla Firefox users can choose **Save Link As**.

Step 3: Import the WSDL File Into Your Development Platform

Once you have the WSDL file, you need to import it into your development platform so that your development environment can generate the necessary objects for use in building client Web service applications in that environment. This section provides sample instructions for WSC and Microsoft Visual Studio. For instructions about other development platforms, see your platform's product documentation.

Note: The process for importing WSDL files is identical for the enterprise and partner WSDL files.

Instructions for Java Environments (WSC)

Java environments access the API through Java objects that serve as proxies for their server-side counterparts. Before using the API, you must first generate these objects from your organization's WSDL file.

Each SOAP client has its own tool for this process. For WSC, use the `wsdlc` utility.

Note: Before you run `wsdlc`, you must have the WSC JAR file installed on your system and referenced in your classpath.

The basic syntax for `wsdlc` is:

```
java -classpath pathToJAR/wsc-22.jar com.sforce.ws.tools.wsdlc pathToWsdl/WsdlFilename
pathToOutputJar/OutputJarFilename
```

This command generates an output jar file based on the specified WSDL file. After the output jar file is created, reference it along with the wsc jar file (for example, wsc-22.jar) in your Java program to create a client application.

Instructions for Microsoft Visual Studio

Visual Studio languages access the API through objects that serve as proxies for their server-side counterparts. Before using the API, you must first generate these objects from your organization's WSDL file.

Once you have the proxy classes for the server-side objects, you need to ensure that you specify whether you have set any values on non-string fields. For more information, see Implementation Considerations.

Visual Studio provides two approaches for importing your WSDL file and generating an XML Web service client: an IDE-based approach and a command line approach. This walkthrough describes how to import your WSDL file through the IDE.

Note: Before you begin, the first step is to create a new application or open an existing application in Visual Studio. In addition, you need to have generated the WSDL file, as described in Generating the WSDL File for Your Organization.

An XML Web service client is any component or application that references and uses an XML Web service. This does not necessarily need to be a client-based application. In fact, in many cases, your XML Web service clients might be other Web applications, such as Web Forms or even other XML Web services. When accessing XML Web services in managed code, a proxy class and the .NET Framework handle all of the infrastructure coding.

To access an XML Web service from managed code:

1. Name your project `Walkthrough` or change the `using` directive in the following sample to *`your_project_name.web_reference_name`*. Then, add a Web reference to your project for the XML Web service that you want to access. The Web reference creates a proxy class with methods that serve as proxies for each exposed method of the XML Web service.
2. Add the namespace for the Web reference.
3. Create an instance of the proxy class and then access the methods of that class as you would the methods of any other class.

You can add either a .NET 2.0 style Web reference, or a .NET 3.0 style Service reference, depending on your version of Visual Studio and preferred developer environment. A .NET 3.0 style reference uses services like SoapClient instead of SforceService.

To add a Web reference:

Note: These steps may be different depending on the version of Visual Studio that you're using. For more information, see "Adding and Removing Web References" in the Visual Studio documentation.

1. If you are using Visual Studio 2010 or earlier, on the Project menu, choose **Add Web Reference.** For later versions of Visual Studio, on the Project menu, choose **Add Service Reference**, select **Advanced** and then select **Add Web Reference**.
2. In the URL box of the Add Web Reference dialog box, type the URL to obtain the service description of the XML Web service you want to access, such as:

   ```
   c:\WSDLFiles\enterprise.wsdl
   ```

3. Click **Go** to retrieve information about the XML Web service.
4. In the Web reference name box, rename the Web reference to `sforce`, which is the name you will use for this Web reference.
5. Click **Add Reference** to add a Web reference for the target XML Web service.
6. Visual Studio retrieves the service description and generates a proxy class to interface between your application and the XML Web service.

Note: If you are using Visual Basic .Net 1.1 and the enterprise WSDL, you will need to modify the generated Web service client to overcome a bug in Visual Studio's client generation utility. The API exposes two objects (Case and Event) whose names conflict with Visual Basic keywords. When the classes that represent these objects are created, Visual Studio wraps the class names with brackets (`[Case]` and `[Event]`). This is the method by which you can reuse keywords.

Unfortunately, in the definition of the `SObject` class, Visual Studio does not wrap Case and Event to class references in the `System.Xml.Serialization.XmlIncludeAttribute` that are part of the `SObject` definition. To work around this problem in Visual Studio, you need to edit the `XmlIncludeAttribute` settings for Case and Event as shown below. This does not apply to C# and only applies when using the enterprise version of the WSDL.

```
System.Xml.Serialization.XmlIncludeAttribute(GetType([Event])), _

System.Xml.Serialization.XmlIncludeAttribute(GetType([Case])), _
```

Step 4: Walk Through the Sample Code

Once you have imported your WSDL file, you can begin building client applications that use the API. Use the following samples to create a basic client application. Comments embedded in the sample explain each section of code.

Java Sample Code

This section walks through a sample Java client application that uses the WSC SOAP client. The purpose of this sample application is to show the required steps for logging into the login server and to demonstrate the invocation and subsequent handling of several API calls.

To run this sample, you must pass the authentication endpoint URL as an argument for your program. You can obtain this URL from the WSDL file. This sample application performs the following main tasks:

1. Prompts the user for their Salesforce username and password.
2. Calls `login()` to log in to the single login server and, if the login succeeds, retrieves user information and writes it to the console along with session information.
3. Calls `describeGlobal()` to retrieve a list of all available objects for the organization's data. The `describeGlobal` method determines the objects that are available to the logged in user. This call should not be made more than once per session, since the data returned from the call is not likely to change frequently. The DescribeGlobalResult is echoed to the console.
4. Calls `describeSObjects()` to retrieve metadata (field list and object properties) for a specified object. The `describeSObject` method illustrates the type of metadata information that can be obtained for each object available to the user. The sample client application executes a `describeSObjects()` call on the object that the user specifies and then echoes the returned metadata information to the console. Object metadata information includes permissions, field types and lengths, and available values for picklist fields and types for `referenceTo` fields.
5. Calls `query()`, passing a simple query string `("SELECT FirstName, LastName FROM Contact")`, and iterating through the returned QueryResult.
6. Calls `logout()` to the log the user out.

The following sample code uses try/catch blocks to handle exceptions that might be thrown by the API calls.

```
package com.example.samples;

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.InputStreamReader;
import java.io.IOException;
import com.sforce.soap.enterprise.DeleteResult;
import com.sforce.soap.enterprise.DescribeGlobalResult;
import com.sforce.soap.enterprise.DescribeGlobalSObjectResult;
import com.sforce.soap.enterprise.DescribeSObjectResult;
import com.sforce.soap.enterprise.EnterpriseConnection;
import com.sforce.soap.enterprise.Error;
import com.sforce.soap.enterprise.Field;
import com.sforce.soap.enterprise.FieldType;
import com.sforce.soap.enterprise.GetUserInfoResult;
import com.sforce.soap.enterprise.LoginResult;
import com.sforce.soap.enterprise.PicklistEntry;
import com.sforce.soap.enterprise.QueryResult;
import com.sforce.soap.enterprise.SaveResult;
import com.sforce.soap.enterprise.sobject.Account;
import com.sforce.soap.enterprise.sobject.Contact;
import com.sforce.soap.enterprise.sobject.SObject;
import com.sforce.ws.ConnectorConfig;
import com.sforce.ws.ConnectionException;

public class QuickstartApiSample {

   private static BufferedReader reader = new BufferedReader(
         new InputStreamReader(System.in));

   EnterpriseConnection connection;
   String authEndPoint = "";

   public static void main(String[] args) {
      if (args.length < 1) {
         System.out.println("Usage: com.example.samples."
               + "QuickstartApiSamples <AuthEndPoint>");

         System.exit(-1);
      }
```

```
      QuickstartApiSample sample = new QuickstartApiSample(args[0]);
      sample.run();
   }

   public void run() {
      // Make a login call
      if (login()) {
         // Do a describe global
         describeGlobalSample();

         // Describe an object
         describeSObjectsSample();

         // Retrieve some data using a query
         querySample();

         // Log out
         logout();
      }
   }

   // Constructor
   public QuickstartApiSample(String authEndPoint) {
      this.authEndPoint = authEndPoint;
   }

   private String getUserInput(String prompt) {
      String result = "";
      try {
         System.out.print(prompt);
         result = reader.readLine();
      } catch (IOException ioe) {
         ioe.printStackTrace();
      }

      return result;
   }

   private boolean login() {
      boolean success = false;
      String username = getUserInput("Enter username: ");
      String password = getUserInput("Enter password: ");

      try {
         ConnectorConfig config = new ConnectorConfig();
         config.setUsername(username);
         config.setPassword(password);

         System.out.println("AuthEndPoint: " + authEndPoint);
         config.setAuthEndpoint(authEndPoint);

         connection = new EnterpriseConnection(config);
         printUserInfo(config);

         success = true;
      } catch (ConnectionException ce) {
         ce.printStackTrace();
      }

      return success;
   }

   private void printUserInfo(ConnectorConfig config) {
      try {
         GetUserInfoResult userInfo = connection.getUserInfo();

         System.out.println("\nLogging in ...\n");
         System.out.println("UserID: " + userInfo.getUserId());
         System.out.println("User Full Name: " + userInfo.getUserFullName());
```

```
         System.out.println("User Email: " + userInfo.getUserEmail());
         System.out.println();
         System.out.println("SessionID: " + config.getSessionId());
         System.out.println("Auth End Point: " + config.getAuthEndpoint());
         System.out
               .println("Service End Point: " + config.getServiceEndpoint());
         System.out.println();
      } catch (ConnectionException ce) {
         ce.printStackTrace();
      }
   }

   private void logout() {
      try {
         connection.logout();
         System.out.println("Logged out.");
      } catch (ConnectionException ce) {
         ce.printStackTrace();
      }
   }

   /**
    * To determine the objects that are available to the logged-in user, the
    * sample client application executes a describeGlobal call, which returns
    * all of the objects that are visible to the logged-in user. This call
    * should not be made more than once per session, as the data returned from
    * the call likely does not change frequently. The DescribeGlobalResult is
    * simply echoed to the console.
    */
   private void describeGlobalSample() {
      try {
         // describeGlobal() returns an array of object results that
         // includes the object names that are available to the logged-in user.
         DescribeGlobalResult dgr = connection.describeGlobal();

         System.out.println("\nDescribe Global Results:\n");
         // Loop through the array echoing the object names to the console
         for (int i = 0; i < dgr.getSobjects().length; i++) {
            System.out.println(dgr.getSobjects()[i].getName());
         }
      } catch (ConnectionException ce) {
         ce.printStackTrace();
      }
   }

   /**
    * The following method illustrates the type of metadata information that can
    * be obtained for each object available to the user. The sample client
    * application executes a describeSObject call on a given object and then
    * echoes the returned metadata information to the console. Object metadata
    * information includes permissions, field types and length and available
    * values for picklist fields and types for referenceTo fields.
    */
   private void describeSObjectsSample() {
      String objectToDescribe = getUserInput("\nType the name of the object to "
            + "describe (try Account): ");

      try {
         // Call describeSObjects() passing in an array with one object type
         // name
         DescribeSObjectResult[] dsrArray = connection
               .describeSObjects(new String[] { objectToDescribe });

         // Since we described only one sObject, we should have only
         // one element in the DescribeSObjectResult array.
         DescribeSObjectResult dsr = dsrArray[0];

         // First, get some object properties
         System.out.println("\n\nObject Name: " + dsr.getName());
```

```
if (dsr.getCustom())
   System.out.println("Custom Object");
if (dsr.getLabel() != null)
   System.out.println("Label: " + dsr.getLabel());

// Get the permissions on the object

if (dsr.getCreateable())
   System.out.println("Createable");
if (dsr.getDeletable())
   System.out.println("Deleteable");
if (dsr.getQueryable())
   System.out.println("Queryable");
if (dsr.getReplicateable())
   System.out.println("Replicateable");
if (dsr.getRetrieveable())
   System.out.println("Retrieveable");
if (dsr.getSearchable())
   System.out.println("Searchable");
if (dsr.getUndeletable())
   System.out.println("Undeleteable");
if (dsr.getUpdateable())
   System.out.println("Updateable");

System.out.println("Number of fields: " + dsr.getFields().length);

// Now, retrieve metadata for each field
for (int i = 0; i < dsr.getFields().length; i++) {
   // Get the field
   Field field = dsr.getFields()[i];

   // Write some field properties
   System.out.println("Field name: " + field.getName());
   System.out.println("\tField Label: " + field.getLabel());

   // This next property indicates that this
   // field is searched when using
   // the name search group in SOSL
   if (field.getNameField())
      System.out.println("\tThis is a name field.");

   if (field.getRestrictedPicklist())
      System.out.println("This is a RESTRICTED picklist field.");

   System.out.println("\tType is: " + field.getType());

   if (field.getLength() > 0)
      System.out.println("\tLength: " + field.getLength());

   if (field.getScale() > 0)
      System.out.println("\tScale: " + field.getScale());

   if (field.getPrecision() > 0)
      System.out.println("\tPrecision: " + field.getPrecision());

   if (field.getDigits() > 0)
      System.out.println("\tDigits: " + field.getDigits());

   if (field.getCustom())
      System.out.println("\tThis is a custom field.");

   // Write the permissions of this field
   if (field.getNillable())
      System.out.println("\tCan be nulled.");
   if (field.getCreateable())
      System.out.println("\tCreateable");
   if (field.getFilterable())
      System.out.println("\tFilterable");
```

```
            if (field.getUpdateable())
               System.out.println("\tUpdateable");

            // If this is a picklist field, show the picklist values
            if (field.getType().equals(FieldType.picklist)) {
               System.out.println("\t\tPicklist values: ");
               PicklistEntry[] picklistValues = field.getPicklistValues();

               for (int j = 0; j < field.getPicklistValues().length; j++) {
                  System.out.println("\t\tValue: "
                        + picklistValues[j].getValue());
               }
            }

            // If this is a foreign key field (reference),
            // show the values
            if (field.getType().equals(FieldType.reference)) {
               System.out.println("\tCan reference these objects:");
               for (int j = 0; j < field.getReferenceTo().length; j++) {
                  System.out.println("\t\t" + field.getReferenceTo()[j]);
               }
            }
            System.out.println("");
         }
      } catch (ConnectionException ce) {
         ce.printStackTrace();
      }
   }

   private void querySample() {
      String soqlQuery = "SELECT FirstName, LastName FROM Contact";
      try {
         QueryResult qr = connection.query(soqlQuery);
         boolean done = false;

         if (qr.getSize() > 0) {
            System.out.println("\nLogged-in user can see "
                  + qr.getRecords().length + " contact records.");

            while (!done) {
               System.out.println("");
               SObject[] records = qr.getRecords();
               for (int i = 0; i < records.length; ++i) {
                  Contact con = (Contact) records[i];
                  String fName = con.getFirstName();
                  String lName = con.getLastName();

                  if (fName == null) {
                     System.out.println("Contact " + (i + 1) + ": " + lName);
                  } else {
                     System.out.println("Contact " + (i + 1) + ": " + fName
                           + " " + lName);
                  }
               }

               if (qr.isDone()) {
                  done = true;
               } else {
                  qr = connection.queryMore(qr.getQueryLocator());
               }
            }
         } else {
            System.out.println("No records found.");
         }
      } catch (ConnectionException ce) {
         ce.printStackTrace();
      }
   }
}
```

C# Sample Code

This section walks through a sample C# client application. The purpose of this sample application is to show the required steps for logging in and to demonstrate the invocation and subsequent handling of several API calls.

This sample application performs the following main tasks:

1. Prompts the user for their Salesforce username and password.
2. Calls `login()` to log in to the single login server and, if the login succeeds:
   - Sets the returned `sessionId` into the session header, which is required for session authentication on subsequent API calls.
   - Resets the Force.com endpoint to the returned `serverUrl`, which is the target of subsequent API calls.

     All client applications that access the API must complete the tasks in this step before attempting any subsequent API calls.
   - Retrieves user information and writes it to the console along with session information.
3. Calls `describeGlobal()` to retrieve a list of all available objects for the organization's data. The `describeGlobal` method determines the objects that are available to the logged in user. This call should not be made more than once per session, since the data returned from the call is not likely to change frequently. The DescribeGlobalResult is echoed to the console.
4. Calls `describeSObjects()` to retrieve metadata (field list and object properties) for a specified object. The `describeSObject` method illustrates the type of metadata information that can be obtained for each object available to the user. The sample client application executes a `describeSObjects()` call on the object that the user specifies and then echoes the returned metadata information to the console. Object metadata information includes permissions, field types and lengths, and available values for picklist fields and types for `referenceTo` fields.
5. Calls `query()`, passing a simple query string `("SELECT FirstName, LastName FROM Contact")`, and iterating through the returned QueryResult.
6. Calls `logout()` to the log the user out.

The following sample code uses try/catch blocks to handle exceptions that might be thrown by the API calls.

The following code begins the sample C# client application.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Web.Services.Protocols;
using Walkthrough.sforce;

namespace Walkthrough
{

   class QuickstartApiSample
   {
      private SforceService binding;

      [STAThread]
      static void Main(string[] args)
      {
         QuickstartApiSample sample = new QuickstartApiSample();
         sample.run();
      }

      public void run()
      {
         // Make a login call
         if (login())
         {
            // Do a describe global
            describeGlobalSample();
```

```
            // Describe an account object
            describeSObjectsSample();

            // Retrieve some data using a query
            querySample();

            // Log out
            logout();
         }
      }

      private bool login()
      {
         Console.Write("Enter username: ");
         string username = Console.ReadLine();
         Console.Write("Enter password: ");
         string password = Console.ReadLine();

         // Create a service object
         binding = new SforceService();

         // Timeout after a minute
         binding.Timeout = 60000;

         // Try logging in
         LoginResult lr;
         try
         {

            Console.WriteLine("\nLogging in...\n");
            lr = binding.login(username, password);
         }

         // ApiFault is a proxy stub generated from the WSDL contract when
         // the web service was imported
         catch (SoapException e)
         {
            // Write the fault code to the console
            Console.WriteLine(e.Code);

            // Write the fault message to the console
            Console.WriteLine("An unexpected error has occurred: " + e.Message);

            // Write the stack trace to the console
            Console.WriteLine(e.StackTrace);

            // Return False to indicate that the login was not successful
            return false;
         }



         // Check if the password has expired
         if (lr.passwordExpired)
         {
            Console.WriteLine("An error has occurred. Your password has expired.");
            return false;
         }


         /** Once the client application has logged in successfully, it will use
          * the results of the login call to reset the endpoint of the service
          * to the virtual server instance that is servicing your organization
          */
         // Save old authentication end point URL
         String authEndPoint = binding.Url;
         // Set returned service endpoint URL
         binding.Url = lr.serverUrl;
```

```
    /** The sample client application now has an instance of the SforceService
     * that is pointing to the correct endpoint. Next, the sample client
     * application sets a persistent SOAP header (to be included on all
     * subsequent calls that are made with SforceService) that contains the
     * valid sessionId for our login credentials. To do this, the sample
     * client application creates a new SessionHeader object and persist it to
     * the SforceService. Add the session ID returned from the login to the
     * session header
     */
    binding.SessionHeaderValue = new SessionHeader();
    binding.SessionHeaderValue.sessionId = lr.sessionId;

    printUserInfo(lr, authEndPoint);

    // Return true to indicate that we are logged in, pointed
    // at the right URL and have our security token in place.
    return true;
  }

  private void printUserInfo(LoginResult lr, String authEP)
  {
    try
    {
      GetUserInfoResult userInfo = lr.userInfo;

      Console.WriteLine("\nLogging in ...\n");
      Console.WriteLine("UserID: " + userInfo.userId);
      Console.WriteLine("User Full Name: " +
          userInfo.userFullName);
      Console.WriteLine("User Email: " +
          userInfo.userEmail);
      Console.WriteLine();
      Console.WriteLine("SessionID: " +
          lr.sessionId);
      Console.WriteLine("Auth End Point: " +
          authEP);
      Console.WriteLine("Service End Point: " +
          lr.serverUrl);
      Console.WriteLine();
    }
    catch (SoapException e)
    {
      Console.WriteLine("An unexpected error has occurred: " + e.Message +
          " Stack trace: " + e.StackTrace);
    }
  }

  private void logout()
  {
    try
    {
      binding.logout();
      Console.WriteLine("Logged out.");
    }
    catch (SoapException e)
    {
      // Write the fault code to the console
      Console.WriteLine(e.Code);

      // Write the fault message to the console
      Console.WriteLine("An unexpected error has occurred: " + e.Message);

      // Write the stack trace to the console
      Console.WriteLine(e.StackTrace);
    }
  }

  /**
```

```
 * To determine the objects that are available to the logged-in
 * user, the sample client application executes a describeGlobal
 * call, which returns all of the objects that are visible to
 * the logged-in user. This call should not be made more than
 * once per session, as the data returned from the call likely
 * does not change frequently. The DescribeGlobalResult is
 * simply echoed to the console.
 */
private void describeGlobalSample()
{
   try
   {
      // describeGlobal() returns an array of object results that
      // includes the object names that are available to the logged-in user.
      DescribeGlobalResult dgr = binding.describeGlobal();

      Console.WriteLine("\nDescribe Global Results:\n");
      // Loop through the array echoing the object names to the console

      for (int i = 0; i < dgr.sobjects.Length; i++)
      {
         Console.WriteLine(dgr.sobjects[i].name);
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An exception has occurred: " + e.Message +
          "\nStack trace: " + e.StackTrace);
   }
}

/**
 * The following method illustrates the type of metadata
 * information that can be obtained for each object available
 * to the user. The sample client application executes a
 * describeSObject call on a given object and then echoes
 * the returned metadata information to the console. Object
 * metadata information includes permissions, field types
 * and length and available values for picklist fields
 * and types for referenceTo fields.
 */
private void describeSObjectsSample()
{
   Console.Write("\nType the name of the object to " +
        "describe (try Account): ");
   string objectType = Console.ReadLine();
   try
   {

      // Call describeSObjects() passing in an array with one object type name
      DescribeSObjectResult[] dsrArray =
            binding.describeSObjects(new string[] { objectType });

      // Since we described only one sObject, we should have only
      // one element in the DescribeSObjectResult array.
      DescribeSObjectResult dsr = dsrArray[0];

      // First, get some object properties
      Console.WriteLine("\n\nObject Name: " + dsr.name);

      if (dsr.custom) Console.WriteLine("Custom Object");
      if (dsr.label != null) Console.WriteLine("Label: " + dsr.label);

      // Get the permissions on the object
      if (dsr.createable) Console.WriteLine("Createable");
      if (dsr.deletable) Console.WriteLine("Deleteable");
      if (dsr.queryable) Console.WriteLine("Queryable");
      if (dsr.replicateable) Console.WriteLine("Replicateable");
      if (dsr.retrieveable) Console.WriteLine("Retrieveable");
```

```
            if (dsr.searchable) Console.WriteLine("Searchable");
            if (dsr.undeletable) Console.WriteLine("Undeleteable");
            if (dsr.updateable) Console.WriteLine("Updateable");

            Console.WriteLine("Number of fields: " + dsr.fields.Length);

            // Now, retrieve metadata for each field
            for (int i = 0; i < dsr.fields.Length; i++)
            {
               // Get the field
               Field field = dsr.fields[i];

               // Write some field properties
               Console.WriteLine("Field name: " + field.name);
               Console.WriteLine("\tField Label: " + field.label);

               // This next property indicates that this
               // field is searched when using
               // the name search group in SOSL
               if (field.nameField)
                  Console.WriteLine("\tThis is a name field.");

               if (field.restrictedPicklist)
                  Console.WriteLine("This is a RESTRICTED picklist field.");

               Console.WriteLine("\tType is: " + field.type.ToString());

               if (field.length > 0)
                  Console.WriteLine("\tLength: " + field.length);

               if (field.scale > 0)
                  Console.WriteLine("\tScale: " + field.scale);

               if (field.precision > 0)
                  Console.WriteLine("\tPrecision: " + field.precision);

               if (field.digits > 0)
                  Console.WriteLine("\tDigits: " + field.digits);

               if (field.custom)
                  Console.WriteLine("\tThis is a custom field.");

               // Write the permissions of this field
               if (field.nillable) Console.WriteLine("\tCan be nulled.");
               if (field.createable) Console.WriteLine("\tCreateable");
               if (field.filterable) Console.WriteLine("\tFilterable");
               if (field.updateable) Console.WriteLine("\tUpdateable");

               // If this is a picklist field, show the picklist values
               if (field.type.Equals(fieldType.picklist))
               {
                  Console.WriteLine("\tPicklist Values");
                  for (int j = 0; j < field.picklistValues.Length; j++)
                     Console.WriteLine("\t\t" + field.picklistValues[j].value);
               }

               // If this is a foreign key field (reference),
               // show the values
               if (field.type.Equals(fieldType.reference))
               {
                  Console.WriteLine("\tCan reference these objects:");
                  for (int j = 0; j < field.referenceTo.Length; j++)
                     Console.WriteLine("\t\t" + field.referenceTo[j]);
               }
               Console.WriteLine("");
            }
         }
         catch (SoapException e)
         {
```

```
            Console.WriteLine("An exception has occurred: " + e.Message +
                "\nStack trace: " + e.StackTrace);
         }
         Console.WriteLine("Press ENTER to continue...");
         Console.ReadLine();
      }

      private void querySample()
      {
         String soqlQuery = "SELECT FirstName, LastName FROM Contact";
         try
         {
            QueryResult qr = binding.query(soqlQuery);
            bool done = false;

            if (qr.size > 0)
            {
               Console.WriteLine("Logged-in user can see "
                     + qr.records.Length + " contact records.");

               while (!done)
               {
                  Console.WriteLine("");
                  sObject[] records = qr.records;
                  for (int i = 0; i < records.Length; i++)
                  {
                     Contact con = (Contact)records[i];
                     string fName = con.FirstName;
                     string lName = con.LastName;
                     if (fName == null)
                        Console.WriteLine("Contact " + (i + 1) + ": " + lName);
                     else
                        Console.WriteLine("Contact " + (i + 1) + ": " + fName
                              + " " + lName);
                  }

                  if (qr.done)
                  {
                     done = true;
                  }
                  else
                  {
                     qr = binding.queryMore(qr.queryLocator);
                  }
               }
            }
            else
            {
               Console.WriteLine("No records found.");
            }
         }
         catch (Exception ex)
         {
            Console.WriteLine("\nFailed to execute query succesfully," +
                "error message was: \n{0}", ex.Message);
         }
         Console.WriteLine("\nPress ENTER to continue...");
         Console.ReadLine();
      }
   }
}
```

The following C# example is the same as the previous C# example, except it uses .NET 3.0 SoapClient services instead of .NET 2.0 SforceService services.

```
using System;
using System.Collections.Generic;
```

```
using System.Linq;
using System.Text;
using System.Threading.Tasks;

using System.ServiceModel;
using Walkthrough.sforce;

namespace Walkthrough
{
    class QuickstartApiSample
    {
        private static SoapClient loginClient; // for login endpoint
        private static SoapClient client; // for API endpoint
        private static SessionHeader header;
        private static EndpointAddress endpoint;

        static void Main(string[] args)
        {
            QuickstartApiSample sample = new QuickstartApiSample();
            sample.run();
        }

        public void run()
        {
            // Make a login call
            if (login())
            {
                // Do a describe global
                describeGlobalSample();

                // Describe an account object
                describeSObjectsSample();

                // Retrieve some data using a query
                querySample();

                // Log out
                logout();
            }
        }

        private bool login()
        {
            Console.Write("Enter username: ");
            string username = Console.ReadLine();
            Console.Write("Enter password: ");
            string password = Console.ReadLine();

            // Create a SoapClient specifically for logging in
            loginClient = new SoapClient();

            // (combine pw and token if necessary)
            LoginResult lr;
            try
            {
                Console.WriteLine("\nLogging in...\n");
                lr = loginClient.login(null, username, password);
            }
            catch (Exception e)
            {
                // Write the fault message to the console
                Console.WriteLine("An unexpected error has occurred: " + e.Message);

                // Write the stack trace to the console
                Console.WriteLine(e.StackTrace);
                return false;
            }

            // Check if the password has expired
```

```
            if (lr.passwordExpired)
            {
                Console.WriteLine("An error has occurred. Your password has expired.");
                return false;
            }

            /** Once the client application has logged in successfully, it will use
             * the results of the login call to reset the endpoint of the service
             * to the virtual server instance that is servicing your organization
             */

            // On successful login, cache session info and API endpoint info
            endpoint = new EndpointAddress(lr.serverUrl);

            /** The sample client application now has a cached EndpointAddress
            * that is pointing to the correct endpoint. Next, the sample client
            * application sets a persistent SOAP header that contains the
            * valid sessionId for our login credentials. To do this, the sample
            * client application creates a new SessionHeader object. Add the session
            * ID returned from the login to the session header
            */
            header = new SessionHeader();
            header.sessionId = lr.sessionId;

            // Create and cache an API endpoint client
            client = new SoapClient("Soap", endpoint);

            printUserInfo(lr, lr.serverUrl);

            // Return true to indicate that we are logged in, pointed
            // at the right URL and have our security token in place.
            return true;
        }

        private void printUserInfo(LoginResult lr, String authEP)
        {
            try
            {
                GetUserInfoResult userInfo = lr.userInfo;

                Console.WriteLine("\nLogging in ...\n");
                Console.WriteLine("UserID: " + userInfo.userId);
                Console.WriteLine("User Full Name: " +
                    userInfo.userFullName);
                Console.WriteLine("User Email: " +
                    userInfo.userEmail);
                Console.WriteLine();
                Console.WriteLine("SessionID: " +
                    lr.sessionId);
                Console.WriteLine("Auth End Point: " +
                    authEP);
                Console.WriteLine("Service End Point: " +
                    lr.serverUrl);
                Console.WriteLine();
            }
            catch (Exception e)
            {
                Console.WriteLine("An unexpected error has occurred: " + e.Message +
                    " Stack trace: " + e.StackTrace);
            }
        }

        private void logout()
        {
            try
            {
                client.logout(header);
                Console.WriteLine("Logged out.");
            }
```

```
            catch (Exception e)
            {
                // Write the fault message to the console
                Console.WriteLine("An unexpected error has occurred: " + e.Message);

                // Write the stack trace to the console
                Console.WriteLine(e.StackTrace);
            }
        }

        /**
        * To determine the objects that are available to the logged-in
        * user, the sample client application executes a describeGlobal
        * call, which returns all of the objects that are visible to
        * the logged-in user. This call should not be made more than
        * once per session, as the data returned from the call likely
        * does not change frequently. The DescribeGlobalResult is
        * simply echoed to the console.
        */
        private void describeGlobalSample()
        {
            try
            {
                // describeGlobal() returns an array of object results that
                // includes the object names that are available to the logged-in user.
                DescribeGlobalResult dgr = client.describeGlobal(
                          header, // session header
                          null // package version header
                          );

                Console.WriteLine("\nDescribe Global Results:\n");
                // Loop through the array echoing the object names to the console

                for (int i = 0; i < dgr.sobjects.Length; i++)
                {
                    Console.WriteLine(dgr.sobjects[i].name);
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("An exception has occurred: " + e.Message +
                    "\nStack trace: " + e.StackTrace);
            }
        }

        /**
        * The following method illustrates the type of metadata
        * information that can be obtained for each object available
        * to the user. The sample client application executes a
        * describeSObject call on a given object and then echoes
        * the returned metadata information to the console. Object
        * metadata information includes permissions, field types
        * and length and available values for picklist fields
        * and types for referenceTo fields.
        */
        private void describeSObjectsSample()
        {
            Console.Write("\nType the name of the object to " +
                 "describe (try Account): ");
            string objectType = Console.ReadLine();
            try
            {

                // Call describeSObjects() passing in an array with one object type name
                DescribeSObjectResult[] dsrArray =
                      client.describeSObjects(
                        header, // session header
                        null, // package version header
                        null, // locale options
```

```
                new string[] { objectType } // object name array
                );

        // Since we described only one sObject, we should have only
        // one element in the DescribeSObjectResult array.
        DescribeSObjectResult dsr = dsrArray[0];

        // First, get some object properties
        Console.WriteLine("\n\nObject Name: " + dsr.name);

        if (dsr.custom) Console.WriteLine("Custom Object");
        if (dsr.label != null) Console.WriteLine("Label: " + dsr.label);

        // Get the permissions on the object
        if (dsr.createable) Console.WriteLine("Createable");
        if (dsr.deletable) Console.WriteLine("Deleteable");
        if (dsr.queryable) Console.WriteLine("Queryable");
        if (dsr.replicateable) Console.WriteLine("Replicateable");
        if (dsr.retrieveable) Console.WriteLine("Retrieveable");
        if (dsr.searchable) Console.WriteLine("Searchable");
        if (dsr.undeletable) Console.WriteLine("Undeleteable");
        if (dsr.updateable) Console.WriteLine("Updateable");

        Console.WriteLine("Number of fields: " + dsr.fields.Length);

        // Now, retrieve metadata for each field
        for (int i = 0; i < dsr.fields.Length; i++)
        {
            // Get the field
            Field field = dsr.fields[i];

            // Write some field properties
            Console.WriteLine("Field name: " + field.name);
            Console.WriteLine("\tField Label: " + field.label);

            // This next property indicates that this
            // field is searched when using
            // the name search group in SOSL
            if (field.nameField)
                Console.WriteLine("\tThis is a name field.");

            if (field.restrictedPicklist)
                Console.WriteLine("This is a RESTRICTED picklist field.");

            Console.WriteLine("\tType is: " + field.type.ToString());

            if (field.length > 0)
                Console.WriteLine("\tLength: " + field.length);

            if (field.scale > 0)
                Console.WriteLine("\tScale: " + field.scale);

            if (field.precision > 0)
                Console.WriteLine("\tPrecision: " + field.precision);

            if (field.digits > 0)
                Console.WriteLine("\tDigits: " + field.digits);

            if (field.custom)
                Console.WriteLine("\tThis is a custom field.");

            // Write the permissions of this field
            if (field.nillable) Console.WriteLine("\tCan be nulled.");
            if (field.createable) Console.WriteLine("\tCreateable");
            if (field.filterable) Console.WriteLine("\tFilterable");
            if (field.updateable) Console.WriteLine("\tUpdateable");

            // If this is a picklist field, show the picklist values
            if (field.type.Equals(fieldType.picklist))
```

```
                    {
                        Console.WriteLine("\tPicklist Values");
                        for (int j = 0; j < field.picklistValues.Length; j++)
                            Console.WriteLine("\t\t" + field.picklistValues[j].value);
                    }

                    // If this is a foreign key field (reference),
                    // show the values
                    if (field.type.Equals(fieldType.reference))
                    {
                        Console.WriteLine("\tCan reference these objects:");
                        for (int j = 0; j < field.referenceTo.Length; j++)
                            Console.WriteLine("\t\t" + field.referenceTo[j]);
                    }
                    Console.WriteLine("");
                }
            }
            catch (Exception e)
            {
                Console.WriteLine("An exception has occurred: " + e.Message +
                    "\nStack trace: " + e.StackTrace);
            }
            Console.WriteLine("Press ENTER to continue...");
            Console.ReadLine();
        }

        private void querySample()
        {
            String soqlQuery = "SELECT FirstName, LastName FROM Contact";
            try
            {
                QueryResult qr = client.query(
                    header, // session header
                    null, // query options
                    null, // mru options
                    null, // package version header
                    soqlQuery // query string
                    );

                bool done = false;

                if (qr.size > 0)
                {
                    Console.WriteLine("Logged-in user can see "
                        + qr.records.Length + " contact records.");

                    while (!done)
                    {
                        Console.WriteLine("");
                        sObject[] records = qr.records;
                        for (int i = 0; i < records.Length; i++)
                        {
                            Contact con = (Contact)records[i];
                            string fName = con.FirstName;
                            string lName = con.LastName;
                            if (fName == null)
                                Console.WriteLine("Contact " + (i + 1) + ": " + lName);
                            else
                                Console.WriteLine("Contact " + (i + 1) + ": " + fName
                                   + " " + lName);
                        }

                        if (qr.done)
                        {
                            done = true;
                        }
                        else
                        {
                            qr = client.queryMore(
```

```
                                    header, // session header
                                    null, // query options
                                    qr.queryLocator // query locator
                                    );
                        }
                    }
                }
                else
                {
                   Console.WriteLine("No records found.");
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine("\nFailed to execute query succesfully," +
                    "error message was: \n{0}", ex.Message);
            }
            Console.WriteLine("\nPress ENTER to continue...");
            Console.ReadLine();
        }
    }
}
```

Chapter 2

Standard and Custom Object Basics

In this chapter ...

- Primitive Data Types
- Field Types
- Compound Fields—Beta
- API Data Types and Salesforce Field Types
- Core Data Types Used in API Calls
- System Fields
- Required Fields
- Frequently-Occurring Fields
- API Field Properties
- Relationships Among Objects
- Relabeling Fields and Tabs and the API
- Force.com AppExchange Object Prefixes and the API
- Custom Objects

Generally speaking, API objects represent database tables that contain your organization's information. For example, the central object in the Salesforce data model represents accounts—companies and organizations involved with your business, such as customers, partners, and competitors. The term “record” describes a particular occurrence of an object (such as a specific account like “IBM” or “United Airlines” that is represented by an Account object). A record is analogous to a row in a database table.

Objects already created for you by Salesforce are called standard objects. Objects you create in your organization are called custom objects.

While this document describes all of the objects available in the API, your applications work with only the objects that you are authorized to access. Programmatic access to objects is determined by the objects defined in your organization, your organization configuration, your user permissions and access settings (which are configured by your organization’s system administrator), your data sharing model, and other factors related specifically to the object.

Most of the objects accessible through the API are read-write objects. However, there are a few objects that are read-only. This fact is noted in the description for the object.

Note: The objects Pricebook and Product are no longer available, and have been removed from this document.

Primitive Data Types

The API uses the following primitive data types:

Value	Details
base64	Base 64-encoded binary data. Fields of this type are used for storing binary files in Attachment records, Document records, and Scontrol records. In these objects, the `Body` or `Binary` field contains the (base64 encoded) data, while the `BodyLength` field defines the length of the data in the `Body` or `Binary` field. In the Document object, you can specify a URL to the document instead of storing the document directly in the record.
boolean	Boolean fields have one of these values: `true` (or 1), or `false` (or 0).
byte	A set of bits.
date	Date data. Fields of this type contain date values, such as `ActivityDate` in the Event object. Unlike dateTime fields, date fields contain no time value—the time portion of a date field is not relevant and is always set to midnight in the Coordinated Universal Time (UTC) time zone. If you specify a date value in a query, you can filter on date fields only.
dateTime	Date/time values (timestamps). Fields of this type handle date/time values (timestamps), such as `ActivityDateTime` in the Event object or the `CreatedDate`, `LastModifiedDate`, or `SystemModstamp` in many objects. Regular dateTime fields are full timestamps with a precision of one second. They are always transferred in the Coordinated Universal Time (UTC) time zone. In your client application, you might need to translate the timestamp to or from a local time zone. If you specify a dateTime value in a query, you can filter on dateTime fields only. Development tools differ in the way that they handle time data. Some development tools report the local time, while others report only the Coordinated Universal Time (UTC) time zone. To determine how your development tool handles time values, refer to its documentation. **Note:** The Event object has a `DurationInMinutes` field that specifies the number of minutes for an event. Even though this is a temporal value, it is an integer type—not a dateTime type.
double	Double values. Fields of this type can contain fractional portions (digits to the right of the decimal place), such as `ConversionRate` in CurrencyType. In the API, all non-integer values (such as Currency Field Type and Percent Field Type) contain values of type double. Some restrictions may be applied to double values: • `scale`: Maximum number of digits to the right of the decimal place. • `precision`: Total number of digits, including those to the left and the right of the decimal place The maximum number of digits to the left of the decimal place is equal to `precision` minus `scale`. In the online application, precision is defined differently—it is the maximum number of digits allowed to the left of the decimal place. Values can be stored in scientific notation if the number is large enough (or, for negative numbers, small enough), as indicated by the W3C XML Schema Part 2: Datatypes Second Edition specification.

Value	Details
	Warning: When the user sets the precision in custom fields in the Salesforce application, it displays the precision set by the user, even if the user enters a more precise value than defined for those fields. However, when you set the precision in custom fields using the API, no rounding occurs when the user retrieves the number field.
int	Fields of this type contain numbers with no fractional portion (digits to the right of a decimal place), such as the `NumberOfEmployees` in an Account. For integer fields, the digits field specifies the maximum number of digits that an integer can have.
string	Character strings. Fields that are of data type `string` contain text and some have length restrictions depending on the data being stored. For example, in the Contact object, the `FirstName` field is 40 characters, the `LastName` field is 80 characters, the `MailingStreet` is 255 characters. **Note:** For fields that contain strings, behavior is different beginning with API version 15.0. In API versions previous to 15.0, if you specify a value for a field, and that value is too large, the value is truncated. For API version 15.0 and later, if a value is specified that is too large, the operation fails and the fault code `STRING_TOO_LONG` is returned. AllowFieldTruncationHeader allows you to specify that the previous behavior, truncation, be used instead of the new behavior in API versions 15.0 and later. This header has no effect in versions 14.0 and earlier. The affected fields are: anyType, email, encryptedstring, multipicklist, phone, picklist, string, and textarea.
time	Time values. Fields of this type handle time values, such as `FridayEndTime` in the BusinessHours object. Development tools differ in the way that they handle time data. Some development tools report the local time, while others report only the Coordinated Universal Time (UTC) time zone. To determine how your development tool handles time values, refer to its documentation.

These data types are used in the SOAP messages that are exchanged between your client application and the API. When writing your client application, follow the data typing rules defined for your programming language and development environment. Your development tool handles the mapping of typed data in your programming language with these SOAP data types.

The primitive data types are:

- specified in the World Wide Web Consortium's publication *XML Schema Part 2: Data Types* at the following URL: `http://www.w3.org/TR/xmlschema-2/`.
- enumerated in the `SOAPType` field of the Field type, which is described in the `fields` property of the DescribeSObjectResult.

Primitive types are used as a standardized way to define, send, receive, and interpret basic data types in the SOAP messages exchanged between client applications and the API. In addition, primitive data types are interpreted in a Salesforce-specific way, which is useful for display formatting and for numeric conversion (adding values of different currencies).

For example, Salesforce chooses to interpret a double value passed via SOAP as a `double` in a number of possible ways, depending on the field definition. If the field type for that data is currency, Salesforce handles the display of the data by prepending it with a currency symbol and inserting a decimal for precision. Similarly, if the field type is percent, Salesforce handles the display of the data by appending a percent sign (%). Regardless of the field type, however, the value is sent in the SOAP message as a double.

The API uses data types called field types that are defined in the WSDLs. For more information, see Field Types.

Field Types

In addition to the primitive data types, the API defines the following data types for fields:

Note: For fields that contain strings, behavior is different beginning with API version 15.0. In API versions previous to 15.0, if you specify a value for a field, and that value is too large, the value is truncated. For API version 15.0 and later, if a value is specified that is too large, the operation fails and the fault code `STRING_TOO_LONG` is returned. AllowFieldTruncationHeader allows you to specify that the previous behavior, truncation, be used instead of the new behavior in API versions 15.0 and later. This header has no effect in versions 14.0 and earlier. The affected fields are: anyType, email, encryptedstring, multipicklist, phone, picklist, string, and textarea.

Field Type	What the Field Contains
address	A compound data type that contains address field data. See Address Compound Fields.
anyType	Polymorphic data type that returns string, picklist, reference, Boolean, currency, int, double, percent, ID, date, datetime, url, or email data depending on the kind of field involved. See AnyType Field Type.
calculated	Fields that are defined by a formula. See Calculated Field Type.
combobox	A combobox, which includes a set of enumerated values and allows the user to specify a value not in the list. See ComboBox Field Type.
currency	Currency values. See Currency Field Type.
DataCategoryGroupReference	Reference to a data category group or a category unique name. See DataCategoryGroupReference Field Type.
email	Email addresses. See Email Field Type.
encryptedstring	Encrypted text fields contain any combination of letters, numbers, or symbols that are stored in encrypted form. You can set a maximum length of up to 175 characters. Available in API versions 11.0 and later.
ID	Primary key field for the object. See ID Field Type. Note that most Web services tools, including .NET andWSC, map the ID simple type defined in the API WSDL (Enterprise or Partner) to a string. However, other tools generate a specific ID class to represent the ID simple type. Please consult your Web services toolkit documentation for more information.
location	A compound data type that contains latitude and longitude values for geolocation fields. See Geolocation Compound Fields.
masterrecord	When records are merged, the ID of the record that is saved (the other records are deleted).
multipicklist	Multi-select picklists, which include a set of enumerated values from which multiple values can be selected. See Multi-Select Picklist Field Type.
percent	Percentage values. See Percent Field Type.
phone	Phone numbers. Values can include alphabetic characters. Client applications are responsible for phone number formatting. See Phone Field Type.
picklist	Picklists, which include a set of enumerated values from which one value can be selected. See Picklist Field Type.

Field Type	What the Field Contains
reference	Cross-references to a different object. Analogous to a foreign key field in SQL. See Reference Field Type.
textarea	String that is displayed as a multiline text field. See Textarea Field Type.
url	URL values. Client applications should commonly display these as hyperlinks. See URL Field Type.

These field types extend primitive data types. While many of these field types follow common data typing conventions that are made explicit in their metadata, certain field types have unique characteristics that you need to understand before using them in your client application.

These field types apply to both standard and custom fields. They are enumerated in the `type` field of the Field type, which is described in the `fields` property of the DescribeSObjectResult.

Note: Some numeric fields have precision and scale limits. In addition, certain text fields have length restrictions. These restrictions are enforced when you `create()` or `update()` objects. However, the API may return data that does not meet these restrictions.

AnyType Field Type

The `anyType` field type is dynamic and returns `string`, `date`, `number`, or `boolean` data depending on the kind of field involved. For example, the element in a SOAP message has an `xsi:type="xsd:string"` attribute if the field is of type `string`. This field type is used in history objects for the `NewValue` and `OldValue` fields. It is also a valid field type for `fieldType` and `soapType`.

Note: Most SOAP toolkits automatically deserialize this element into the correct native type.

Calculated Field Type

Calculated fields are read-only fields in the API. These are fields defined by a formula, which is an algorithm that derives its value from other fields, expressions, or values. You can filter on these fields in SOQL, but you should not replicate these fields. The length of text calculated fields is 3900 characters or less—anything longer will be truncated.

Calculated fields are called formula fields in the Salesforce user interface.

ComboBox Field Type

A combobox is a picklist that also allows users to type a value that is not already specified in the list. A combobox is defined as a string value.

Currency Field Type

Currency fields contain currency values, such as the `ExpectedRevenue` field in a Campaign, and are defined as type double.

For organizations that have the multicurrency option enabled, the `CurrencyIsoCode` field is defined for any object that can have currency fields. The `CurrencyIsoCode` field and currency fields are linked in a special way. On any specific record, the `CurrencyIsoCode` field defines the currency of that record, and thus, the values of all currency fields on that record will be expressed in that currency.

For most cases, clients do not need to consider the linking of the `CurrencyIsoCode` field and the currency fields on an object. However, clients may need to consider the following:

- The `CurrencyIsoCode` field exists only for those organizations that have enabled multicurrency support.

- When displaying the currency values in a user interface, it is preferred to prepend each currency value with its `CurrencyIsoCode` value and a space separator.
- The `CurrencyIsoCode` field is a restricted picklist field. The set of allowable values, defined in the CurrencyType object, can vary from organization to organization. Attempting to set it to a value that is not defined for an organization causes the operation to be rejected.
- If you update the `CurrencyIsoCode` field on an object, it implicitly converts all currency values on that object to the new currency code, using the conversion rates that are defined for that organization in the Salesforce user interface. If you specify currency values in that same `update()` call, the new currency values you specify are interpreted in the new `CurrencyIsoCode` field value, without conversion.
- The picklist values in a `CurrencyIsoCode` field do not exactly match the labels displayed in Salesforce.

To perform currency conversions, client applications can look up the `CurrencyIsoCode` in the CurrencyType object.

DataCategoryGroupReference Field Type

A data category group has categories that classify articles in Salesforce Knowledge and questions in the Answers feature. Every article and question object has two fields of type DataCategoryGroupReference which contain the category group and category unique name. You can use the `describeDataCategoryGroups()` and `describeDataCategoryGroupStructures()` calls to retrieve the category groups and categories associated to these objects.

Email Field Type

Email fields contain email addresses. Client applications are responsible for specifying valid and properly formatted email addresses in `create()` and `update()` calls.

ID Field Type

With rare exceptions, all objects in the API have a field of type ID that is named `Id` and contains a unique identifier for each record in the object. It is analogous to a primary key in relational databases. When you `create()` a new record, the Web service generates an ID value for the record, ensuring that it is unique within your organization's data. You cannot use the `update()` call on ID fields. Because the ID value stays constant over the lifetime of the record, you can refer to the record by its ID value in subsequent API calls. Also, the ID value contains a three-character code that identifies the object type, which client applications can retrieve via the `describeSObjects()` call.

In addition, certain objects, including custom objects, have one or more fields of type `reference` that contain the ID value for a related record. These fields have names that end in the suffix "-Id", for example, `OwnerId` in the account object. `OwnerId` contains the ID of the user who owns that object. Unlike the field named `Id`, `reference` fields are analogous to foreign keys and can be changed via the `update()` call. For more information, see Reference Field Type.

Some API calls, such as `retrieve()` and `delete()`, accept an array of IDs as parameters—each array element uniquely identifies the row to retrieve or delete. Similarly, the `update()` call accepts an array of sObject records—each sObject contains an `Id` field that uniquely identifies the sObject.

ID fields in the Salesforce user interface contain 15-character, base-62, case-sensitive strings. Each of the 15 characters can be a numeric digit (0-9), a lowercase letter (a-z), or an uppercase letter (A-Z). Two unique IDs may only be different by a change in case.

Because there are applications like Access which do not recognize that 50130000000014c is a different ID from 50130000000014C, an 18-digit, case-safe version of the ID is returned by all API calls. The 18 character IDs have been formed by adding a suffix to each ID in the Force.com API. 18-character IDs can be safely compared for uniqueness by case-insensitive applications, and can be used in all API calls when creating, editing, or deleting data.

If you need to convert the 18-character ID to a 15-character version, truncate the last three characters. Salesforce.com recommends that you use the 18-character ID.

Note: Most Web services tools, including .NET and WSC, map the ID simple type defined in the API WSDL (Enterprise or Partner) to a string. However, other tools generate a specific ID class to represent the ID simple type. Please consult your web services toolkit documentation for more information.

Multi-Select Picklist Field Type

Multi-select picklist fields contain a list of one or more items from which a user can choose multiple items. One of the items can be configured as the default item. Selections are maintained as a string containing a series of attributes delimited by semicolons. For example, a query might return the values of a multivalue picklist as "first value; second value; third value". For information on querying multi-select picklists, see Querying Multi-Select Picklists in the *Salesforce SOQL and SOSL Reference Guide*.

Percent Field Type

Percent fields contain percent values. Percent fields are defined as type double.

Phone Field Type

Phone fields contain phone numbers, which can include alphabetic characters. Client applications are responsible for phone number formatting.

Picklist Field Type

Picklist fields contain a list of one or more items from which a user chooses a single item. They display as drop-down lists in the Salesforce user interface. One of the items can be configured as the default item.

In the Field object associated with the DescribeSObjectResult, the `restrictedPicklist` field defines whether the field is a restricted picklist or not. The API does not enforce the list of values for advisory (unrestricted) picklist fields on `create()` or `update()`. When inserting an unrestricted picklist field that does not have a PicklistEntry, the system creates an "inactive" picklist value. This value can be promoted to an "active" picklist value by adding the picklist value in the Salesforce user interface.

When creating new, inactive picklists, the API checks to see if there is a match. This check is case-insensitive.

In theField object associated with the DescribeSObjectResult, the `picklistValues` field contains an array of items (PicklistEntry objects). Each PicklistEntry defines the item's label, value, and whether it is the default item in the picklist (a picklist has no more than one default value).

Enumerated fields support localization of the labels to the language of the user. For example, for the `Industry` field on an Account, the value "Agriculture" may be translated to various languages. The enumerated field values are fixed and do not change with a user's language. However, each value may have a specified "label" field that provides the localized label for that value. You must always use the value when inserting or updating a field. The `query()` call always returns the value, not the label. The corresponding label for a value in the describeSObjectResult should be used when displaying the value to the user in any user interface.

The API supports the retrieval of the certain picklists in the following objects: CaseStatus, ContractStatus, LeadStatus, OpportunityStage, PartnerRole, SolutionStatus, TaskPriority, and TaskStatus. Each object represents a value in the respective picklist. These picklist entries always specify some other piece of information, such as whether the status is converted, and so on. Your client application can invoke the `query()` call on any of these objects (such as CaseStatus) to retrieve the set of values in the picklist, and then use that information while processing other objects (such as Case objects) to find more information about those objects (such as a given case). These objects are read-only via the API. To modify items in picklists, you must use the Salesforce user interface.

Reference Field Type

A reference field contains an `Id` value that points to a unique record (usually the parent record) on another object. This is analogous to the concept of a foreign key in relational databases. The name of a reference field ends, by convention, with the letters `Id` (such as `CaseId` or `OpportunityId`). For example, in the OpportunityCompetitor object, the `OpportunityId` field is a reference field that points to the Opportunity object. It contains an ID value that uniquely identifies an Opportunity record.

In some cases, an object can refer to another object of its same type. For example, an Account can have a parent link that points to another Account.

The Event and Task objects both have `WhoId` and `WhatId` cross-reference ID fields. Each of these cross-reference fields can point to one of several other objects. The `WhoId` field can point to a Contact or Lead, and the `WhatId` field can point to an Account, Opportunity, Campaign, or Case. In addition, if the `WhoId` field refers to a Lead, then the `WhatId` field must be empty.

You can describe and query each cross-referenced object. When you query a cross-reference ID field, it returns an object ID of the appropriate type. You can then query that ID to get additional information about the object, using the ID in the `id` field for that query.

The cross-reference ID field value is either:

- a valid record in your organization, or
- an empty value, which indicates an empty reference

The cross-reference ID field value, if non-`null`, is guaranteed to be an object in your organization. However, it is not guaranteed that you can query that object. Users with the "View All Data" permission can always query that object. Other users may be restricted from viewing or editing the referenced object.

When specifying a value for a cross-reference ID field in a `create()` or `update()` call, the value must be a valid value of type ID, and the user must have appropriate access to that object. The exact requirements vary from field to field.

Textarea Field Type

Textarea fields contain text that can be longer than 4000 bytes. Unlike string fields, textarea fields cannot be specified in the WHERE clause of a queryString of a `query()` call. To filter records on this field, you must do so while processing records in the QueryResult. For fields with this restriction, its `filterable` field in the Field type (described in the `fields` property of the DescribeSObjectResult) is `false`.

URL Field Type

URL fields contain URLs. Client applications are responsible for specifying valid and properly formatted URLs in `create()` and `update()` calls.

Compound Fields—Beta

Compound fields group together multiple elements of primitive data types, such as numbers or strings, to represent complex data types, such as a location or an address. Compound fields are an abstraction that can simplify application code that handles the values, leading to more concise, understandable code. Address and geolocation compound fields are available in the SOAP and REST APIs in API version 30.0 and later.

Note: This release contains a beta version of compound fields that is production quality but has known limitations. You can provide feedback and suggestions for compound fields, especially compound address fields, on the IdeaExchange.

Compound fields are accessible as a single, structured field, or as individual component fields. The values contained within the compound field and the values in individual fields both map to the same underlying data stored in Salesforce; they always have identical values. Code that references individual component fields is unaffected by the new compound fields.

Compound fields are read-only. Changes are performed by writing to the individual component fields. This maintains a single, consistent method for performing updates, and avoids the possibility of conflicts. For example, if both the `BillingAddress` compound field and `BillingCity` individual component field were updated in the same API call, it would be unclear which value should be saved.

Compound fields are available only through the SOAP and REST APIs. Compound fields are described in both the Enterprise and Partner WSDLs. Update your WSDL to at least API 30.0 to access the new compound data types.

Address Compound Fields

Standard addresses—addresses built into standard objects in Salesforce—are accessible in the SOAP and REST APIs as an `Address`, a structured compound data type, as well as individual address elements.

Using API 30.0 and later, standard addresses are available in the SOAP and REST APIs as a compound field of type `Address`, a structured data type that combines the following fields.

- `City`
- `Country`
- `CountryCode`
- `Latitude` (beta)
- `Longitude` (beta)
- `PostalCode`
- `State`
- `StateCode`
- `Street`

Note: `StateCode` and `CountryCode` are always available on compound address fields, whether or not state and country picklists are enabled in your organization.

The `Address` type extends the `Location` type, the data type used for compound geolocation fields.

Address fields are provided on many standard objects, such as Account, Contact, Quote, and User. Some objects provide fields for multiple addresses. For example, Account provides for four different addresses. In this case, address field names are prefixed with the type of address, for example, `BillingCity`, `BillingState`, and so on.

Note: Standard address compound fields are read-only, and are only accessible using the SOAP and REST APIs. See Compound Field Considerations and Limitations on page 36 for additional details of the restrictions this imposes.

Retrieving Compound Address Fields

Using compound fields can simplify code that works with addresses, especially for SOQL queries. SOQL `SELECT` clauses can reference addresses directly, instead of all of the individual component fields.

```
SELECT Name, BillingAddress
FROM Account
```

To write code that's compatible with API versions before 30.0, as well as API 30.0 and above, use the individual fields:

```
SELECT Name, BillingStreet, BillingCity, BillingState, BillingPostalCode,
       BillingCountry, BillingLatitude, BillingLongitude
FROM Account
```

Compound address field values are returned as a structured data type, `Address`. Code that works with compound address fields needs to reference the individual components of the returned value. See the code sample below.

Retrieve a Standard Address Compound Field with the SOAP API

The following Java method uses the Salesforce SOAP API to retrieve and display the Mailing Address for a list of contacts.

```
// Modified version of code in the SOAP API QuickStart
private void querySample() {
    String soqlQuery = "SELECT FirstName, LastName, MailingAddress FROM Contact";
    try {
        QueryResult qr = connection.query(soqlQuery);
        boolean done = false;

        if (qr.getSize() > 0) {
            System.out.println("\nLogged-in user can see "
              + qr.getRecords().length + " contact records.");

            while (!done) {
                System.out.println("");
                SObject[] records = qr.getRecords();
                for (int i = 0; i < records.length; ++i) {
                    Contact con = (Contact) records[i];
                    String fName = con.getFirstName();
                    String lName = con.getLastName();

                    // Access the compound address field MailingAddress
                    Address addr = (Address) con.getMailingAddress();
                    String streetAddr = "";
                    if (null != addr) streetAddr = addr.getStreet();

                    if (fName == null) {
                        System.out.println("Contact " + (i + 1) + ": " + lName +
                            " -- " + streetAddr);
                    } else {
                        System.out.println("Contact " + (i + 1) + ": " + fName +
                            " " + lName +
                            " -- " + streetAddr);
                    }
                }

                if (qr.isDone()) {
                    done = true;
                } else {
                    qr = connection.queryMore(qr.getQueryLocator());
                }
            }
        } else {
            System.out.println("No records found.");
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

Using Compound Address Fields as Locations

Compound address fields include a geolocation field, and can be used as locations in SOQL `WHERE` and `ORDER BY` clauses. For example, here's a SOQL query to retrieve the ten accounts closest to San Francisco.

```
SELECT Id, Name, BillingAddress
FROM Account
WHERE DISTANCE(BillingAddress, GEOLOCATION(37.775,-122.418), 'mi') < 20
ORDER BY DISTANCE(BillingAddress, GEOLOCATION(37.775,-122.418), 'mi')
LIMIT 10
```

Note: The geolocation component of address fields isn't set automatically. You need to use a geocoding service to set these values before you can use address fields as locatable values. Search the AppExchange for "geocode" to find a number of possible solutions.

Geolocation Compound Fields

Geolocation fields are accessible in the SOAP and REST APIs as a `Location`, a structured compound data type, as well as individual latitude and longitude elements.

Using API 30.0 and later, geolocation fields are available in the SOAP and REST APIs as a compound field of type `Location`, a structured data type that contains the following fields.

- latitude
- longitude

Geolocation fields are provided on many standard objects, such as Account, Contact, Quote, and User, as part of their address field or fields. Geolocation fields can also be added as custom fields to standard or custom objects.

Note:

- Geolocation compound fields are read-only, and are only accessible using the SOAP and REST APIs. See Compound Field Considerations and Limitations on page 36 for additional details of the restrictions this imposes.
- Although geolocation fields appear as a single field in the user interface, custom geolocation fields count as *three* custom fields towards your organization's limits: one for latitude, one for longitude, and one for internal use.

Retrieving Compound Geolocation Fields

Using compound fields can simplify code that works with geolocations, especially for SOQL queries. SOQL `SELECT` clauses can reference geolocations directly, instead of the individual component fields.

```
SELECT location__c
FROM Warehouse__c
```

To write code that's compatible with API versions before 30.0, as well as API 30.0 and above, use the individual latitude and longitude fields.

```
SELECT location__latitude__s, location__longitude__s
FROM Warehouse__c
```

Compound geolocation field values are returned as a structured data type, `Location`. Code that works with compound geolocation fields needs to reference the individual components of the returned value. See the sample code in Address Compound Fields on page 35.

Compound Field Considerations and Limitations

Address and geolocation compound fields are convenient, and result in more concise, clear code. However, there are a number of things you should consider when using them in your apps.

Both address and geolocation compound fields have the following limitations.

- Compound fields are read-only. Use the individual field components to update field values.
- Compound fields are accessible only through the SOAP and REST APIs. The compound versions of fields aren't accessible anywhere in the Salesforce user interface.

- Compound fields can’t be used in Apex, because there is no Apex data type for compound fields. Use the individual field components to access or update field values.
- Compound fields can’t be used in Visualforce, for example, in an `<apex:outputField>`. Use the individual field components to access or update field values.
- Compound fields aren’t searchable. For proximity searches, use a `WHERE DISTANCE(field, GEOLOCATION(lat, long), "units") < desiredPoximity` clause. For text string matches—for example, matching a city or postal code—search the individual fields.
- Compound fields can’t be imported or exported. Use the individual field components to load or export values.

Address compound fields have the following limitations.

- Compound address fields are only available for standard addresses, that is, those address fields that exist as a basic part of the standard objects included in Salesforce. You can’t create custom compound address fields.
- The geolocation component of address fields isn’t set automatically. You need to use a geocoding service to set these values before you can use address fields as locatable values. Search the AppExchange for “geocode” to find a number of possible solutions.

API Data Types and Salesforce Field Types

Generally, API data types and field types in the user interface have the same names. For example, a date field is represented by a date data type in the API. However, some field types are represented differently depending on whether you are inspecting an object via the API or the user interface. The following table contains the mapping for field types and data types that are different:

API Data Type	Corresponding Field Types in the User Interface
ID	Lookup relationship, master-detail relationship
string	Auto number, email, phone, picklist, multi-select picklist, text, text area, long text area, rich text area, data category group reference and URL. Different maximum lengths are specified in the WSDL for text, text area, and long text area.
boolean	Checkbox
double	Currency, formula, number, percent, and roll-up summary
Varies by type	When formula fields are created in the user interface, a type must be specified. This type corresponds to the API data type of the same name: currency, date, date/time, number, percent, or text.

All other fields that you can create in the user interface fall into one of the following categories:

- The field is not available in both the user interface and the API. For example, the BusinessHours object has fields of API data type time, but you cannot create a custom field of this type.
- Field types are the same as their corresponding API data type. For example, if you create a date field in the user interface, that field is the date data type in the API.

For more information about API data types, see Primitive Data Types and Field Types.

Core Data Types Used in API Calls

Many calls in the API use the following data types:

- sObject

- ID (String). See ID Field Type.

The API also uses several error handling objects. If an error occurs during a SOAP request, the API returns a SOAP fault message. The message contains different content, depending on the type of error:

- If an error affects the entire request, an API Fault Element, is returned, containing an ExceptionCode and the associated error message text.
- If the error affects some records and not others, an Error is returned, containing a StatusCode. These errors typically occur during bulk operations, such as creating, updating, or deleting multiple records with a single call.

sObject

An sObject represents an object, such as an Account or Campaign. For a list of standard objects, see Standard Objects.

An sObject has the following properties:

Name	Type	Description
`fieldsToNull`	string[]	Array of one or more field names whose value you want to explicitly set to `null`. When used with `update()` or `upsert()`, you can specify only those fields that you can update and that have the `nillable` property. When used with `create()`, you can specify only those fields that you can create and that have the `nillable` or the `default on create` property. For example, if specifying an ID field or required field results in a run-time error, you can specify that field name in `fieldsToNull`. Similarly, if you need to set a picklist value to none when creating a record, but the picklist has a default value, you can specify the field in `fieldsToNull`.
`ID`	ID	Unique ID for this individual object. For the `create()` call, this value is null. For all other API calls, this value must be specified.

API Fault Element

The following table lists the API fault elements that the API returns if an error occurs when processing a service request.

Fault	Description
`ApiQueryFault`	The row and column numbers where the problem occurred.
`LoginFault`	An error occurred during the `login()` call.
`InvalidSObjectFault`	An invalid sObject in a `describeSObject()`, `describeSObjects()`, `describeLayout()`, `describeDataCategoryGroups()`, `describeDataCategoryGroupStructures()`, `create()`, `update()`, `retrieve()`, or `query()` call.
`InvalidFieldFault`	An invalid field in a `retrieve()` or `query()` call.
`InvalidOrNullForRestrictedPicklist`	An invalid `appMenuType` in a `describeAppMenu()` call.
`MalformedQueryFault`	A problem in the queryString passed in a `query()` call.
`InvalidQueryLocatorFault`	A problem in the `queryLocator` passed in a `queryMore()` call.
`MalformedSearchFault`	A problem in the `search` passed in a `search()` call.
`InvalidIdFault`	A specified ID was invalid in a `setPassword()` or `resetPassword()` call.

Fault	Description
`UnexpectedErrorFault`	An unexpected error occurred. The error is not associated with any other API fault.

ExceptionCode

The following list of `ExceptionCode` values is defined in your WSDL file. Some codes may not be in your WSDL, depending on the features you have enabled.

`API_CURRENTLY_DISABLED`

Because of a system problem, API functionality is temporarily unavailable.

`API_DISABLED_FOR_ORG`

API access has not been enabled for the organization. Contact salesforce.com to enable API access.

`CANT_ADD_STANDARD_PORTAL_USER_TO_TERRITORY`

A user with a standard portal license can't be added to a territory.

`CIRCULAR_OBJECT_GRAPH`

The request failed because it contained a circular object reference.

`CLIENT_NOT_ACCESSIBLE_FOR_USER`

The current user does not have permission to access the specified client.

`CLIENT_REQUIRE_UPDATE_FOR_USER`

The current user is required to use a newer version of the specified client, and will have access until the client is updated.

`DELETE_REQUIRED_ON_CASCADE`

The delete operation triggers a cascade delete on a record, but the logged-in user does not have delete permission on that related object.

`DUPLICATE_COMM_NICKNAME`

You can't create a user with the same nickname as another user.

`DUPLICATE_VALUE`

You can't supply a duplicate value for a field that must be unique. For example, you may have submitted two copies of the same session ID in a `invalidateSessions()` call.

`EMAIL_BATCH_SIZE_LIMIT_EXCEEDED`

A method tried to process more email records than the maximum batch size.

`EMAIL_TO_CASE_INVALID_ROUTING`

An email to case record has been submitted for processing but the feature is not enabled.

`EMAIL_TO_CASE_LIMIT_EXCEEDED`

The daily converted email limit for the Email-to-Case feature has been exceeded.

`EMAIL_TO_CASE_NOT_ENABLED`

The Email-to-Case feature has not been enabled.

EXCEEDED_ID_LIMIT

Too many IDs were specified in a call. For example, more than 2000 IDs were requested in a `retrieve()` call, or more than 200 session IDs were specified in a `logout()` call.

EXCEEDED_LEAD_CONVERT_LIMIT

Too many IDs were sent to a `convertLead()` call.

EXCEEDED_MAX_SIZE_REQUEST

The size of the message sent to the API exceeded 50 MB.

EXCEEDED_MAX_TYPES_LIMIT

The number of object types to describe is too large.

EXCEEDED_QUOTA

The size limit for organization data storage was exceeded during a `create()` call.

FUNCTIONALITY_NOT_ENABLED

Functionality has been temporarily disabled. Other calls may continue to work.

INACTIVE_OWNER_OR_USER

The user or record owner is not active.

INACTIVE_PORTAL

The referenced portal is inactive.

INSUFFICIENT_ACCESS

The user does not have sufficient access to perform the operation.

INVALID_ASSIGNMENT_RULE

An invalid AssignmentRuleHeader value was specified.

INVALID_BATCH_SIZE

The query options have an invalid batch size value.

INVALID_CLIENT

The client is invalid.

INVALID_CROSS_REFERENCE_KEY

An invalid foreign key can't be set on a field. For example, if an object share such as AccountShare can't be deleted because the share row is a result of a sharing rule.

INVALID_FIELD

The specified field name is invalid.

INVALID_FILTER_LANGUAGE

The specified language can't be used as a filter.

INVALID_FILTER_VALUE

A SOQL query with `LIKE` specified an invalid character, for example, an incorrectly placed asterisk (*). Correct the query and resubmit.

INVALID_ID_FIELD

The specified ID is correctly formatted, but is not valid, for example, it is an ID of the wrong type, or the object it identifies no longer exists.

INVALID_GOOGLE_DOCS_URL

An invalid Salesforce record URL was used when trying to associate a Google Doc to that record. Correct the URL before trying the operation again.

INVALID_LOCATOR

The locator is invalid.

INVALID_LOGIN

The `login()` credentials are not valid, or the maximum number of logins have been exceeded. Contact your administrator for more information.

INVALID_NEW_PASSWORD

The new password does not conform with the password policies of the organization.

INVALID_OPERATION

The client application tried to modify a record that is locked by an approval process.

INVALID_OPERATION_WITH_EXPIRED_PASSWORD

Due to password expiration, a valid password must be set using `setPassword()` before the call can be invoked.

INVALID_QUERY_FILTER_OPERATOR

An invalid operator was used in the `query()` filter clause, at least for that field.

INVALID_QUERY_LOCATOR

An invalid `queryLocator` parameter was specified in a `queryMore()` call. It is also possible that you've exceed the maximum number of calls, which is ten per user for the API, and five for Apex and Visualforce.

INVALID_QUERY_SCOPE

The specified search scope is invalid.

INVALID_REPLICATION_DATE

The date for replication is out of the allowed range, such as before the organization was created.

INVALID_SETUP_OWNER

The setup owner must be an Organization, Profile, or User.

INVALID_SEARCH

The `search()` call has invalid syntax or grammar. For more information, see the *Salesforce SOQL and SOSL Reference Guide*.

INVALID_SEARCH_SCOPE

The specified search scope is invalid.

INVALID_SESSION_ID

The specified `sessionId` is malformed (incorrect length or format) or has expired. Log in again to start a new session.

INVALID_SOAP_HEADER

There is an error in the SOAP header. If you are migrating from an earlier version of the API, be advised that the SaveOptions header can't be used with API version 6.0 or later. Use AssignmentRuleHeader instead.

INVALID_SSO_GATEWAY_URL

The URL provided to configure the Single Sign-On gateway was not a valid URL.

INVALID_TYPE

The specified sObject type is invalid.

INVALID_TYPE_FOR_OPERATION

The specified sObject type is invalid for the specified operation.

LIMIT_EXCEEDED

An array is too long. For example, there are too many BCC addresses, targets, or email messages.

LOGIN_DURING_RESTRICTED_DOMAIN

The user is not allowed to log in from this IP address.

LOGIN_DURING_RESTRICTED_TIME

The user is not allowed to log in during this time period.

MALFORMED_ID

An invalid ID string was specified. For information about IDs, see ID Field Type.

MALFORMED_QUERY

An invalid query string was specified. For example, the query string was longer than 20,000 characters.

MALFORMED_SEARCH

An invalid search string was specified. For example, the search string was longer than 20,000 characters.

MISSING_ARGUMENT

A required argument is missing.

MIXED_DML_OPERATION

There are limits on what kinds of DML operations can be performed in the same transaction. For more information, see the *Force.com Apex Code Developer's Guide*.

NOT_MODIFIED

The describe call response has not changed since the specified date.

NO_SOFTPHONE_LAYOUT

If an organization has the CTI feature enabled, but no softphone layout has been defined, this exception is returned if a describe call is issued. This is most often caused because no call center has been defined: during call center definition, a default softphone layout is created.

If an organization does not have the CTI feature enabled, `FUNCTIONALITY_NOT_ENABLED` is returned instead.

NUMBER_OUTSIDE_VALID_RANGE

The number specified is outside the valid range for the field.

OPERATION_TOO_LARGE

The query has returned too many results. If certain queries are run by a user without the “View All Data” permission and many records are returned, those queries would require sharing rule checking. An example of this would be queries that are run on objects, such as Task, that use a polymorphic foreign key. Such queries return this exception because the operation requires too many resources. To correct, add filters to the query to narrow the scope, or use filters such as date ranges to break the query up into a series of smaller queries.

ORG_LOCKED

The organization has been locked. You must contact salesforce.com to unlock the organization.

ORG_NOT_OWNED_BY_INSTANCE

The user tried to log in to the wrong server instance. Choose another server instance or log in at `https://login.salesforce.com`. You can use `http` instead of `https`.

PASSWORD_LOCKOUT

The user has exceeded the allowed number of login attempts. The user must contact his or her administrator to regain login access.

PORTAL_NO_ACCESS

Access to the specified portal is not available.

QUERY_TIMEOUT

The query has timed out. For more information, see the *Salesforce SOQL and SOSL Reference Guide*.

QUERY_TOO_COMPLICATED

SOQL query is either selecting too many fields or there are too many filter conditions. Try reducing the number of formula fields referenced in the query.

REQUEST_LIMIT_EXCEEDED

Exceeded either the concurrent request limit or the request rate limit for your organization. For details on API request limits, see API Usage Metering.

REQUEST_RUNNING_TOO_LONG

A request has taken too long to be processed.

SERVER_UNAVAILABLE

A server that is necessary for this call is currently unavailable. Other types of requests might still work.

SSO_SERVICE_DOWN

The service was unavailable, and an authentication call to the organization’s specified Single Sign-On server failed.

TOO_MANY_APEX_REQUESTS

Too many Apex requests have been issued. If this persists, contact Salesforce Customer Support.

TRIAL_EXPIRED

The trial period for the organization has expired. A representative from the organization must contact salesforce.com to re-enable the organization.

UNSUPPORTED_API_VERSION

A method call was made that does not exist in the accessed API version, for example, trying to use `upsert()` (new in 8.0) against version 5.0.

UNSUPPORTED_CLIENT

This version of the client is no longer supported.

Error

An `Error` contains information about an error that occurred during a `create()`, `merge()`, `process()`, `update()`, `upsert()`, `delete()`, or `undelete()` call. For more information, see Error Handling. An `Error` has the following properties:

Name	Type	Description
`statusCode`	StatusCode	A code that characterizes the error. The full list of status codes is available in the WSDL file for your organization (see Generating the WSDL File for Your Organization).
`message`	string	Error message text.
`fields`	string[]	Array of one or more field names. Identifies which fields in the object, if any, affected the error condition.

StatusCode

The following table lists API status codes that are returned with an error. Some codes may not be in your WSDL, depending on the features you have enabled.

ALL_OR_NONE_OPERATION_ROLLED_BACK

The bulk operation was rolled back because one of the records wasn't processed successfully. See AllOrNoneHeader.

ALREADY_IN_PROCESS

You can't submit a record that is already in an approval process. You must wait for the previous approval process to complete before resubmitting a request with this record.

ASSIGNEE_TYPE_REQUIRED

You must designate an assignee for the approval request (ProcessInstanceStep or ProcessInstanceWorkitem).

BAD_CUSTOM_ENTITY_PARENT_DOMAIN

The changes you are trying to make can't be completed because changes to the associated master-detail relationships can't be made.

BCC_NOT_ALLOWED_IF_BCC_COMPLIANCE_ENABLED

Your client applicationblind carbon-copied an email address on an email even though the organization's Compliance BCC Email option is enabled. This option specifies a particular email address that automatically receives a copy of all outgoing email. When this option is enabled, you can't BCC any other email address. To disable the option, log into the user interface and from Setup, click **Security Controls** > **Compliance BCC Email**.

BCC_SELF_NOT_ALLOWED_IF_BCC_COMPLIANCE_ENABLED

Your client application blind carbon copied the logged-in user's email address on an email even though the organization's BCC COMPLIANCE option is set to true. This option specifies a particular email address that automatically receives a copy of all outgoing email. When this option is enabled, you can't BCC any other email address. To disable the option, log into the user interface and from Setup, click **Security Controls** > **Compliance BCC Email**.

`CANNOT_CASCADE_PRODUCT_ACTIVE`

An update to a product caused by a cascade can't be done because the associated product is active.

`CANNOT_CHANGE_FIELD_TYPE_OF_APEX_REFERENCED_FIELD`

You can't change the type of a field that is referenced in an Apex script.

`CANNOT_CREATE_ANOTHER_MANAGED_PACKAGE`

You can only create one managed package in an organization.

`CANNOT_DEACTIVATE_DIVISION`

You can't deactivate Divisions if an assignment rule references divisions or if the `DefaultDivision` field on a user record isn't set to null.

`CANNOT_DELETE_LAST_DATED_CONVERSION_RATE`

You must have at least one DatedConversionRate record if dated conversions are enabled.

`CANNOT_DELETE_MANAGED_OBJECT`

You can't modify components that are included in a managed package.

`CANNOT_DISABLE_LAST_ADMIN`

You must have at least one active administrator user.

`CANNOT_ENABLE_IP_RESTRICT_REQUESTS`

If you exceed the limit of five IP ranges specified in a profile, you can't enable restriction of login by IP addresses. Reduce the number of specified ranges in the profile and try the request again.

`CANNOT_INSERT_UPDATE_ACTIVATE_ENTITY`

You do not have permission to create, update, or activate the specified record.

`CANNOT_MODIFY_MANAGED_OBJECT`

You can't modify components that are included in a managed package.

`CANNOT_RENAME_APEX_REFERENCED_FIELD`

You can't rename a field that is referenced in an Apex script.

`CANNOT_RENAME_APEX_REFERENCED_OBJECT`

You can't rename an object that is referenced in an Apex script.

`CANNOT_REPARENT_RECORD`

You can't define a new parent record for the specified record.

`CANNOT_RESOLVE_NAME`

A `sendEmail()` call could not resolve an object name.

`CANNOT_UPDATE_CONVERTED_LEAD`

A converted lead could not be updated.

`CANT_DISABLE_CORP_CURRENCY`

You can't disable the corporate currency for an organization. To disable a currency that is set as the corporate currency, first use the user interface to change the corporate currency to a different currency, and then disable the original currency.

CANT_UNSET_CORP_CURRENCY

You can't change the corporate currency for an organization from the API. Use the user interface to change the corporate currency.

CHILD_SHARE_FAILS_PARENT

You can't change the owner of or define sharing rules for a record that is a child of another record if you do not also have the appropriate permissions on the parent. For example, you can't change the owner of a contact record if you can't edit its parent account record.

CIRCULAR_DEPENDENCY

You can't create a circular dependency between metadata objects in your organization. For example, public group A can't include public group B, if public group B already includes public group A.

COMMUNITY_NOT_ACCESSIBLE

You do not have permission to access the community that this entity belongs to. You must be given permission to access the community before you can access this entity.

CUSTOM_CLOB_FIELD_LIMIT_EXCEEDED

You can't exceed the maximum size for a CLOB field.

CUSTOM_ENTITY_OR_FIELD_LIMIT

You have reached the maximum number of custom objects or custom fields for your organization.

CUSTOM_FIELD_INDEX_LIMIT_EXCEEDED

You have reached the maximum number of indexes on a field for your organization.

CUSTOM_INDEX_EXISTS

You can create only one custom index per field.

CUSTOM_LINK_LIMIT_EXCEEDED

You have reached the maximum number of custom links for your organization.

CUSTOM_TAB_LIMIT_EXCEEDED

You have reached the maximum number of custom tabs for your organization.

DELETE_FAILED

You can't delete a record because it is in use by another object.

DEPENDENCY_EXISTS

You can't perform the requested operation because of an existing dependency on the specified object or field.

DUPLICATE_CASE_SOLUTION

You can't create a relationship between the specified case and solution because it already exists.

DUPLICATE_CUSTOM_ENTITY_DEFINITION

Custom object or custom field IDs must be unique.

DUPLICATE_CUSTOM_TAB_MOTIF

You can't create a custom object or custom field with a duplicate master name.

DUPLICATE_DEVELOPER_NAME

You can't create a custom object or custom field with a duplicate developer name.

DUPLICATE_EXTERNAL_ID

A user-specified external ID matches more than one record during an upsert.

DUPLICATE_MASTER_LABEL

You can't create a custom object or custom field with a duplicate master name.

DUPLICATE_SENDER_DISPLAY_NAME

A `sendEmail()` call could not choose between OrgWideEmailAddress.`DisplayName` or `senderDisplayName`. Define only one of the two fields.

DUPLICATE_USERNAME

A create, update, or upsert failed because of a duplicate user name.

DUPLICATE_VALUE

You can't supply a duplicate value for a field that must be unique. For example, you may have submitted two copies of the same session ID in a `invalidateSessions()` call.

EMAIL_NOT_PROCESSED_DUE_TO_PRIOR_ERROR

Because of an error earlier in the call, the current email was not processed.

EMPTY_SCONTROL_FILE_NAME

The Scontrol file name was empty, but the binary was not empty.

ENTITY_FAILED_IFLASTMODIFIED_ON_UPDATE

You can't update a record if the date in`LastModifiedDate` is later than the current date.

ENTITY_IS_ARCHIVED

You can't access a record if it has been archived.

ENTITY_IS_DELETED

You can't reference an object that has been deleted. Note that this status code only occurs in version 10.0 of the API and later. Previous releases of the API use `INVALID_ID_FIELD` for this error.

ENTITY_IS_LOCKED

You can't edit a record that is locked by an approval process.

ENVIRONMENT_HUB_MEMBERSHIP_CONFLICT

You can’t add an organization to more than one Environment Hub.

ERROR_IN_MAILER

An email address is invalid, or another error occurred during an email-related transaction.

FAILED_ACTIVATION

The activation of a Contract failed.

FIELD_CUSTOM_VALIDATION_EXCEPTION

You can't define a custom validation formula that violates a field integrity rule.

FIELD_FILTER_VALIDATION_EXCEPTION

You can't violate field integrity rules.

FILTERED_LOOKUP_LIMIT_EXCEEDED

The creation of the lookup filter failed because it exceeds the maximum number of lookup filters allowed per object.

HTML_FILE_UPLOAD_NOT_ALLOWED

Your attempt to upload an HTML file failed. HTML attachments and documents, including HTML attachments to a Solution, can't be uploaded if the `Disallow HTML documents and attachments` checkbox is selected under Setup, in **Security Controls** > **HTML Documents and Attachments Settings**.

IMAGE_TOO_LARGE

The image exceeds the maximum width, height, and file size.

INACTIVE_OWNER_OR_USER

The owner of the specified item is an inactive user. To reference this item, either reactivate the owner or reassign ownership to another active user.

INSUFFICIENT_ACCESS_ON_CROSS_REFERENCE_ENTITY

An operation affects an object that is cross-referenced by the specified object, but the logged-in user does not have sufficient permissions on the cross-referenced object. For example, if the logged-in user attempts to modify an account record, that user might not have permission to approve, reject, or reassign a ProcessInstanceWorkitem that is submitted after that action.

INSUFFICIENT_ACCESS_OR_READONLY

You can't perform the specified action because you do not have sufficient permissions.

INVALID_ACCESS_LEVEL

You can't define a new sharing rule if it provides less access than the specified organization-wide default.

INVALID_ARGUMENT_TYPE

You supplied an argument that is of the wrong type for the operation being attempted.

INVALID_ASSIGNEE_TYPE

You specified an assignee type that is not a valid integer between one and six.

INVALID_ASSIGNMENT_RULE

You specified an assignment rule that is invalid or that is not defined in the organization.

INVALID_BATCH_OPERATION

The specified batch operation is invalid.

INVALID_CONTENT_TYPE

The outgoing email has anEmailFileAttachment with an invalid`contentType` property. See RFC2045 - Internet Message Format.

INVALID_CREDIT_CARD_INFO

The specified credit card information is not valid.

INVALID_CROSS_REFERENCE_KEY

The specified value in a relationship field is not valid, or data is not of the expected type.

`INVALID_CROSS_REFERENCE_TYPE_FOR_FIELD`
The specified cross reference type is not valid for the specified field.

`INVALID_CURRENCY_CONV_RATE`
You must specify a positive, non-zero value for the currency conversion rate.

`INVALID_CURRENCY_CORP_RATE`
You can't modify the corporate currency conversion rate.

`INVALID_CURRENCY_ISO`
The specified currency ISO code is not valid..

`INVALID_EMAIL_ADDRESS`
A specified email address is invalid.

`INVALID_EMPTY_KEY_OWNER`
You can't set the value for `owner` to null.

`INVALID_FIELD`
You specified an invalid field name when trying to update or upsert a record.

`INVALID_FIELD_FOR_INSERT_UPDATE`
You can't combine a person account record type change with any other field update.

`INVALID_FIELD_WHEN_USING_TEMPLATE`
You can't use an email template with an invalid field name.

`INVALID_FILTER_ACTION`
The specified filter action can't be used with the specified object. For example, an alert is not a valid filter action for a Task.

`INVALID_ID_FIELD`
The specified ID field (`ID`, `ownerId`), or cross-reference field is invalid.

`INVALID_INET_ADDRESS`
A specified Inet address is not valid.

`INVALID_LINEITEM_CLONE_STATE`
You can't clone a Pricebook2 or PricebookEntry record if those objects are not active.

`INVALID_MASTER_OR_TRANSLATED_SOLUTION`
The solution is invalid. For example, this error can occur if you try to associate a translated solution with a master solution when another translated solution in the same language is already associated with the master solution.

`INVALID_MESSAGE_ID_REFERENCE`
The outgoing email's References or In-Reply-To fields are invalid. These fields must contain valid Message-IDs. See RFC2822 - Internet Message Format.

`INVALID_OPERATION`
There is no applicable approval process for the specified object.

INVALID_OPERATOR

The specified operator is not applicable for the field type when used as a workflow filter.

INVALID_OR_NULL_FOR_RESTRICTED_PICKLIST

You specified an invalid or null value for a restricted picklist.

INVALID_PARTNER_NETWORK_STATUS

The specified partner network status is invalid for the specified template field.

INVALID_PERSON_ACCOUNT_OPERATION

You can't delete a person account.

INVALID_READ_ONLY_USER_DML

You can't create, update, or delete data while the instance where your organization resides is being upgraded to the latest release. Please try again after the release has completed. For release schedules, see `trust.salesforce.com`.

INVALID_SAVE_AS_ACTIVITY_FLAG

You must specify `true` or `false` for the `saveAsActivity` flag.

INVALID_SESSION_ID

The specified `sessionId` is malformed (incorrect length or format) or has expired. Log in again to start a new session.

INVALID_STATUS

The specified organization status change is not valid.

INVALID_TYPE

The specified type is not valid for the specified object.

INVALID_TYPE_FOR_OPERATION

The specified type is not valid for the specified operation.

INVALID_TYPE_ON_FIELD_IN_RECORD

The specified value is not valid for the specified field's type.

IP_RANGE_LIMIT_EXCEEDED

The specified IP address is outside the IP range specified for the organization.

JIGSAW_IMPORT_LIMIT_EXCEEDED

The number of records you attempted to purchase from Data.com exceeds your available record addition limit.

LICENSE_LIMIT_EXCEEDED

You have exceeded the number of licenses assigned to your organization.

LIGHT_PORTAL_USER_EXCEPTION

You attempted an action with a high-volume portal user that's not allowed. For example, trying to add the user to a case team.

LIMIT_EXCEEDED

You have exceeded a limit. The limit may be on a field size or value, license, or other component.

LOGIN_CHALLENGE_ISSUED

An email containing a security token was sent to the user's email address because he or she logged in from an IP address that is not included in their organization's list of trusted IP addresses. The user can't log in until he or she adds the security token to the end of his or her password.

LOGIN_CHALLENGE_PENDING

The user logged in from an IP address that is not included in their organization's list of trusted IP addresses, but a security token has not yet been issued.

LOGIN_MUST_USE_SECURITY_TOKEN

The user must add a security token to the end of his or her password to log in.

MALFORMED_ID

An ID must be either 15 characters, or 18 characters with a valid case-insensitive extension. There is also an exception code of the same name.

MANAGER_NOT_DEFINED

A manager has not been defined for the specified approval process.

MASSMAIL_RETRY_LIMIT_EXCEEDED

A mass mail retry failed because your organization has exceeded its mass mail retry limit.

MASS_MAIL_LIMIT_EXCEEDED

The organization has exceeded its daily limit for mass email. Mass email messages can't be sent again until the next day.

MAXIMUM_CCEMAILS_EXCEEDED

You have exceeded the maximum number of specified CC addresses in a workflow email alert.

MAXIMUM_DASHBOARD_COMPONENTS_EXCEEDED

You have exceeded the document size limit for a dashboard.

MAXIMUM_HIERARCHY_LEVELS_REACHED

You have reached the maximum number of levels in a hierarchy.

MAXIMUM_SIZE_OF_ATTACHMENT

You have exceeded the maximum size of an attachment.

MAXIMUM_SIZE_OF_DOCUMENT

You have exceeded the maximum size of a document.

MAX_ACTIONS_PER_RULE_EXCEEDED

You have exceeded the maximum number of actions per rule.

MAX_ACTIVE_RULES_EXCEEDED

You have exceeded the maximum number of active rules.

MAX_APPROVAL_STEPS_EXCEEDED

You have exceeded the maximum number of approval steps for an approval process.

MAX_FORMULAS_PER_RULE_EXCEEDED

You have exceeded the maximum number of formulas per rule.

MAX_RULES_EXCEEDED

You have exceeded the maximum number of rules for an object.

MAX_RULE_ENTRIES_EXCEEDED

You have exceeded the maximum number of entries for a rule.

MAX_TASK_DESCRIPTION_EXCEEDED

The task description is too long.

MAX_TM_RULES_EXCEEDED

You have exceeded the maximum number of rules per Territory.

MAX_TM_RULE_ITEMS_EXCEEDED

You have exceeded the maximum number of rule criteria per rule for a Territory.

MERGE_FAILED

A merge operation failed.

MISSING_ARGUMENT

You did not specify a required argument.

NONUNIQUE_SHIPPING_ADDRESS

You can't insert a reduction order item if the original order shipping address is different from the shipping address of other items in the reduction order.

NO_APPLICABLE_PROCESS

A `process()` request failed because the record submitted does not satisfy the entry criteria of any active approval processes for which the user has permission.

NO_ATTACHMENT_PERMISSION

Your organization does not permit email attachments.

NO_INACTIVE_DIVISION_MEMBERS

You can't add members to an inactive Division.

NO_MASS_MAIL_PERMISSION

You do not have permission to send the specified email. You must have “Mass Email” if you are sending mass mail or “Send Email” if you are sending individual email.

NUMBER_OUTSIDE_VALID_RANGE

The number specified is outside the valid range of values.

NUM_HISTORY_FIELDS_BY_SOBJECT_EXCEEDED

The number of history fields specified for the sObject exceeds the allowed limit.

OP_WITH_INVALID_USER_TYPE_EXCEPTION

The operation you attempted can't be performed for one or more users. For example, you can't add high-volume portal users to a group.

OPTED_OUT_OF_MASS_MAIL

An email can't be sent because the specified User has opted out of mass mail.

PACKAGE_LICENSE_REQUIRED

The logged-in user can't access an object that is in a licensed package if the logged-in user does not have a license for the package.

PORTAL_USER_ALREADY_EXISTS_FOR_CONTACT

A create User operation failed because you can't create a second portal user under a Contact.

PRIVATE_CONTACT_ON_ASSET

You can't have a private contact on an asset.

RECORD_IN_USE_BY_WORKFLOW

You can't access a record if it is currently in use by a workflow or approval process.

REQUEST_RUNNING_TOO_LONG

A request that has been running too long may be canceled.

REQUIRED_FIELD_MISSING

A call requires a field that was not specified.

SELF_REFERENCE_FROM_TRIGGER

You can't recursively update or delete the same object from an Apex trigger. This error often occurs when:

- You try to update or delete an object from within its before trigger.
- You try to delete an object from within its after trigger.

This error occurs with both direct and indirect operations. The following is an example of an indirect operation:

1. A request is submitted to update Object A.
2. A `before update` trigger on object A creates an object B.
3. Object A is updated.
4. An `after insert` trigger on object B queries object A and updates it. This is an indirect update of object A because of the before trigger of object A, so an error is generated.

SHARE_NEEDED_FOR_CHILD_OWNER

You can't delete a sharing rule for a parent record if its child record needs it.

SINGLE_EMAIL_LIMIT_EXCEEDED

(API version 18.0 and higher) The organization has exceeded its daily limit for single emails. Single email messages can't be sent again until the next day.

STANDARD_PRICE_NOT_DEFINED

Custom prices can't be defined without corresponding standard prices.

STORAGE_LIMIT_EXCEEDED

You have exceeded your organization's storage limit.

STRING_TOO_LONG

The specified string exceeds the maximum allowed length.

TABSET_LIMIT_EXCEEDED

You have exceeded the number of tabs allowed for a tabset.

`TEMPLATE_NOT_ACTIVE`

The template specified is unavailable. Specify another template or make the template available for use.

`TERRITORY_REALIGN_IN_PROGRESS`

An operation can't be performed because a territory realignment is in progress.

`TEXT_DATA_OUTSIDE_SUPPORTED_CHARSET`

The specified text uses a character set that is not supported.

`TOO_MANY_APEX_REQUESTS`

Too many Apex requests have been sent. This error is transient. Resend your request after a short wait.

`TOO_MANY_ENUM_VALUE`

A request failed because too many values were passed in for a multi-select picklist. You can select a maximum of 100 values for a multi-select picklist.

`TRANSFER_REQUIRES_READ`

You can't assign the record to the specified User because the user does not have read permission.

`UNABLE_TO_LOCK_ROW`

A deadlock or timeout condition has been detected:

- Deadlocks involve at least two transactions that are attempting to update overlapping sets of objects. Note that if the transaction involves a summary field, the parent objects are locked, making these transactions especially prone to deadlocks. To debug, check your code for deadlocks and correct. Deadlocks are usually not the result of an issue with Salesforce operations.
- Timeouts occur when a transaction takes too long to complete, for example, when replacing a value in a picklist, or changing a custom field definition. These are temporary states. There is no corrective action needed.

If an object in a batch can't be locked, the entire batch fails with this error.

`UNAVAILABLE_RECORDTYPE_EXCEPTION`

The appropriate default record type could not be found.

`UNDELETE_FAILED`

An object could not be undeleted because it does not exist or has not been deleted.

`UNKNOWN_EXCEPTION`

The system encountered an internal error. Please report this problem to salesforce.com.

Note: Do not report this exception code to salesforce.com if it results from a `sendEmail()` call. The `sendEmail()` call returns this exception code when it is used to send an email to one or more recipients who have the **Email Opt Out** option selected.

`UNSPECIFIED_EMAIL_ADDRESS`

The specified user does not have an email address.

`UNSUPPORTED_APEX_TRIGGER_OPERATION`

You can't save recurring events with an Apex trigger.

UNVERIFIED_SENDER_ADDRESS

A `sendEmail()` call attempted to use an unverified email address defined in the OrgWideEmailAddress object.

WEBLINK_SIZE_LIMIT_EXCEEDED

The size of a WebLink URL or JavaScript code exceeds the limit.

WEBLINK_URL_INVALID

The WebLink URL has failed the URL string validation check.

WRONG_CONTROLLER_TYPE

The controller type for your Visualforce email template does not match the object type being used.

If you receive a status code not listed in the previous table, contact Customer Support.

System Fields

The following fields are read-only fields found on most objects. These fields are automatically updated during API operations. For example, the ID field is automatically generated during a create operation and the LastModifiedDate is automatically updated during any operation on an object.

Field	Field Type	Description
`Id`	ID	Globally unique string that identifies a record. For information on IDs, see ID Field Type. Because this field exists in every object, it is not listed in the field table for each object. `Id` fields have Defaulted on create and Filter access.
`IsDeleted`	boolean	Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). Because this field does not appear in all objects, it is listed in the field table for each object.
		Audit Fields
`CreatedById`	reference	ID of the User who created this record. `CreatedById` fields have Defaulted on create and Filter access.
`CreatedDate`	dateTime	Date and time when this record was created. `CreatedDate` fields have Defaulted on create and Filter access.
`LastModifiedById`	reference	ID of the User who last updated this record. `LastModifiedById` fields have Defaulted on create and Filter access.
`LastModifiedDate`	dateTime	Date and time when this record was last modified by a user. `LastModifiedDate` fields have Defaulted on create and Filter access.
`SystemModstamp`	dateTime	Date and time when this record was last modified by a user or by an automated process (such as a trigger). `SystemModstamp` fields have Defaulted on create and Filter access.

If you import data into Salesforce and need to set the value for an audit field, contact salesforce.com. Once salesforce.com enables this capability for your organization, you can set audit field values for the following objects: Account, CampaignMember, Case, CaseComment, Contact, FeedComment, FeedItem, Idea, IdeaComment, Lead, Opportunity, and Vote. The only audit field you cannot set a value for is `systemModstamp`.

For information on setting audit fields for custom objects, see Audit Fields for Custom Objects.

Not all standard objects have all audit fields. Check the Enterprise WSDL to verify which audit fields are available for a given object.

Parent Reference Fields

If an object has a relationship to a parent object, two fields are added:

- `Parent_Name` contains the object name of the parent. For example, Case has a `Contact` field that contains a reference to the contact parent of the case.
- `Parent_NameId` contains the ID of the parent. For example, Case has a `ContactId` field that refers to the contact parent of the case. This field is used in SOQL relationship queries such as the following:

```
SELECT Case.ContactId, Case.Contact.Name FROM Case
```

Even if the object can parent itself, these fields occur. For example, the Campaign object has a `Campaign` and `CampaignId` field for referencing the parent Campaign.

Required Fields

Required fields must have a non-`null` value. This rule affects the create and update calls:

- In a create call, the system automatically populates the data for certain required fields (such as system fields and the object ID fields). Similarly, if a required field has a default value (its `defaultedOnCreate attribute` is set to `true`, as described in Field in a describe result, then the system implicitly assigns a value for this field when the object is created, even if a value for this field is not explicitly passed in on the create call. For all other required fields, such as ID fields that are analogous to foreign keys in SQL, a client application must explicitly assign a value when the object is created (it cannot be `null`).
- In updates, a required field cannot be set to `null`, and many required fields can't be changed.

Any field not specified as required in the object description is optional, that is, it can be `null` when updated or created.

Some required fields for some objects require special handling.

Frequently-Occurring Fields

In addition to system fields, the following fields are found on many objects:

- `OwnerId`
- `RecordTypeId`
- `CurrencyIsoCode`

`OwnerId`

Objects have an `ownerId` field that is an reference to the user who owns that object. Ownership is an important concept that affects the security model and has other implications throughout the system. Any user can query the owner field for any record they can access. However, setting the `ownerId` field has the following limitations:

- For most users and most objects, this field can't be set directly upon insert. It is implicitly set to the current user when inserting an object.
- When creating or updating a Case or Lead, a client application (that is logged in with sufficient permissions to transfer a record) can set this field to any valid User in the organization or to any valid queue of the appropriate type in the organization.

- Updating this field via the API changes only the owner of that record. The change of ownership does not cascade to associated records as it does when you transfer record ownership in the Salesforce user interface.
- Updating this field on an account deletes the existing sharing information and reapplies the organization-wide sharing defaults and sharing rules.
- To update the `ownerId` field, the user must have the "Transfer Record" permission and Read access to the new owner.

In API version 12.0 and later, if your organization has set up opportunity teams, `OwnerId` fields behave the same for Account and Opportunity objects as for other objects. That is, if you update the `OwnerId` field in either object, any AccountShare or OpportunityShare records with `RowCause` set to Sales Team are kept. In API version 11.0 and earlier, the sharing records are deleted.

RecordTypeId

Record types are used to offer different business processes and subsets of picklist values to different User records based on their Profile settings. (In addition, person accounts use record types to manage a number of additional elements.

Record types are configured in the user interface or by creating, editing, or deleting the RecordType object in the API. Retrieve the list of valid record type IDs (String) for an object by querying the RecordType object.

The `RecordTypeId` field in an object contains the ID of the RecordType record that is associated with a standard or custom object. You can create or update this field.

Note: You can't create or update the `RecordTypeId` field on the CampaignMember records. Set the CampaignMember record type using the `CampaignMemberRecordTypeId` field on Campaign.

When specified in a create or update call, the record type ID (String) must refer to a valid record type for that object.

Note: The `RecordTypeId` field is in your organization's WSDL only if at least one record type is configured for your organization in the Salesforce user interface.

CurrencyIsoCode

For organizations that have multicurrency enabled, the `CurrencyIsoCode` field contains the string representation of the currency ISO code associated with currency values in the object. Note that the User object also has a `DefaultCurrencyIsoCode` field, which is the default currency for that user. For example, a user in France could have a `DefaultCurrencyIsoCode` set to Euros, and that would be their default currency in the application. However, the User object could have currency custom fields stored in a different currency, that will correspond to the organization currency at the time the user record is created.

API Field Properties

Fields on objects represent the details of each object and are analogous to columns in a database table. Each field on each object has one or more of the following properties:

Property	Description
Autonumber	The API creates an autonumber.
Create	Value for the field can be specified during create using the API.
Defaulted on create	When created, a default value is supplied if no other value is specified.
Delete	Value for the field can be deleted using the API.
Filter	Can be used as filter criteria in a SOQL query FROM or WHERE clause.

Property	Description
Group	Can be included in the GROUP BY clause of a SOQL query (`true`) or not (`false`). Available in API version 18.0 and later.
idLookup	Can be used to specify a record in an upsert call. The `Id` field of each object has this property and some `Name` fields. There are exceptions, so check for the property in any object you wish to upsert.
Nillable	The field can contain a null value.
Query	The field can be queried with SOQL using the API.
Replicate	The value of the field can be replicated using the API.
Restricted picklist	A picklist that depends on the value of another picklist for the values it displays.
Retrieve	Value of the field can be retrieved using the API.
Search	Can be searched with SOSL using the API.
Sort	Indicates whether a query can sort on this field (`true`) or not (`false`).
Update	Can be updated using the API.

Relationships Among Objects

Relationships associate objects with other objects. For example, a relationship can link a custom object to standard object in a related list, such as linking a custom object called Bugs to cases to track product defects associated with customer cases. To view the parent and child relationships among standard objects, see the ERD diagrams in Data Model.

Note: You can use parent-child relationships in SOQL queries. For more information, see Relationship Queries in the *Salesforce SOQL and SOSL Reference Guide*.

You can define different types of relationships by creating custom relationship fields on an object. The differences between relationship types include how they handle data deletion, record ownership, security, and required fields in page layouts:

- **Master-Detail (1:n)** — A parent-child relationship in which the master object controls certain behaviors of the detail object:
  - ◊ When a record of the master object is deleted, its related detail records are also deleted.
  - ◊ The `Owner` field on the detail object is not available and is automatically set to the owner of its associated master record. Custom objects on the detail side of a master-detail relationship cannot have sharing rules, manual sharing, or queues, as these require the `Owner` field.
  - ◊ The detail record inherits the sharing and security settings of its master record.
  - ◊ The master-detail relationship field is required on the page layout of the detail record.
  - ◊ By default, records can't be reparented in master-detail relationships. Administrators can, however, allow child records in master-detail relationships on custom objects to be reparented to different parent records by selecting the `Allow reparenting` option in the master-detail relationship definition.

  You can define master-detail relationships between custom objects or between a custom object and a standard object. However, the standard object cannot be on the detail side of a relationship with a custom object. In addition, you cannot create a master-detail relationship in which the User or Lead objects are the master.

  When you define a master-detail relationship, the custom object on which you are working is the detail side. Its data can appear as a custom related list on page layouts for the other object.

- **Many-to-many** — You can use master-detail relationships to model *many-to-many* relationships between any two objects. A many-to-many relationship allows each record of one object to be linked to multiple records from another object and vice versa. For example, you may have a custom object called "Bug" that relates to the standard case object such that a bug could be related to multiple cases and a case could also be related to multiple bugs. To create a many-to-many relationship, simply create a custom junction object with two master-detail relationship fields, each linking to the objects you want to relate. See the Salesforce online help for details.

  Custom objects with two master-detail relationships are supported in API version 11 and later.
- **Lookup (1:n)** — This type of relationship links two objects together, but has no effect on deletion or security. Unlike master-detail fields, lookup fields are not automatically required. When you define a lookup relationship, data from one object can appear as a custom related list on page layouts for the other object. See the Salesforce online help for details.

To create relationships, use the user interface or Salesforce Metadata API.

Relabeling Fields and Tabs and the API

The user interface allows you to change the labels on some fields and tabs. Although you cannot relabel fields or tabs using the API, you can retrieve the current values. To do so, issue a `describeSObjects()` call and inspect the label field of the returned DescribeSObjectResult.

Force.com AppExchange Object Prefixes and the API

If you have an unmanaged package and a managed package version becomes available, the API names of custom fields, custom objects, and Scontrol objects in the package changes, because a namespace prefix is added to each of these components to make them unique: *`name`*`__c` becomes *`prefix`*`__`*`name`*`__c`. To move from an unmanaged package to a managed package version of the same application, we advise that you export the data, uninstall the old package, install the new package, review the name changes, and import with the relevant mapping. For details, see the *ISVforce Guide*.

Custom Objects

In the user interface, you can extend your organization's data by defining custom objects. Custom objects are custom database tables that allow you to store information unique to your organization. For custom objects, the `custom` flag—a Boolean field in the describe results—is `true`.

Client applications with sufficient permissions can invoke API calls on existing custom objects. You can create new custom objects with the user interface, or by using the metadata WSDL with a client application or using the Force.com IDE. For more information about using the metadata WSDL to create new custom objects, see the *Force.com Metadata API Developer's Guide*. For more information about the Force.com IDE, see *Developer Force.*

Use the following topics to understand how the API interacts with custom objects and fields:

- Naming Conventions for Custom Objects
- Relationships Among Custom Objects
- Audit Fields for Custom Objects
- Sharing and Custom Objects
- Required Fields in Custom Objects
- Managed Packages and API Names

Naming Conventions for Custom Objects

Custom objects have an associated name field that is defined by your Salesforce administrator during setup. Custom objects must have unique names within your organization.

In the API, the names of custom objects are identified by a suffix of two underscores immediately followed by a lowercase "c" character. For example, a custom object labeled "Issue" in the Salesforce user interface is seen as `Issue__c` in that organization's WSDL.

Relationships change the naming convention, see Relationships Among Custom Objects for more information.

In order for a custom object record to appear in the Salesforce user interface, its name field must be populated. If you use the API to create a custom object record that does not have a name, then the ID of that record will be used as its name.

Relationships Among Custom Objects

Custom objects relate to other objects and behave just like standard objects, as described in Relationships Among Objects. For example, cascading deletes are supported in custom objects in a Master-Detail relationship.

Custom objects require special treatment so that they can participate in Relationship Queries. For the relationship field name of a custom object, `__r` is appended to the name to create the ID, and `__c` is appended to the name to create the parent object pointer. For example, if the relationship field name is MyRel, the name of the ID becomes `MyRelId__r`, and the parent object pointer becomes `MyRel__c`, and the relationship name is `MyRel__r`. For more information, see Understanding Relationship Names, Custom Objects, and Custom Fields in the *Salesforce SOQL and SOSL Reference Guide*.

The following table summarizes whether a standard object can be:

- The master in a master-detail relationship with a custom object. Master-detail relationships involve cascading deletes and sharing rules that are controlled by the parent.
- The lookup in a lookup relationship on a custom object. In other words, whether a custom object can have a lookup to the standard object.
- Extended with custom fields.

Standard Object	Master-Detail	Lookup	Custom Fields
Account	Yes	Yes	Yes
Campaign	Yes	Yes	Yes
Case	Yes	Yes	Yes
Contact	Yes	Yes	Yes
Contract	Yes	Yes	Yes
Event	No	No	Yes
Lead	No	No	Yes
Opportunity	Yes	Yes	Yes
Product2	No	Yes	Yes
Solution	Yes	Yes	Yes
Task	No	No	Yes
User	No	Yes	Yes

Custom objects can also have many-to-many relationships with other custom objects or standard objects. A many-to-many relationship allows each record of one object to be linked to multiple records from another object and vice versa. For more information, see Relationships Among Objects.

Audit Fields for Custom Objects

Custom objects can have the same audit fields as standard objects. Your organization must be API enabled and have the ability to create audit fields enabled, and you must have the “Modify All Data” permission.

When you create a custom object, the four audit fields `CreatedById`, `CreatedDate`, `LastModifiedById` and `LastModifiedDate` are automatically created and populated for the object. These fields are read only.

Note these restrictions:

- `CreatedDate` can't be greater than the `LastModifiedDate`.
- You can't set any date field to be greater than the current time.

For more information about audit fields, see System Fields.

Sharing and Custom Objects

A sharing rule object is created for each custom object that does not have a master-detail relationship to another object. They are similar to standard object sharing rules, for example AccountOwnerSharingRule. If the user creating the custom object has the “Manage Users” permission, a sharing rule object is automatically created for it.

Apex sharing reasons can be retrieved describing the custom object's sharing object, and examining the information in the `rowCause` field. The name of a sharing object for each custom object is of the form: *`MyObjectName`*__Share, similar to AccountShare and other standard object sharing objects.

Tags and Custom Objects

When a custom object is created, a Tag object related to it is also created. These object names are of the form: *`MyObjectName`*__Tag, similar to AccountTag and other standard object tag objects.

Required Fields in Custom Objects

In the user interface, you can mark a custom field as required, and this is also enforced in the API. Every custom field has a field `isRequired`, with a data type boolean. The default value is `false`. If set to `true`, every request must supply a value (or leave the current value) to this field. Otherwise, the request will fail. Once the value is set to `true`, the next time the field is edited or created, the validation applies, and if there is no value supplied or default value specified, the request fails.

To edit the `isRequired` field, you must log in as a user with the “Customize Application” permission.

If you change a custom object field to be required in an existing client application or integration, be sure that a value is always supplied for that field. For example, if the custom picklist field `Education Level` on the contact object is required, supply a default value for that custom field. If a required field does not have a specified or default value, an error with the status code `REQUIRED_FIELD_MISSING` is returned.

Managed Packages and API Names

If you have an unmanaged package and a managed package version becomes available, the API names of custom fields, custom objects, and Scontrol objects in the package changes, because a namespace prefix is added to each of these components to make them unique: *`name`*`__c` becomes *`prefix`*`__`*`name`*`__c`. To move from an unmanaged package to a managed package version of the same application, we advise that you export the data, uninstall the old package, install the new package, review the name changes, and import with the relevant mapping. For details, see the *ISVforce Guide*.

Chapter 3

API Call Basics

In this chapter ...

- Characteristics of API Calls
- Factors that Affect Data Access
- Package Version Settings

API calls represent specific operations that your client applications can invoke at runtime to perform tasks, for example:

- Query data in your organization.
- Add, update, and delete data.
- Obtain metadata about your data.
- Run utilities to perform administration tasks.

Using your development environment, you can construct Web service client applications that use standard Web service protocols to programmatically:

- Log in to the login server (`login()`) and receive authentication information to be used for subsequent calls
- Query your organization's information (`query()`, `queryAll()`, `queryMore()`, and `retrieve()` calls)
- Perform text searches across your organization's information (`search()` call)
- Create, update, and delete data (`create()`, `merge()`, `update()`, `upsert()`, `delete()`, and `undelete()` calls)
- Perform administrative tasks, such as retrieving user information (`getUserInfo()` call), changing passwords (`setPassword()` and `resetPassword()` calls), and getting the system time (`getServerTimestamp()` call)
- Replicate data locally (`getDeleted()` and `getUpdated()` calls)
- Obtain and navigate metadata about your organization's data (`describeGlobal()`, `describeSObject()`, `describeSObjects()`, `describeLayout()`, and `describeTabs()` calls)
- Work with approval processes (`process()`)
- Return category groups and categories from your organization (`describeDataCategoryGroups()` and `describeDataCategoryGroupStructures()`).

See Core Calls, Describe Calls, and Utility Calls for complete details about each call.

Characteristics of API Calls

All API calls are:

- **Service Requests and Responses**—Your client application prepares and submits a service request to the Force.com Web Service via the API, the Force.com Web Service processes the request and returns a response, and the client application handles the response.
- **Synchronous**—Once the API call is invoked, your client application waits until it receives a response from the service. Asynchronous calls are not supported.
- **Committed Automatically Versus Rollback on Error**—By default, every operation that writes to a Salesforce object is committed automatically. This is analogous to the AUTOCOMMMIT setting in SQL. For `create()`, `update()`, and `delete()` calls that attempt to write to multiple records for an object, the write operation for each record is treated as a separate transaction. For example, if a client application attempts to create two new accounts, they're created using mutually exclusive insert operations that succeed or fail individually, not as a group.

  For API version 20.0 and later, there is an AllOrNoneHeader header that allows a call to roll back all changes unless all records are processed successfully. This header is supported by the `create()`, `delete()`, `undelete()`, `update()`, and `upsert()` calls.

Note: The default behavior means that client applications may need to handle some failures: for example, if you create an opportunity that has shipments (a custom object), and the opportunity line item gets created but the shipment creation fails, if your business rules required all opportunities be created with shipment, your client application would need to roll back the opportunity creation. The easiest way to do this is to use AllOrNoneHeader.

Factors that Affect Data Access

When using the API, the following factors affect access to your organization's data:

Access

Your organization must be enabled for API access.

Objects may not be available until you contact salesforce.com and request access. For example Territory is visible only if territory management has been enabled in the application. Such requirements are in the "Usage" section for each object.

Sometimes a feature must be used once before objects related to it can be accessed with the API. For example, the `recordTypeIds` is available only after at least one record type has been created for your organization in the user interface.

To investigate data access issues, you can start by inspecting the WSDL:

- **Enterprise WSDL**: The generated enterprise WSDL file contains all of the objects that are available to your organization. By using the API, a client application can access objects that are defined in your enterprise WSDL file.
- **Partner WSDL**: When using the generated partner WSDL file, a client application can access objects that are returned in the `describeGlobal()` call.

Object-Level and Field-Level Security

The API respects object-level and field-level security configured in the user interface. You can access objects and fields only if the logged-in user's permissions and access settings allow such access. For example, fields that are not visible to a given user are not returned in a `query()` or `describeSObjects()` call. Similarly, read-only fields can't be updated.

User Permissions

A user attempting to access the API must have the permission "API Enabled" selected. It's selected by default.

Your client application logs in as a user called a *logged-in* user. The logged-in user's permissions grant or deny access to specific objects and fields in your organization:

- **Read**—Users can only view objects of this type.
- **Create**—Users can read and create objects of this type.
- **Edit**—Users can read and update objects of this type.
- **Delete**—Users can read, edit, and delete objects of this type.

User permissions do not affect field-level security. If field-level security specifies that a field is hidden, users with "Read" on that object can view only those fields that are not hidden on the record. In addition, users with "Read" on an object can view only those records that sharing settings allow. The one exception is the "Edit Read Only Fields" permission, which gives users the ability to edit fields marked as read only via field-level security.

Sharing

For most API calls, data that is outside of the logged-in user's sharing model is not returned. Users are granted the most permissive access that is available to them, either through organization-wide defaults or manual record sharing, just as in the application.

User Permissions that Override Sharing

- **View All**—Users can view all records associated with this object, regardless of sharing settings.
- **Modify All**—Users can read, edit, delete, transfer, and approve all records associated with this object, regardless of sharing settings.
- **Modify All Data**—users can read, edit, delete, transfer, and approve all records regardless of sharing settings. This permission is not an object-level permission, unlike "View All" and "Modify All."

To protect the security of your data, give the logged-in user only the permissions needed to successfully execute all the calls made by the application. For large integration applications, "Modify All Data" may speed up call response times. If you are loading a large number of records, use the Bulk API instead.

Related Objects

Some objects depend on other objects for permission. For example, AccountTeamMember follows sharing on the associated permission-assigned object such as the Account record. Similarly, a Partner depends on the permissions in the associated .

Ownership changes to a record do not automatically cascade to related records. For example, if ownership changes for a given Account, ownership does not then automatically change for any Contract associated with that Account—each ownership change must be made separately and explicitly by the client application.

Object Properties

To create an object with the `create()` call, the object's `createable` attribute must be set to `true`. To determine what operations are allowed on a given object, your client application can invoke the `describeSObjects()` call on the object and inspect the properties in the DescribeSObjectResult.

Note: `replicatable` allows`getUpdated()` and `getDeleted()` calls.

Page Layouts and Record Types

Requirements defined in the Salesforce user interface for page layouts and record types are not enforced by the API:

- Page layouts can specify whether a given field is required, but the API does not enforce such layout-specific field restrictions or validations in `create()` and `update()` calls. It's up to the client application to enforce any such constraints, if applicable.
- Record types can control which picklist values can be chosen in a given record and which page layouts users with different profiles can see. However, such rules that are configured and enforced in the user interface are not enforced in the API. For example, the API does not validate whether the value in a picklist field is allowed per any record type restrictions associated with the profile of the logged-in user. Similarly, the API does not prevent a client application from adding data to a particular field simply because that field does not appear in a layout associated with the profile of the logged-in user.

Referential Integrity

To ensure referential integrity, the API forces or prevents certain behaviors:

- ID values in reference fields are validated in `create()` and `update()` calls.
- If a client application deletes a record, then its children are automatically deleted as part of the call if the `cascadeDelete` property on ChildRelationship for that child has a value of `true`. For example, if a client application deletes an Opportunity, then any associated OpportunityLineItem records are also deleted. However, if an OpportunityLineItem is not deletable or is currently being used, then deletion of the parent Opportunity fails. For example, if a client application deletes an Invoice_Statement, then any associated Line_Item records are also deleted. However, if a Line_Item is not deletable or is currently being used, then deletion of the parent Invoice_Statement fails. Use DescribeSObjectResult to view the ChildRelationship value if you want to be sure what will be deleted.

  There are certain exceptions that prevent the execution of a `cascadeDelete`. For example, you can't delete an account if it has associated cases, if it has related opportunities that are owned by other users, or if associated contacts are enabled for the Customer Portal. In addition, if you attempt to delete an account that has closed/won opportunities owned by you or has active contracts, then the delete request for that record will fail.

Package Version Settings

When your API client is referencing components in managed packages, you can specify the version of each installed package that you want to reference for your integration. This allows your API client to continue to function with specific, known behavior even when you install subsequent versions of a package. You can use the PackageVersionHeader SOAP header to set different package versions for different calls, if necessary.

A package version is a number that identifies the set of components uploaded in a package. The version number has the format *`majorNumber.minorNumber.patchNumber`* (for example, 2.1.3). The major and minor numbers increase to a chosen value during every major release. The *`patchNumber`* is generated and updated only for a patch release. Publishers can use package versions to evolve the components in their managed packages gracefully by releasing subsequent package versions without breaking existing customer integrations using the package.

Default package versions for API calls provide fallback settings if package versions are not provided by an API call. Many API clients do not include package version information, so the default settings maintain existing behavior for these clients.

You can specify the default package versions for enterprise API and partner API calls. The enterprise WSDL is for customers who want to build an integration with their Salesforce organization only. It is strongly typed, which means that calls operate on objects and fields with specific data types, such as `int` and `string`. The partner WSDL is for customers, partners, and ISVs who want to build an integration that can work across multiple Salesforce organizations, regardless of their custom objects or fields. It is loosely typed, which means that calls operate on name-value pairs of field names and values instead of specific data types.

You must associate the enterprise WSDL with specific package versions to maintain existing behavior for clients. There are options for setting the package version bindings for an API call from client applications using either the enterprise or partner

WSDL. The package version information for API calls issued from a client application based on the enterprise WSDL is determined by the first match in the following settings.

1. The PackageVersionHeader SOAP header.
2. The SOAP endpoint contains a URL with a format of `serverName/services/Soap/c/api_version/ID` where `api_version` is the version of the API, such as 31.0, and `ID` encodes your package version selections when the enterprise WSDL was generated.
3. The default enterprise package version settings.

The partner WSDL is more flexible as it is used for integration with multiple organizations. If you choose the Not Specified option for a package version when configuring the default partner package versions, the behavior is defined by the latest installed package version. This means that behavior of package components, such as an Apex trigger, could change when a package is upgraded and that change would immediately impact the integration. Subscribers may want to select a specific version for an installed package for all partner API calls from client applications to ensure that subsequent installations of package versions do not affect their existing integrations.

The package version information for partner API calls is determined by the first match in the following settings.

1. The PackageVersionHeader SOAP header.
2. An API call from a Visualforce page uses the package versions set for the Visualforce page.
3. The default partner package version settings.

To configure default package versions for API calls:

1. From Setup, click **Develop** > **API**.
2. Click **Configure Enterprise Package Version Settings** or **Configure Partner Package Version Settings**. These links are only available if you have at least one managed package installed in your organization.
3. Select a `Package Version` for each of your installed managed packages. If you are unsure which package version to select, you should leave the default selection.
4. Click **Save**.

Note: Installing a new version of a package in your organization does not affect the current default settings.

Chapter 4

Error Handling

In this chapter ...

- Error Handling for Session Expiration
- More About Error Handling

The API calls return error data that your client application can use to identify and resolve runtime errors. If an error occurs during the invocation of most API calls, then the API provides the following types of error handling:

- For errors resulting from badly formed messages, failed authentication, or similar problems, the API returns a SOAP fault message with an associated ExceptionCode.
- For most calls, if the error occurs because of a problem specific to the query, the API returns an Error. For example, if a `create()` request contains more than 200 objects.

Error Handling for Session Expiration

When you sign on via the `login()` call, a new client session begins and a corresponding unique session ID is generated. Sessions expire automatically after a predetermined length of inactivity, which can be configured in Salesforce from Setup by clicking **Security Controls**. The default is 120 minutes (two hours). If you make an API call, the inactivity timer is reset to zero.

When your session expires, the exception code INVALID_SESSION_ID is returned. If this happens, you must invoke the `login()` call again.

More About Error Handling

For more information about errors, see the following topics:

- API Fault Element
- ExceptionCode
- Error

Chapter 5

Security and the API

In this chapter ...

- User Authentication
- User Profile and Permission Sets Configuration
- Security Token
- Sharing
- Implicit Restrictions for Objects and Fields
- API Access in Force.com AppExchange Packages
- Outbound Port Restrictions

Client applications that access your organization's Salesforce data are subject to the same security protections that are used in the Salesforce user interface. Additional protection is available for organizations that install Force.com AppExchange managed packages if those packages contain components that access Salesforce via the API.

User Authentication

Client applications must log in using valid credentials for an organization. The server authenticates these credentials and, if valid, provides the client application with:

- a `sessionId` that must be set into the session header so that all subsequent calls to the Web service are authenticated
- a URL address (`serverUrl`) for the client application's Web service requests

Salesforce.com supports only the Secure Sockets Layer (SSL) protocol SSLv3, the Transport Layer Security (TLS) protocol, and frontdoor.jsp. Ciphers must have a key length of at least 128 bits.

User Profile and Permission Sets Configuration

An organization's Salesforce administrator controls the availability of various features and views by configuring profiles and permission sets, and assigning users to them. To access the API (to issue calls and receive the call results), a user must be granted the "API Enabled" permission. Client applications can query or update only those objects and fields to which they have access via the permissions of the logged-in user.

To create, edit, or delete a profile, from Setup, click **Manage Users** > **Profiles** in the Salesforce user interface. To create, edit, or delete a permission set, from Setup, click **Manage Users** > **Permission Sets**.

Note: The Web services WSDL files return all available objects and fields for an organization.

Security Token

When users log in to Salesforce, either via the user interface, the API, or a desktop client such as Connect for Outlook, Salesforce for Outlook, Connect Offline, Connect for Office, Connect for Lotus Notes, or the Data Loader, Salesforce confirms that the login is authorized as follows:

1. Salesforce checks whether the user's profile has login hour restrictions. If login hour restrictions are specified for the user's profile, any login outside the specified hours is denied.
2. If the user has the "Two-Factor Authentication for User Interface Logins" permission, Salesforce prompts the user for a time-based token (which the user may also be prompted to create if it hasn't already been added to the account) upon logging in.
3. If the user has the "Two-Factor Authentication for API Logins" permission and a time-based token has been added to the account, Salesforce returns an error if a time-based token is not used to access the service in place of the standard security token.
4. Salesforce then checks whether the user's profile has IP address restrictions. If IP address restrictions are defined for the user's profile, any login from an undesignated IP address is denied, and any login from a specified IP address is allowed.
5. If profile-based IP address restrictions are not set, Salesforce checks whether the user is logging in from an IP address they have not used to access Salesforce before:
   - If the user's login is from a browser that includes a Salesforce cookie, the login is allowed. The browser will have the Salesforce cookie if the user has previously used that browser to log in to Salesforce, and has not cleared the browser cookies.
   - If the user's login is from an IP address in your organization's trusted IP address list, the login is allowed.
   - If the user's login is from neither a trusted IP address nor a browser with a Salesforce cookie, the login is blocked.

Whenever a login is blocked or returns an API login fault, Salesforce must verify the user's identity:

- For access via the user interface, the user is prompted toenter a token (also called a verification code) to confirm the user's identity.

  Note: Users aren't asked for a verification code the first time they log in to Salesforce.

- For access via the API or a client, users must add their security token (or time-based token if **Two-Factor Authentication on API Logins** is set on the user's profile and the user has added a time-based token to his or her account) to the end of their password in order to log in.

  A security token is an automatically-generated key from Salesforce. For example, if a user's password is `mypassword`, and the security token is `XXXXXXXXXX`, then the user must enter `mypasswordXXXXXXXXXX` to log in.

  Users can obtain their security token by changing their password or resetting their security token via the Salesforce user interface. When a user changes their password or resets their security token, Salesforce sends a new security token to the email address on the user's Salesforce record. The security token is valid until a user resets their security token, changes their password, or has their password reset.

  Tip: We recommend that you obtain your security token using the Salesforce user interface from a trusted network prior to attempting to access Salesforce from a new IP address.

For more information about tokens, see "Resetting Your Security Token" in the Salesforce online help.

When a user's password is changed, the user's security token is automatically reset. The user will experience a blocked login until he or she adds the automatically-generated security token to the end of his or her password or enters the new password after the administrator adds their IP address to the organization's list of trusted IP addresses.

If Single Sign-On (SSO) is enabled for your organization, users who access the API or a desktop client cannot log in to Salesforce unless their IP address is included on your organization's list of trusted IP addresses or on their profile, if their profile has IP address restrictions set. Futhermore, the delegated authentication authority usually handles login lockout policies for users with the "Uses Single Sign-On" permission. However, if the security token is enabled for your organization, then your organization's login lockout settings determine the number of times a user can attempt to log in with an invalid security token before being locked out of Salesforce. For more information, see "Setting Login Restrictions" and "Setting Password Policies" in the online help.

Sharing

In the Salesforce user interface, sharing refers to the act of granting read or write access to a user or group so that they can view or edit a record owned by other users, if the default organization access levels do not otherwise permit such access. All API calls respect the sharing model.

The following table describes the types of access levels.

API Value	Salesforce User Interface Label	API Picklist Label	Description
`None`	`Private`	`Private`	Only the record owner and Users above that role in the hierarchy can view and edit the record.
`Read`	`Read Only`	`Read Only`	All Users and Groups can view the record but not edit it. Only the owner and users above that role in the hierarchy can edit the record.

API Value	Salesforce User Interface Label	API Picklist Label	Description
`Edit`	`Read/Write`	`Read/Write`	All Users and Groups can view and edit the record.
`ReadEditTransfer`	`Read/Write/Transfer`	`Read/Write/Transfer`	All Users and Groups can view, edit, delete, and transfer the record. (Only available for cases and leads as an organization-wide default setting.)
`All`	`Full Access`	`Owner`	All Users and Groups can view, edit, transfer, delete, and share the record. (Only available for campaigns as an organization-wide default setting.)
`ControlledByParent`	`Controlled by Parent`	`Controlled By Parent`	(Contacts only.) All Users and Groups can perform an action (such as view, edit, or delete) on the contact based on whether he or she can perform that same action on the record associated with it.

Not all access levels are available for every object. See the Fields table for each object to learn which access levels are available, as well as other sharing details specific to that object.

For more information about sharing in general, see the Salesforce online help.

Note: In the API, you can create and update objects such as AccountShare and OpportunityShare that define sharing entries for records.

Implicit Restrictions for Objects and Fields

Certain objects can be created or deleted only in the Salesforce user interface. Other objects are read-only—client applications cannot `create()`, `delete()`, or `update()` such objects. Similarly, certain fields within some objects can be specified on `create()` but not on `update()`. Other fields are read-only—client applications cannot specify field values in `create()` or `update()` calls. For more information, see the respective object descriptions in Standard and Custom Object Basics.

API Access in Force.com AppExchange Packages

The API allows access to objects and calls based on the permissions of the user who logs into the API. To prevent security issues from arising when installed packages have components that access data via the API, Salesforce provides additional security:

- When a developer creates an AppExchange package with components that access the API, the developer can restrict the API access for those components.
- When an administrator installs an AppExchange package, the administrator can accept or reject the access. Rejecting the access cancels the installation.
- After an administrator installs a package, the administrator can restrict the API access of components in the package that access the API.

Editing API access for a package is done in the Salesforce user interface. For more information, see "Managing API and Dynamic Apex Access in Packages" in the Salesforce online help.

API access for a package affects the API requests originating from components within the package; it determines the objects that the API requests can access. If the API access for a package is not defined, then the objects that the API requests have access to are determined by the user's permissions.

The API access for a package never allows users to do more than the permissions granted to the user. API access in a package only reduces what the user's permissions allow.

Choosing `Restricted` for the `API Access` setting in a package affects the following:

- API access in a package overrides the following user permissions:
  - ◊ Author Apex
  - ◊ Customize Application
  - ◊ Edit HTML Templates
  - ◊ Edit Read Only Fields
  - ◊ Manage Billing
  - ◊ Manage Call Centers
  - ◊ Manage Categories
  - ◊ Manage Custom Report Types
  - ◊ Manage Dashboards
  - ◊ Manage Letterheads
  - ◊ Manage Package Licenses
  - ◊ Manage Public Documents
  - ◊ Manage Public List Views
  - ◊ Manage Public Reports
  - ◊ Manage Public Templates
  - ◊ Manage Users
  - ◊ Transfer Record
  - ◊ Use Team Reassignment Wizards
  - ◊ View Setup and Configuration
  - ◊ Weekly Export Data
- If `Read`, `Create`, `Edit`, and `Delete` access are not selected in the API access setting for objects, users do not have access to those objects from the package components, even if the user has the "Modify All Data" and "View All Data" permissions.
- A package with `Restricted` API access can't create new users.
- Salesforce denies access to Web service and `executeanonymous` requests from an AppExchange package that has `Restricted` access.

The following considerations also apply to API access in packages:

- Workflow rules and Apex triggers fire regardless of API access in a package.
- If a component is in more than one package in an organization, API access is unrestricted for that component in all packages in the organization regardless of the access setting.
- If Salesforce introduces a new standard object after you select restricted access for a package, access to the new standard object is not granted by default. You must modify the restricted access setting to include the new standard object.
- When you upgrade a package, changes to the API access are ignored even if the developer specified them. This ensures that the administrator installing the upgrade has full control. Installers should carefully examine the changes in package access in each upgrade during installation and note all acceptable changes. Then, because those changes are ignored, the administrator should manually apply any acceptable changes after installing an upgrade.

- S-controls are served by Salesforce and rendered inline in Salesforce. Because of this tight integration, there are several means by which an s-control in an installed package could escalate its privileges to the user's full privileges. In order to protect the security of organizations that install packages, s-controls have the following limitations:
  - For packages you are developing (that is, not installed from AppExchange), you can only add s-controls to packages with the default `Unrestricted` API access. Once a package has an s-control, you cannot enable `Restricted` API access.
  - For packages you have installed, you can enable access restrictions even if the package contains s-controls. However, access restrictions provide only limited protection for s-controls. Salesforce recommends that you understand the JavaScript in an s-control before relying on access restriction for s-control security.
  - If an installed package has `Restricted` API access, upgrades will be successful only if the upgraded version does not contain any s-controls. If s-controls are present in the upgraded version, you must change the currently installed package to `Unrestricted` API access.

To manage API access to packages, see "Managing API and Dynamic Apex Access in Packages" in the Salesforce online help.

Note: XML-RPC requests that originate from restricted packages will be denied access.

Outbound Port Restrictions

For security reasons, Salesforce restricts the outbound ports you may specify to one of the following:

- 80: This port only accepts HTTP connections.
- 443: This port only accepts HTTPS connections.
- 1024–66535 (inclusive): These ports accept HTTP or HTTPS connections.

The port restriction applies to any feature where a port is specified, for example outbound messages, AJAX proxy, or single-sign on.

Chapter 6

Using the Partner WSDL

In this chapter ...

- Obtaining the Partner WSDL File
- Calls and the Partner WSDL
- Objects, Fields, and Field Data and the Partner WSDL
- Queries and the Partner WSDL
- Namespaces in the Partner WSDL
- Package Versions and the Partner WSDL
- User Interface Themes
- Examples Using the Partner WSDL

The API provides two WSDLs to choose from:

- **Enterprise Web Services WSDL**—Used by enterprise developers to build client applications for a single Salesforce organization. The enterprise WSDL is strongly typed, which means that it contains objects and fields with specific data types, such as `int` and `string`. Customers who use the enterprise WSDL document must download and re-consume it whenever their organization makes a change to its custom objects or fields or whenever they want to use a different version of the API. To access the current WSDL for your organization, log in to your Salesforce organization and from Setup, click **Develop** > **API** > **Generate Enterprise WSDL**.
- **Partner Web Services WSDL**—Used for client applications that are metadata-driven and dynamic in nature. It is particularly—but not exclusively—useful to salesforce.com partners who are building client applications for multiple organizations. As a loosely typed representation of the Salesforce data model that works with name-value pairs of field names and values instead of specific data types, it can be used to access data within any organization. This WSDL is most appropriate for developers of clients that can issue a query call to get information about an object before the client acts on the object. The partner WSDL document only needs to be downloaded and consumed once per version of the API. To access the current WSDL for your organization, log in to your Salesforce organization and from Setup, click **Develop** > **API** > **Generate Partner WSDL**.

In general, the enterprise WSDL is more straightforward to use, while the partner WSDL is more flexible and dynamically adaptable to different organizations, allowing you to write a single application that can be used for multiple users and multiple organizations.

Obtaining the Partner WSDL File

To use the partner WSDL, download a copy of the file using either of the following methods:

- Obtain it from your organization's Salesforce administrator, or
- Generate from Setup in Salesforce in the **Develop** > **Force.com API** area of the user interface according to the instructions in Step 2: Generate or Obtain the Web Service WSDL.

While the enterprise WSDL file needs to be regenerated whenever custom fields or custom objects are added to an organization's Salesforce information, the partner WSDL file remains the same regardless of underlying changes in the organization's Salesforce data.

Calls and the Partner WSDL

The partner WSDL file defines exactly the same API calls found in the enterprise WSDL file. A client application using the partner WSDL will likely use the following API calls to determine an organization's metadata:

Task / Call	Description
`describeGlobal()`	Retrieves a list of available objects for your organization's data.
`describeLayout()`	Retrieves metadata about page layouts for the specified object type.
`describeSObject()`	`describeSObject()` has been superseded by `describeSObjects()`.
`describeSObjects()`	Use to obtain metadata for a given object. You can first call to retrieve a list of all objects for your organization, then iterate through the list and use to obtain metadata about individual objects.
`describeTabs()`	In the user interface, users have access to standard apps (and may also have access to custom apps) as listed in the Force.com app menu at the top of the page. Selecting a standard app or custom app in the user interface allows the user to switch between the listed apps at any time.

To explore an organization's metadata, a client application can:

1. Call `describeGlobal()` to obtain a list of available objects.
2. In the returned DescribeGlobalResult object, retrieve an array of DescribeGlobalSObjectResult objects by calling `sobjects`.
3. Get the sObject type name for each returned sObject by calling `name` on the DescribeGlobalSObjectResult objects.
4. The DescribeGlobalSObjectResult object provides some metadata about the sObject, such as whether the sObject is createable or updateable. If you want to get more information about particular sObjects, like their fields and child relationships, call `describeSObjects()` by passing it an array of the sObject type names that you're interested in obtaining more information about.

sObject Reference Reuse

An sObject reference can't be reused within a single operation.

Use a different reference. For example, the following code snippet creates an account and contact with a custom field and an event using two different references:

```
SObject account = new com.sforce.soap.partner.sobject.wsc.SObject();
account.setType("Account");
account.setField("Name","myAccount");
account.setField("XID1__c", "1");
```

```
SObject refAcc1 = new com.sforce.soap.partner.sobject.wsc.SObject();
refAcc1.setType("Account");
refAcc1.setField("XID1__c", "1");
SObject refAcc2 = new com.sforce.soap.partner.sobject.wsc.SObject();
refAcc2.setType("Account");
refAcc2.setField("XID1__c", "1");

SObject contact = new com.sforce.soap.partner.sobject.wsc.SObject();
contact.setType("Contact");
contact.setField("LastName", "LName");
contact.setField( "XID2__c", "2");
contact.setField( "Account", refAcc1);
SObject refCon = new com.sforce.soap.partner.sobject.wsc.SObject();
contact.setType("Contact");
contact.setField( "XID2__c", "2");


SObject event = new com.sforce.soap.partner.sobject.wsc.SObject();
contact.setType("Event");
contact.setField("Subject", "myEvent");
contact.setField( "ActivityDateTime", Calendar.getInstance());
contact.setField("DurationInMinutes", 60);
contact.setField("Who", refCon);
contact.setField("What", refAcc2);

client.create(new SObject[] { account, contact, event}); // exception thrown here
```

Any call that takes a parameter of the form `sObject[] sObjects` is subject to this limitation.

Objects, Fields, and Field Data and the Partner WSDL

While the enterprise WSDL file defines all of the specific objects (such as Account, Contact, and other objects described in Standard Objects) in a Salesforce organization, the partner WSDL file defines a single, generic object (sObject) that represents all of the objects. For a particular object, its type is defined in the `name` field in the returned DescribeSObjectResult.

With the partner WSDL, your client application code handles fields as arrays of name-value pairs that represent the field data. When referring to the names of individual fields, use the value in its `name` field of the Field type in the DescribeSObjectResult.

Languages vary in the way they handle name-value pairs and map typed values to the primitive XML data types defined in SOAP messages. With the enterprise WSDL, the mapping is handled implicitly. With the partner WSDL, however, you need to manage values and data types when building client applications. When specifying the value of a particular field, be sure to use a value that is valid for the field (range, format, and data type). Make sure that you understand the mapping between data types in your programming language with XML primitive data types (one of the values in the `SOAPType` field of the Field type in the DescribeSObjectResult).

Queries and the Partner WSDL

When using the `query()` call with the partner WSDL, consider the following guidelines:

- The queryString parameter is case-insensitive. The API will accept field names in the *fieldList* using any combination of uppercase and lowercase letters. However, in the QueryResult, the case of field names (both predefined and custom fields) will match exactly the value in the `name` field of the Field type in the DescribeSObjectResult. It is recommended that you use the proper case when specifying fields in the *fieldList*.
- For the partner WSDL, the ordering of fields in the QueryResult is determined by the field order in the *fieldList*, not the field order in the WSDL file.
- The *fieldList* cannot contain duplicate field names. For example:
  - ◊ Invalid (returns an error): `"SELECT Firstname, Lastname, Firstname FROM User"`

◊ Valid: `"SELECT Firstname, Lastname FROM User"`

- The QueryResult always contains all of the fields specified in the *fieldList*, even if some of the fields contain no data (`null`). Although SOAP allows you to omit fields that contain no values in the result set, the API always returns an array containing all fields.
- If you use the partner WSDL, a query that includes ID will return the ID field twice in the SOAP XML response data. Similarly, a query that does not include ID will return a single null ID field in the SOAP XML response data. For example, a query for `SELECT ID, FirstName, LastName FROM Contact` might return a SOAP XML response with records like:

```
<records xsi:type="sf:sObject" xmlns="urn:partner.soap.sforce.com">
    <sf:type>Contact</sf:type>
    <sf:Id>0038000000FrjoBQRW</sf:Id>
    <sf:Id>0038000000FrjoBQRW</sf:Id>
    <sf:FirstName>John</sf:FirstName>
    <sf:LastName>Smith</sf:LastName>
</records>
```

  This is expected behavior and something to be aware of if you are accessing the full SOAP XML response data and not using WSC to access the web service response.

Namespaces in the Partner WSDL

In XML, every tag has a defined namespace. In the `enterprise.wsdl`, namespaces are handled implicitly. When using API calls with the partner WSDL, however, you need to explicitly specify the correct namespaces for API calls, objects, and fields, and faults. This rule applies to predefined and custom objects and fields.

For	Namespace
API Calls	`urn:partner.soap.sforce.com`
sObjects	`urn:sobject.partner.soap.sforce.com`
Fields	`urn:sobject.partner.soap.sforce.com`
Faults	`urn:fault.partner.soap.sforce.com`

Package Versions and the Partner WSDL

The partner WSDL is loosely typed. This makes it more flexible for partners who want to integrate with multiple organizations. Default package versions for API calls provide fallback settings if package versions are not provided by an API call.

The behavior of a package in partner API calls is defined by the latest installed package version if the default value (`Not Specified`) is selected for the installed package. This means that behavior of package components, such as an Apex trigger, could change when a package is upgraded and that change would immediately impact the integration. Subscribers may want to select a specific version for an installed package for all partner API calls from client applications to ensure that subsequent installations of package versions do not affect their existing integrations.

An API client developer should communicate with the administrator of the default partner package version settings if these are two different roles in your organization and the developer recommends changing the settings. Alternatively, an API client developer can set the package versions in the PackageVersionHeader SOAP header for the client.

A partner that is developing a package that references another package should always supply version information for the base package in their partner API calls. This ensures that the extension package is not affected by a component being deprecated in the base package.

The package version information for partner API calls is determined by the first match in the following settings.

1. The PackageVersionHeader SOAP header.
2. An API call from a Visualforce page uses the package versions set for the Visualforce page.
3. The default partner package version settings.

To configure default package versions for API calls with the partner WSDL, see Package Version Settings.

User Interface Themes

Starting with the Winter '06 release, the online application supports multiple user interface themes. Currently there are two user interface themes, "Salesforce" and "Salesforce Classic." The `getUserInfo()` call returns a getUserInfoResult object, which includes the `userUiSkin` property. This property informs you of the user's current user interface theme.

Different themes can use different sets of icons and colors for things like tabs, quick actions, and objects in the user interface. Use the `describeQuickActions()`, `describeTabs()`, and `describeTheme()` calls and their return types to get information on theme icons and colors.

Stylesheets are available so partners can mimic the look and feel of the online user interface. For more information, see Styling Visualforce Pages in the *Visualforce Developer's Guide*.

Examples Using the Partner WSDL

This section includes examples in Java and C# for making API calls using the partner WSDL. Before running these samples, perform the following steps in the quick start tutorial to get the partner WSDL file and generate the proxy client code for your development environment.

- Step 2: Generate or Obtain the Web Service WSDL
- Step 3: Import the WSDL File Into Your Development Platform

After you generate the proxy client code and set up your development environment, you can start writing your client application. First, your application needs to log into the Salesforce service using the partner authentication endpoint. After a successful login, you can execute the sample methods.

For your convenience, template classes are provided, one in Java and one in C#, that make a login call. You can use them to execute the sample methods provided later in this section.

Sample template class for Java: This sample prompts the user to enter the username, password, and authentication endpoint. Next, it logs the user in. For the authentication endpoint URL, pass in the endpoint found in the partner WSDL file.

```
import com.sforce.soap.partner.PartnerConnection;
import com.sforce.soap.partner.sobject.*;
import com.sforce.soap.partner.*;
import com.sforce.ws.ConnectorConfig;
import com.sforce.ws.ConnectionException;
import com.sforce.soap.partner.Error;
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.BufferedReader;
import java.util.*;

public class PartnerSamples {
```

```
    PartnerConnection partnerConnection = null;
    private static BufferedReader reader =
        new BufferedReader(new InputStreamReader(System.in));

    public static void main(String[] args) {
        PartnerSamples samples = new PartnerSamples();
        if (samples.login()) {
            // Add calls to the methods in this class.
            // For example:
            // samples.querySample();
        }
    }

    private String getUserInput(String prompt) {
        String result = "";
        try {
          System.out.print(prompt);
          result = reader.readLine();
        } catch (IOException ioe) {
          ioe.printStackTrace();
        }
        return result;
    }

    private boolean login() {
        boolean success = false;
        String username = getUserInput("Enter username: ");
        String password = getUserInput("Enter password: ");
        String authEndPoint = getUserInput("Enter auth end point: ");

        try {
          ConnectorConfig config = new ConnectorConfig();
          config.setUsername(username);
          config.setPassword(password);

          config.setAuthEndpoint(authEndPoint);
          config.setTraceFile("traceLogs.txt");
          config.setTraceMessage(true);
          config.setPrettyPrintXml(true);

          partnerConnection = new PartnerConnection(config);

          success = true;
        } catch (ConnectionException ce) {
          ce.printStackTrace();
        } catch (FileNotFoundException fnfe) {
          fnfe.printStackTrace();
        }

        return success;
      }

    //
    // Add your methods here.
    //
}
```

Sample template class for C#: This sample prompts the user to enter the username and password. Next, it logs the user in. The project name for this sample is assumed to be `TemplatePartner` and the Web reference name `sforce`. If these values are different for your project, make sure to change the using directive to appropriate values for your project: `using your_project_name.web_reference_name;`.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
```

```
using System.Web.Services.Protocols;
using System.Collections;
using TemplatePartner.sforce;

namespace TemplatePartner
{
    class PartnerSamples
    {
        private SforceService binding;

        static void Main(string[] args)
        {
            PartnerSamples samples = new PartnerSamples();
            if (samples.login())
            {
            // Add calls to the methods in this class.
            // For example:
            // samples.querySample();
            }
        }

        private bool login()
        {
            Console.Write("Enter username: ");
            string username = Console.ReadLine();
            Console.Write("Enter password: ");
            string password = Console.ReadLine();

            // Create a service object
            binding = new SforceService();

            // Timeout after a minute
            binding.Timeout = 60000;

            // Try logging in
            LoginResult lr;
            try
            {

                Console.WriteLine("\nLogging in...\n");
                lr = binding.login(username, password);
            }

            // ApiFault is a proxy stub generated from the WSDL contract when
            // the web service was imported
            catch (SoapException e)
            {
                // Write the fault code to the console
                Console.WriteLine(e.Code);

                // Write the fault message to the console
                Console.WriteLine("An unexpected error has occurred: " + e.Message);

                // Write the stack trace to the console
                Console.WriteLine(e.StackTrace);

                // Return False to indicate that the login was not successful
                return false;
            }

            // Check if the password has expired
            if (lr.passwordExpired)
            {
                Console.WriteLine("An error has occurred. Your password has expired.");
                return false;
            }

            // Set the returned service endpoint URL
            binding.Url = lr.serverUrl;
```

```
            // Set the SOAP header with the session ID returned by
            // the login result. This will be included in all
            // API calls.
            binding.SessionHeaderValue = new SessionHeader();
            binding.SessionHeaderValue.sessionId = lr.sessionId;

            // Return true to indicate that we are logged in, pointed
            // at the right URL and have our security token in place.
            return true;
        }

    //
    // Add your methods here.
    //
}
```

This partner WSDL samples are:

- Sample query and queryMore Calls
- Sample search Call
- Sample create Call
- Sample update Call

Sample query and queryMore Calls

The following Java and C# examples show usage of the `query()` and `queryMore()` calls for the partner WSDL. Each example sets the batch size of the query to 250 items returned. It then performs a query call to get the first name and last name of all contacts and iterates through the contact records returned. For each contact, it writes the contact's first name and last name to the output, or only the last name if the first name is null. Finally, if there are more items to be returned by the query, as indicated by a `QueryResult.done` property value of `false`, it calls `queryMore()` to get the next batch of items, and repeats the process until no more records are returned.

To execute the sample method, you can use the corresponding Java or C# template class provided in Examples Using the Partner WSDL.

Java Example

```
public void querySample() {
    try {
        // Set query batch size
        partnerConnection.setQueryOptions(250);

        // SOQL query to use
        String soqlQuery = "SELECT FirstName, LastName FROM Contact";
        // Make the query call and get the query results
        QueryResult qr = partnerConnection.query(soqlQuery);

        boolean done = false;
        int loopCount = 0;
        // Loop through the batches of returned results
        while (!done) {
            System.out.println("Records in results set " + loopCount++
                    + " - ");
            SObject[] records = qr.getRecords();
            // Process the query results
            for (int i = 0; i < records.length; i++) {
                SObject contact = records[i];
                Object firstName = contact.getField("FirstName");
                Object lastName = contact.getField("LastName");
```

```
                if (firstName == null) {
                    System.out.println("Contact " + (i + 1) +
                            ": " + lastName
                    );
                } else {
                    System.out.println("Contact " + (i + 1) + ": " +
                            firstName + " " + lastName);
                }
            }
            if (qr.isDone()) {
                done = true;
            } else {
                qr = partnerConnection.queryMore(qr.getQueryLocator());
            }
        }
    } catch(ConnectionException ce) {
        ce.printStackTrace();
    }
    System.out.println("\nQuery execution completed.");
}
```

C# Example

```
public void querySample()
{
    try
    {
        QueryResult qr = null;
        binding.QueryOptionsValue = new sforce.QueryOptions();
        binding.QueryOptionsValue.batchSize = 250;
        binding.QueryOptionsValue.batchSizeSpecified = true;

        qr = binding.query("SELECT FirstName, LastName FROM Contact");

        bool done = false;
        int loopCount = 0;
        while (!done)
        {
            Console.WriteLine("\nRecords in results set " +
                Convert.ToString(loopCount++)
                    + " - ");
            // Process the query results
            for (int i = 0; i < qr.records.Length; i++)
            {
                sforce.sObject con = qr.records[i];
                string fName = con.Any[0].InnerText;
                string lName = con.Any[1].InnerText;
                if (fName == null)
                    Console.WriteLine("Contact " + (i + 1) + ": " + lName);
                else
                    Console.WriteLine("Contact " + (i + 1) + ": " + fName
                        + " " + lName);
            }

            if (qr.done)
                done = true;
            else
                qr = binding.queryMore(qr.queryLocator);
        }
    }
    catch (SoapException e)
    {
        Console.WriteLine("An unexpected error has occurred: " + e.Message +
            " Stack trace: " + e.StackTrace);
    }
    Console.WriteLine("\nQuery execution completed.");
}
```

Sample search Call

The following Java and C# examples show how to use the `search()` call for the partner WSDL. Each example accepts a phone number string value that is used in the SOQL query. The search call looks for phone fields that match the passed in phone value in all contacts, leads, and accounts. Next, the example iterates through the returned search results that contain the matching records, adds them to arrays, and writes their field values to the console. The record fields returned correspond to the fields specified in the SOQL query for each record type.

To execute the sample method, you can use the corresponding Java or C# template class provided in Examples Using the Partner WSDL.

Java Example

```
public void searchSample(String phoneNumber) {
    try {
        // Example of phoneNumber format: 4155551212
        String soslQuery =
            "FIND {" + phoneNumber + "} IN Phone FIELDS " +
            "RETURNING " +
            "Contact(Id, Phone, FirstName, LastName), " +
            "Lead(Id, Phone, FirstName, LastName)," +
            "Account(Id, Phone, Name)";
        // Perform SOSL query
        SearchResult sResult = partnerConnection.search(soslQuery);
        // Get the records returned by the search result
        SearchRecord[] records = sResult.getSearchRecords();
        // Create lists of objects to hold search result records
        List<SObject> contacts = new ArrayList<SObject>();
        List<SObject> leads = new ArrayList<SObject>();
        List<SObject> accounts = new ArrayList<SObject>();

        // Iterate through the search result records
        // and store the records in their corresponding lists
        // based on record type.
        if (records != null && records.length > 0) {
          for (int i = 0; i < records.length; i++){
            SObject record = records[i].getRecord();
            if (record.getType().toLowerCase().equals("contact")) {
              contacts.add(record);
            } else if (record.getType().toLowerCase().equals("lead")){
              leads.add(record);
            } else if (record.getType().toLowerCase().equals("account")) {
              accounts.add(record);
            }
          }
          // Display the contacts that the search returned
          if (contacts.size() > 0) {
            System.out.println("Found " + contacts.size() +
                " contact(s):");
            for (SObject contact : contacts) {
              System.out.println(contact.getId() + " - " +
                  contact.getField("FirstName") + " " +
                  contact.getField("LastName") + " - " +
                  contact.getField("Phone")
              );
            }
          }
          // Display the leads that the search returned
          if (leads.size() > 0) {
            System.out.println("Found " + leads.size() +
                " lead(s):");
            for (SObject lead : leads) {
              System.out.println(lead.getId() + " - " +
                  lead.getField("FirstName") + " " +
```

```
                  lead.getField("LastName") + " - " +
                  lead.getField("Phone")
              );
            }
          }
          // Display the accounts that the search returned
          if (accounts.size() > 0) {
            System.out.println("Found " +
                accounts.size() + " account(s):");
            for (SObject account : accounts) {
              System.out.println(account.getId() + " - " +
                  account.getField("Name") + " - " +
                  account.getField("Phone")
              );
            }
          }
        } else {
          // The search returned no records
          System.out.println("No records were found for the search.");
        }
      } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

C# Example

```
public void searchSample(String phoneNumber)
{
    try
    {
        // Example of phoneNumber format: 4155551212
        String soslQuery =
            "FIND {" + phoneNumber + "} IN Phone FIELDS " +
            "RETURNING " +
            "Contact(Id, Phone, FirstName, LastName), " +
            "Lead(Id, Phone, FirstName, LastName)," +
            "Account(Id, Phone, Name)";
        // Perform SOSL query
        SearchResult sResult = binding.search(soslQuery);
        // Get the records returned by the search result
        SearchRecord[] records = sResult.searchRecords;
        // Create lists of objects to hold search result records
        ArrayList contacts = new System.Collections.ArrayList();
        ArrayList leads = new System.Collections.ArrayList();
        ArrayList accounts = new System.Collections.ArrayList();

        // Iterate through the search result records
        // and store the records in their corresponding lists
        // based on record type.
        if ((records != null) && (records.Length > 0))
        {
            for (int i = 0; i < records.Length; i++)
            {
                sObject record = records[i].record;

                if (record.type.ToLower().Equals("contact"))
                {
                    contacts.Add(record);
                }
                else if (record.type.ToLower().Equals("lead"))
                {
                    leads.Add(record);
                }
                else if (record.type.ToLower().Equals("account"))
                {
                    accounts.Add(record);
```

```
                }
            }
            // Display the contacts that the search returned
            if (contacts.Count > 0)
            {
                Console.WriteLine("Found " + contacts.Count + " contact(s):");
                for (int i = 0; i < contacts.Count; i++)
                {
                    sObject c = (sObject)contacts[i];
                    Console.WriteLine(c.Any[0].InnerText + " - " +
                                c.Any[2].InnerText + " " +
                                c.Any[3].InnerText + " - " + c.Any[1].InnerText);
                }
            }
            // Display the leads that the search returned
            if (leads.Count > 0)
            {
                Console.WriteLine("Found " + leads.Count + " lead(s):");
                for (int i = 0; i < leads.Count; i++)
                {
                    sObject l = (sObject)leads[i];
                    Console.WriteLine(l.Any[0].InnerText + " - " +
                                l.Any[2].InnerText + " " +
                                l.Any[3].InnerText + " - " + l.Any[1].InnerText);
                }
            }
            // Display the accounts that the search returned
            if (accounts.Count > 0)
            {
                Console.WriteLine("Found " + accounts.Count + " account(s):");
                for (int i = 0; i < accounts.Count; i++)
                {
                    sObject a = (sObject)accounts[i];
                    Console.WriteLine(a.Any[0].InnerText + " - " +
                                    a.Any[2].InnerText + " - " +
                                    a.Any[1].InnerText);
                }
            }
        }
        else
        {
            // The search returned no records
            Console.WriteLine("No records were found for the search.");
        }
    }
    catch (SoapException e)
    {
        Console.WriteLine("An unexpected error has occurred: " + e.Message +
            " Stack trace: " + e.StackTrace);
    }
}
```

Sample create Call

The following Java and C# examples show how to use the `create()` call for the partner WSDL. Each example creates a contact record with several fields. It iterates through the results of the create call and checks whether the operation was successful or not. If the create operation was successful, it writes the ID of the contact created to the console. Otherwise, it iterates through the errors and writes details of each error to the console. In this case, the output of the example is the ID of the new contact.

To execute the sample method, you can use the corresponding Java or C# template class provided in Examples Using the Partner WSDL.

Java Example

```
public String createSample() {
    String result = null;
    try {
        // Create a new sObject of type Contact
           // and fill out its fields.
        SObject contact = new SObject();
        contact.setType("Contact");
        contact.setField("FirstName", "Otto");
        contact.setField("LastName", "Jespersen");
        contact.setField("Salutation", "Professor");
        contact.setField("Phone", "(999) 555-1234");
        contact.setField("Title", "Philologist");

        // Add this sObject to an array
        SObject[] contacts = new SObject[1];
        contacts[0] = contact;
        // Make a create call and pass it the array of sObjects
        SaveResult[] results = partnerConnection.create(contacts);

        // Iterate through the results list
        // and write the ID of the new sObject
        // or the errors if the object creation failed.
        // In this case, we only have one result
        // since we created one contact.
        for (int j = 0; j < results.length; j++) {
            if (results[j].isSuccess()) {
                result = results[j].getId();
                System.out.println(
                    "\nA contact was created with an ID of: " + result
                );
             } else {
                // There were errors during the create call,
                // go through the errors array and write
                // them to the console
                for (int i = 0; i < results[j].getErrors().length; i++) {
                    Error err = results[j].getErrors()[i];
                    System.out.println("Errors were found on item " + j);
                    System.out.println("Error code: " +
                        err.getStatusCode().toString());
                    System.out.println("Error message: " + err.getMessage());
                }
             }
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
    return result;
}
```

C# Example

```
public void createSample()
{
    try
    {
        // Create a new sObject of type Contact
        // and fill out its fields.
        sObject contact = new sforce.sObject();
        System.Xml.XmlElement[] contactFields = new System.Xml.XmlElement[6];

        // Create the contact's fields
        System.Xml.XmlDocument doc = new System.Xml.XmlDocument();
        contactFields[0] = doc.CreateElement("FirstName");
        contactFields[0].InnerText = "Otto";
        contactFields[1] = doc.CreateElement("LastName");
```

```
        contactFields[1].InnerText = "Jespersen";
        contactFields[2] = doc.CreateElement("Salutation");
        contactFields[2].InnerText = "Professor";
        contactFields[3] = doc.CreateElement("Phone");
        contactFields[3].InnerText = "(999) 555-1234";
        contactFields[4] = doc.CreateElement("Title");
        contactFields[4].InnerText = "Philologist";

        contact.type = "Contact";
        contact.Any = contactFields;

        // Add this sObject to an array
        sObject[] contactList = new sObject[1];
        contactList[0] = contact;

        // Make a create call and pass it the array of sObjects
        SaveResult[] results = binding.create(contactList);
        // Iterate through the results list
        // and write the ID of the new sObject
        // or the errors if the object creation failed.
        // In this case, we only have one result
        // since we created one contact.
        for (int j = 0; j < results.Length; j++)
        {
            if (results[j].success)
            {
                Console.Write("\nA contact was created with an ID of: "
                                + results[j].id);
            }
            else
            {
                // There were errors during the create call,
                // go through the errors array and write
                // them to the console
                for (int i = 0; i < results[j].errors.Length; i++)
                {
                    Error err = results[j].errors[i];
                    Console.WriteLine("Errors were found on item " + j.ToString());
                    Console.WriteLine("Error code is: " + err.statusCode.ToString());
                    Console.WriteLine("Error message: " + err.message);
                }
            }
        }
    }
    catch (SoapException e)
    {
        Console.WriteLine("An unexpected error has occurred: " + e.Message +
            " Stack trace: " + e.StackTrace);
    }
}
```

Sample update Call

The following Java and C# examples show how to use the `update()` call for the Partner WSDL. Each example takes the ID of the contact to update as an argument. It creates two sObject records of type Contact—one to hold the valid passed in ID and the other has an invalid ID. Next, it sets a new phone number for the valid contact and `null` for the last name of the invalid contact. It then makes the update call and iterates through the results. For a successful update operation, it writes the ID of the contact that got updated. For a failed update operation, it writes the details of all returned errors to the console. In this case, the output is the ID of the contact that was successfully updated and an error for the invalid contact update.

To execute the sample method, you can use the corresponding Java or C# template class provided in Examples Using the Partner WSDL.

Java Example

```
public void updateSample(String id) {
  try {
      // Create an sObject of type contact
      SObject updateContact = new SObject();
      updateContact.setType("Contact");

      // Set the ID of the contact to update
      updateContact.setId(id);
      // Set the Phone field with a new value
      updateContact.setField("Phone", "(415) 555-1212");

      // Create another contact that will cause an error
      // because it has an invalid ID.
      SObject errorContact = new SObject();
      errorContact.setType("Contact");
      // Set an invalid ID on purpose
      errorContact.setId("SLFKJLFKJ");
      // Set the value of LastName to null
      errorContact.setFieldsToNull(new String[] {"LastName"});

      // Make the update call by passing an array containing
      // the two objects.
      SaveResult[] saveResults = partnerConnection.update(
         new SObject[] {updateContact, errorContact}
      );
      // Iterate through the results and write the ID of
      // the updated contacts to the console, in this case one contact.
      // If the result is not successful, write the errors
      // to the console. In this case, one item failed to update.
      for (int j = 0; j < saveResults.length; j++) {
          System.out.println("\nItem: " + j);
          if (saveResults[j].isSuccess()) {
              System.out.println("Contact with an ID of " +
                      saveResults[j].getId() + " was updated.");
          }
          else {
            // There were errors during the update call,
            // go through the errors array and write
            // them to the console.
            for (int i = 0; i < saveResults[j].getErrors().length; i++) {
              Error err = saveResults[j].getErrors()[i];
              System.out.println("Errors were found on item " + j);
              System.out.println("Error code: " +
                  err.getStatusCode().toString());
              System.out.println("Error message: " + err.getMessage());
            }
          }
      }
  } catch (ConnectionException ce) {
      ce.printStackTrace();
  }
}
```

For more information about `setFieldsToNull` (or its equivalent in client tools other than WSC), see `fieldsToNull` and Resetting Values to null.

C# Example

```
public void updateSample(String id) {
    try
    {
        // Create an sObject of type contact
        sObject updateContact = new sObject();
        updateContact.type = "Contact";
```

```
        // Set the ID of the contact to update
        updateContact.Id = id;
        // Set the Phone field to a new value.
        // The Phone field needs to be created as an XML element.
        System.Xml.XmlDocument doc = new System.Xml.XmlDocument();
        System.Xml.XmlElement phoneField = doc.CreateElement("Phone");
        phoneField.InnerText = "(415) 555-1212";

        // Add the Phone field to the contact
        updateContact.Any = new System.Xml.XmlElement[] {phoneField};

        // Create another contact that will cause an error
        // because it has an invalid ID.
        sObject errorContact = new sObject();
        errorContact.type = "Contact";
        // Set an invalid ID on purpose
        errorContact.Id = "SLFKJLFKJ";
        // Set the value of LastName to null
        errorContact.fieldsToNull = new String[] { "LastName" };

        // Make the update call by passing an array containing
        // the two objects.
        SaveResult[] saveResults = binding.update(
            new sObject[] {updateContact, errorContact});
        // Iterate through the results and write the ID of
        // the updated contacts to the console, in this case one contact.
        // If the result is not successful, write the errors
        // to the console. In this case, one item failed to update.
        for (int j = 0; j < saveResults.Length; j++) {
            Console.WriteLine("\nItem: " + j);
            if (saveResults[j].success)
            {
                Console.WriteLine("Contact with an ID of " +
                saveResults[j].id + " was updated.");
            }
            else
            {
                // There were errors during the update call,
                // go through the errors array and write
                // them to the console.
                for (int i = 0; i < saveResults[j].errors.Length; i++) {
                    Error err = saveResults[j].errors[i];
                    Console.WriteLine("Errors were found on item " + j.ToString());
                    Console.WriteLine("Error code: " +
                        err.statusCode.ToString());
                    Console.WriteLine("Error message: " + err.message);
                }
            }
        }
    }
    catch (SoapException e)
    {
        Console.WriteLine("An unexpected error has occurred: " + e.Message +
            " Stack trace: " + e.StackTrace);
    }
}
```

REFERENCE

Chapter 7

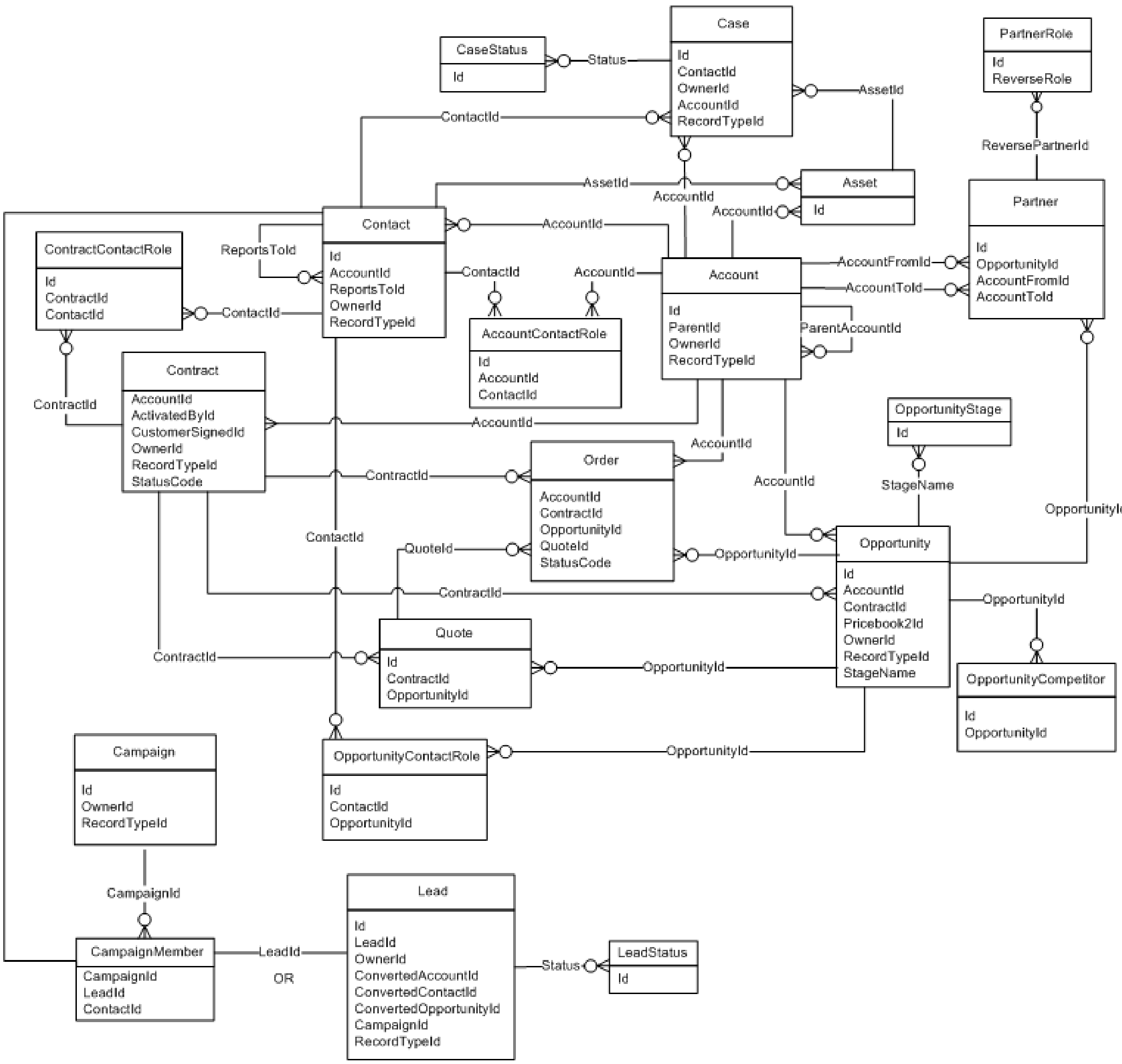
Data Model

The entity relationship diagrams (ERDs) for standard Salesforce objects in this section illustrate important relationships between objects. The available ERDs are:

- Sales Objects—includes accounts, contacts, opportunities, leads, campaigns, and other related objects
- Task and Event Objects—includes tasks and events and their related objects
- Support Objects—includes cases and solutions and their related objects
- Salesforce Knowledge Objects—includes view and vote statistics, article versions, and other related objects
- Document, Note, and Attachment Objects—includes documents, notes, and attachments and their related objects
- User, Sharing, and Permission Objects—includes users, profiles, and roles
- Profile and Permission Objects—includes users, profiles, permission sets, and related permission objects
- Record Type Objects—includes record types and business processes and their related objects
- Product and Schedule Objects—includes opportunities, products, and schedules
- Sharing and Team Selling Objects—includes account teams, opportunity teams, and sharing objects
- Customizable Forecasting Objects—includes forecasts and related objects
- Forecasts Objects—includes objects for Collaborative Forecasts.
- Territory Management—includes territories and related objects
- Process Objects—includes approval processes and related objects
- Content Objects—includes content and libraries and their related objects
- Chatter Feed Objects—includes objects related to feeds

Each entity relationship diagram includes links to the topics that describe the fields in objects related to the diagram. The data model for your custom objects depends on what you create.

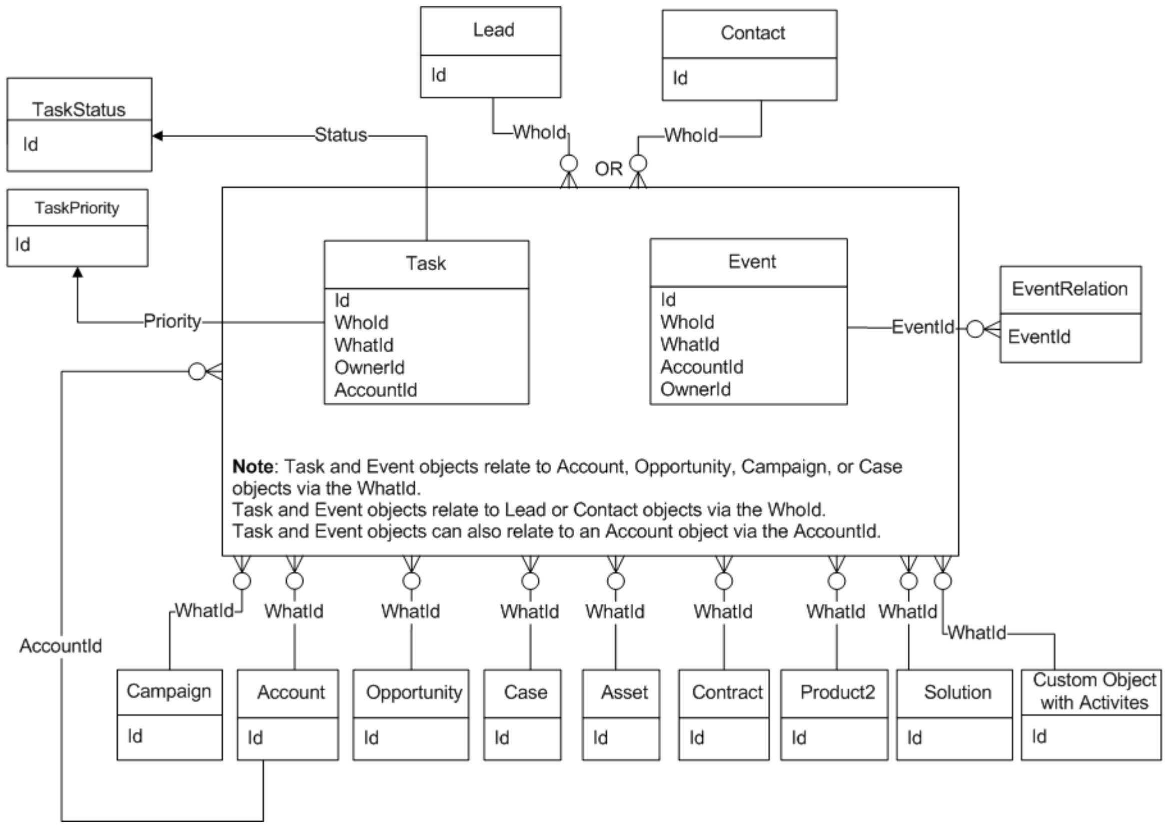
Sales Objects



See Also:

- *Standard Objects*
- *Data Model*

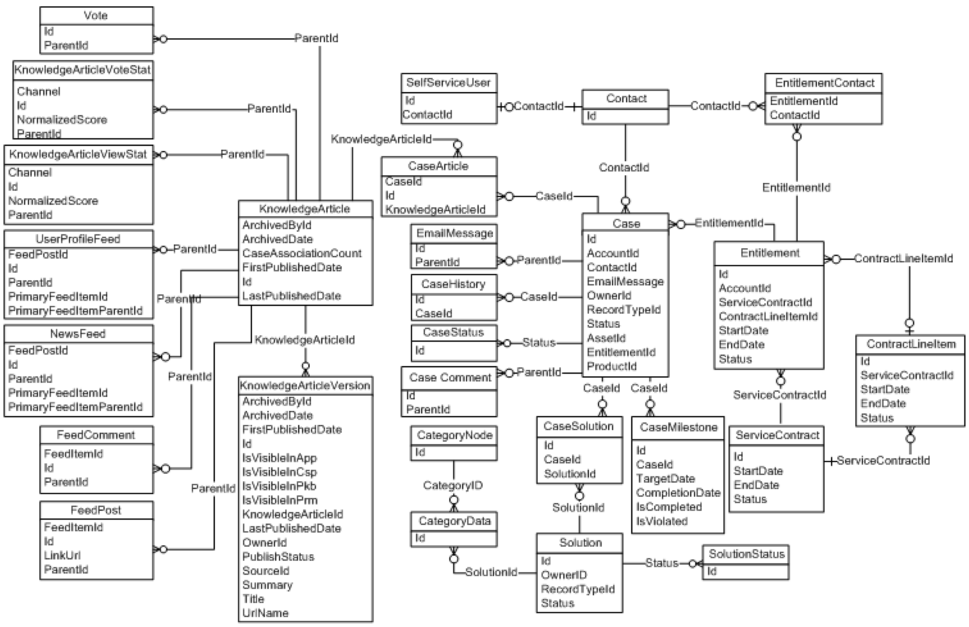
Task and Event Objects



See Also:

Standard Objects
Data Model

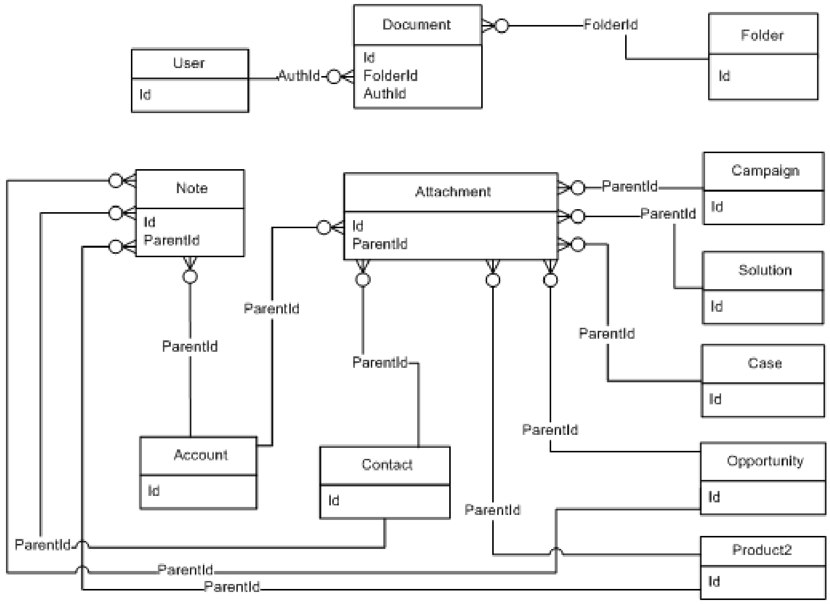
Support Objects



See Also:

Standard Objects
Data Model

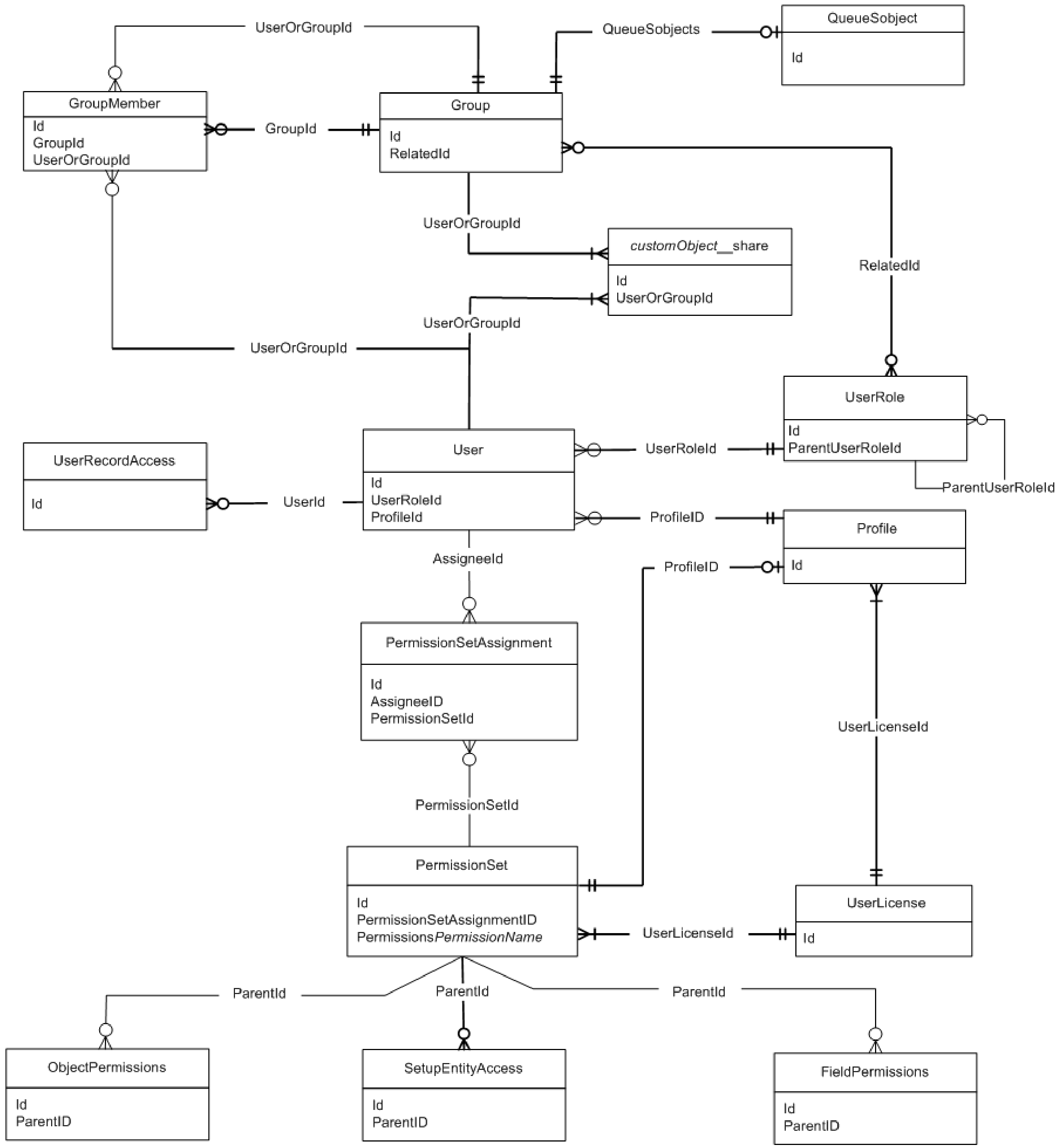
Document, Note, and Attachment Objects



See Also:

Standard Objects
Data Model

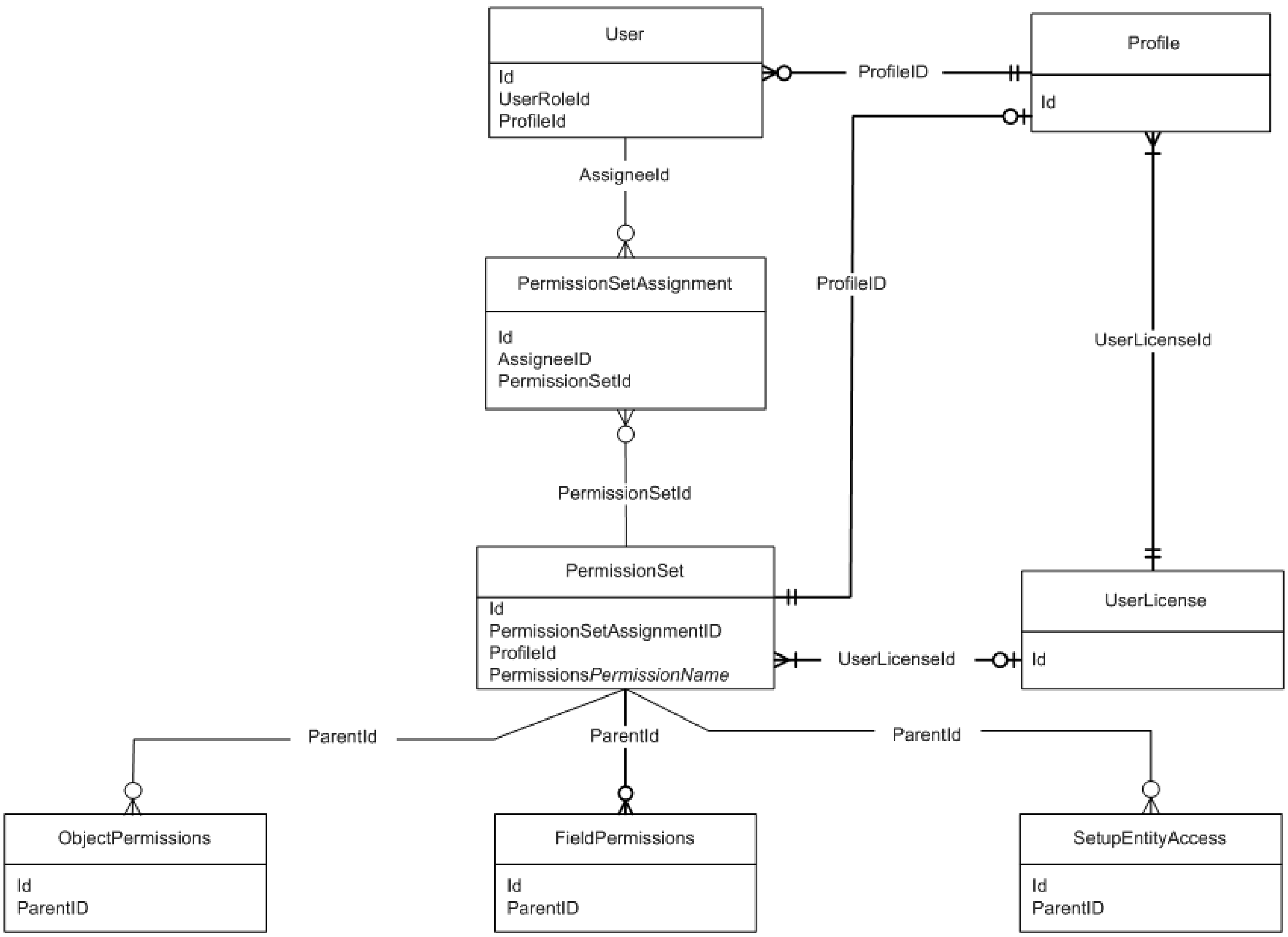
User, Sharing, and Permission Objects



See Also:

Standard Objects
Data Model
Profile and Permission Objects

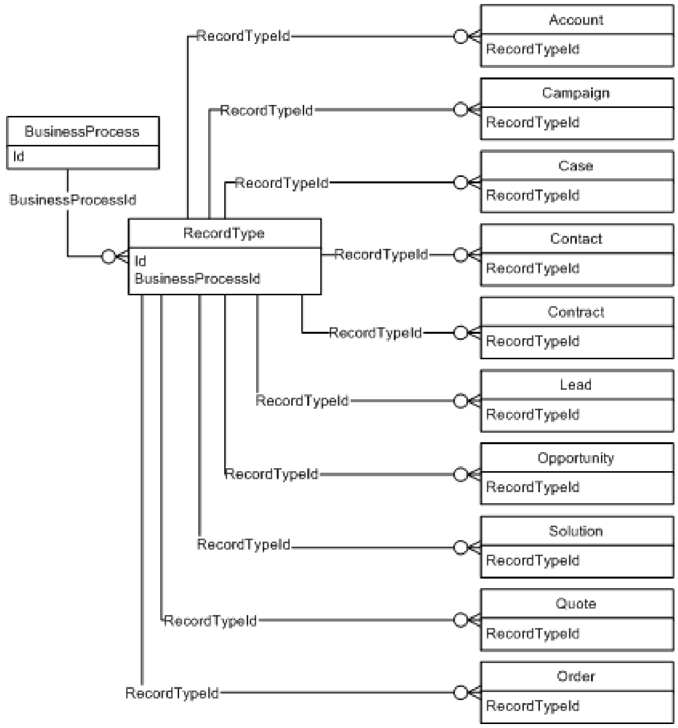
Profile and Permission Objects



See Also:

Standard Objects
Data Model

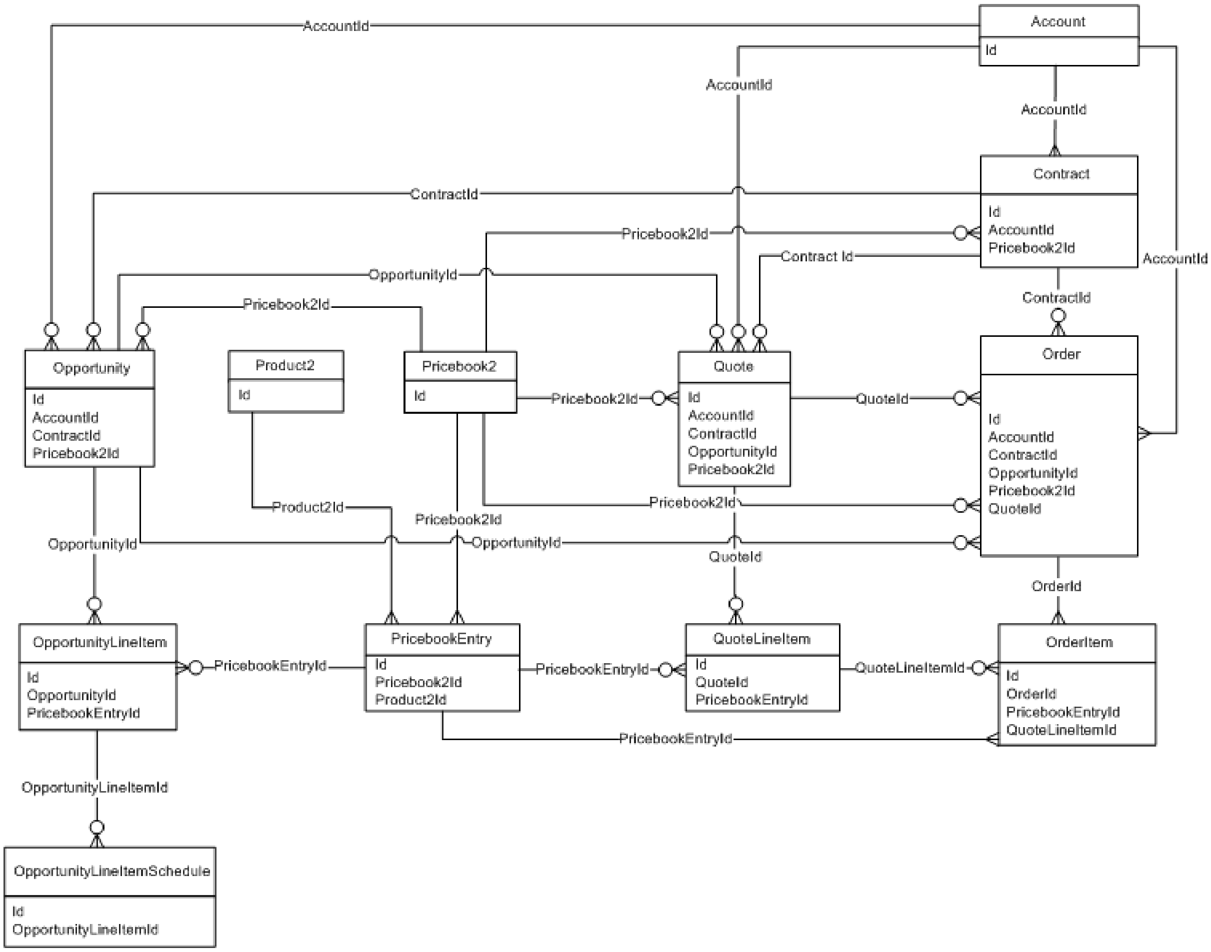
Record Type Objects



See Also:

Standard Objects
Data Model

Product and Schedule Objects

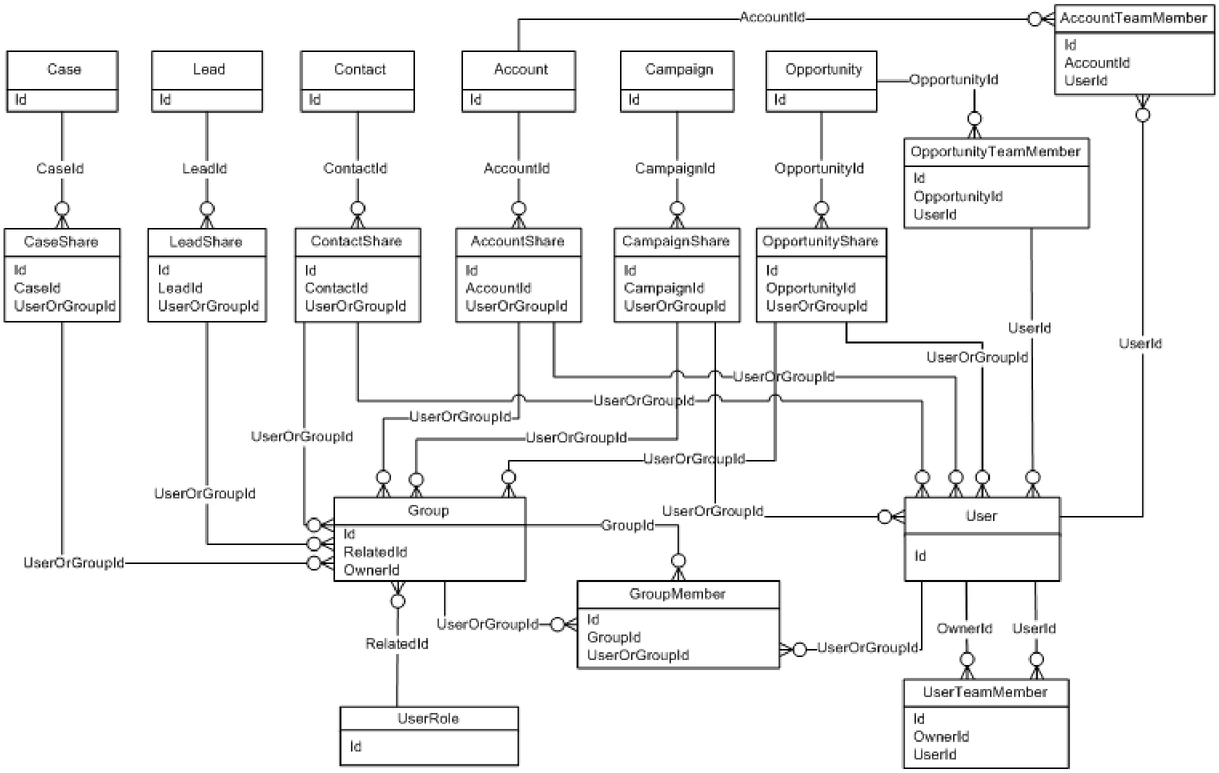


Create a separate PricebookEntry for each currency and price combination.

See Also:

Standard Objects
Data Model

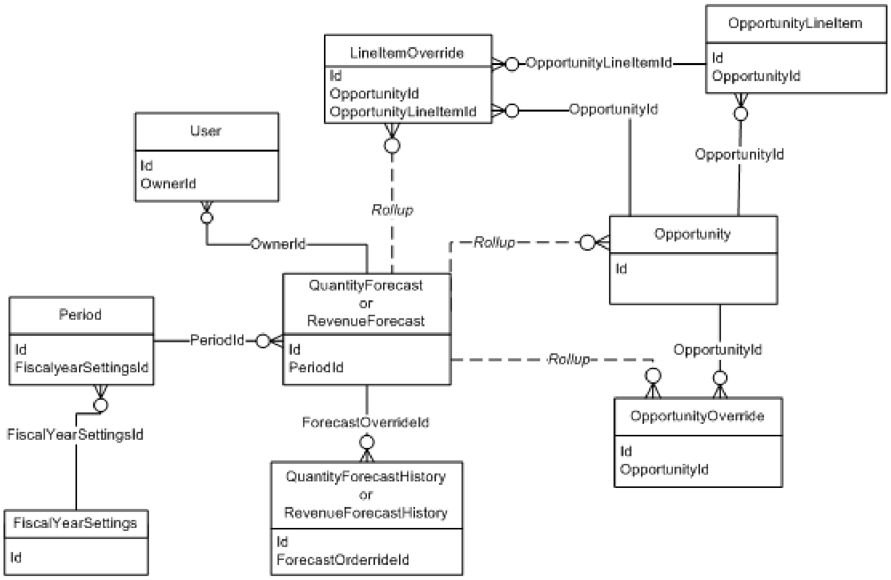
Sharing and Team Selling Objects



See Also:

Standard Objects
Data Model

Customizable Forecasting Objects



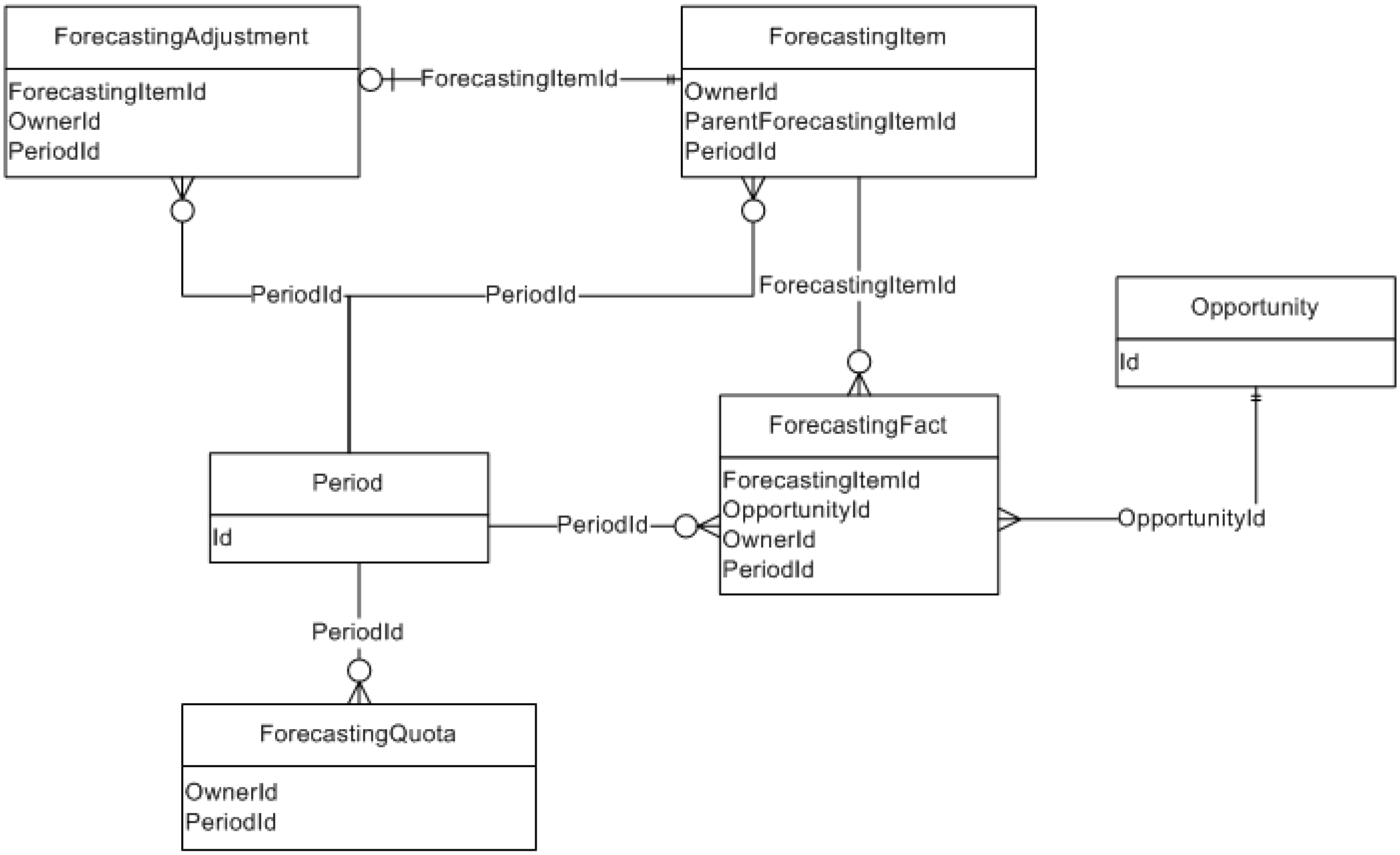
See Also:

Standard Objects

Data Model

Forecasts Objects

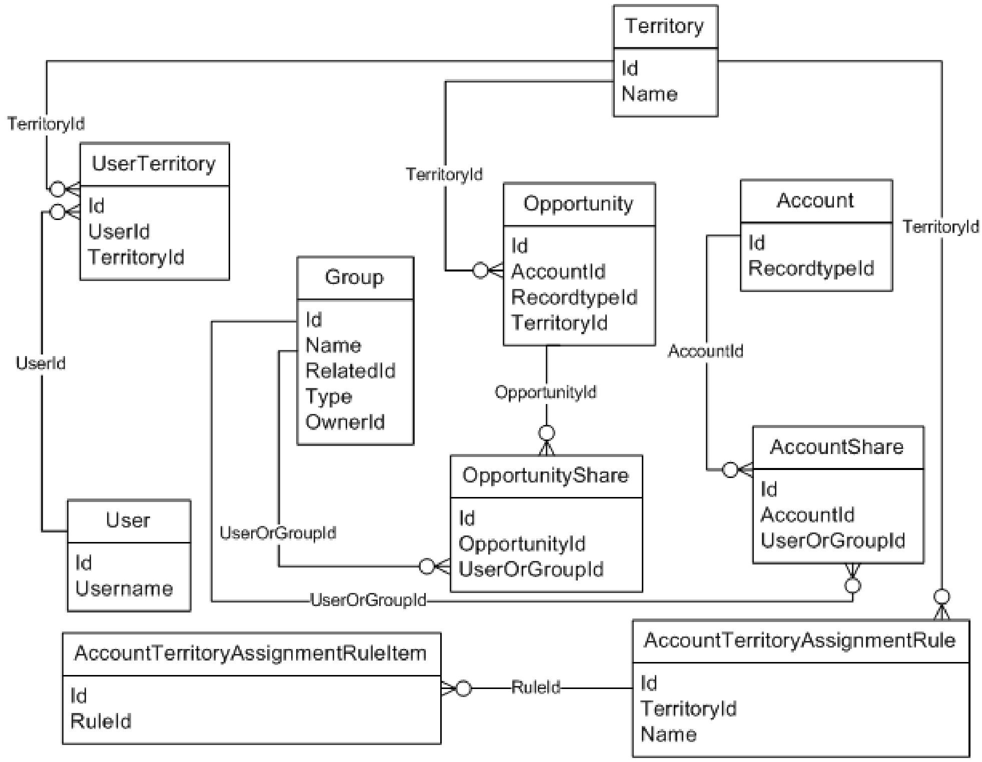
Note: This information only applies to Collaborative Forecasts.



See Also:

Standard Objects
Data Model

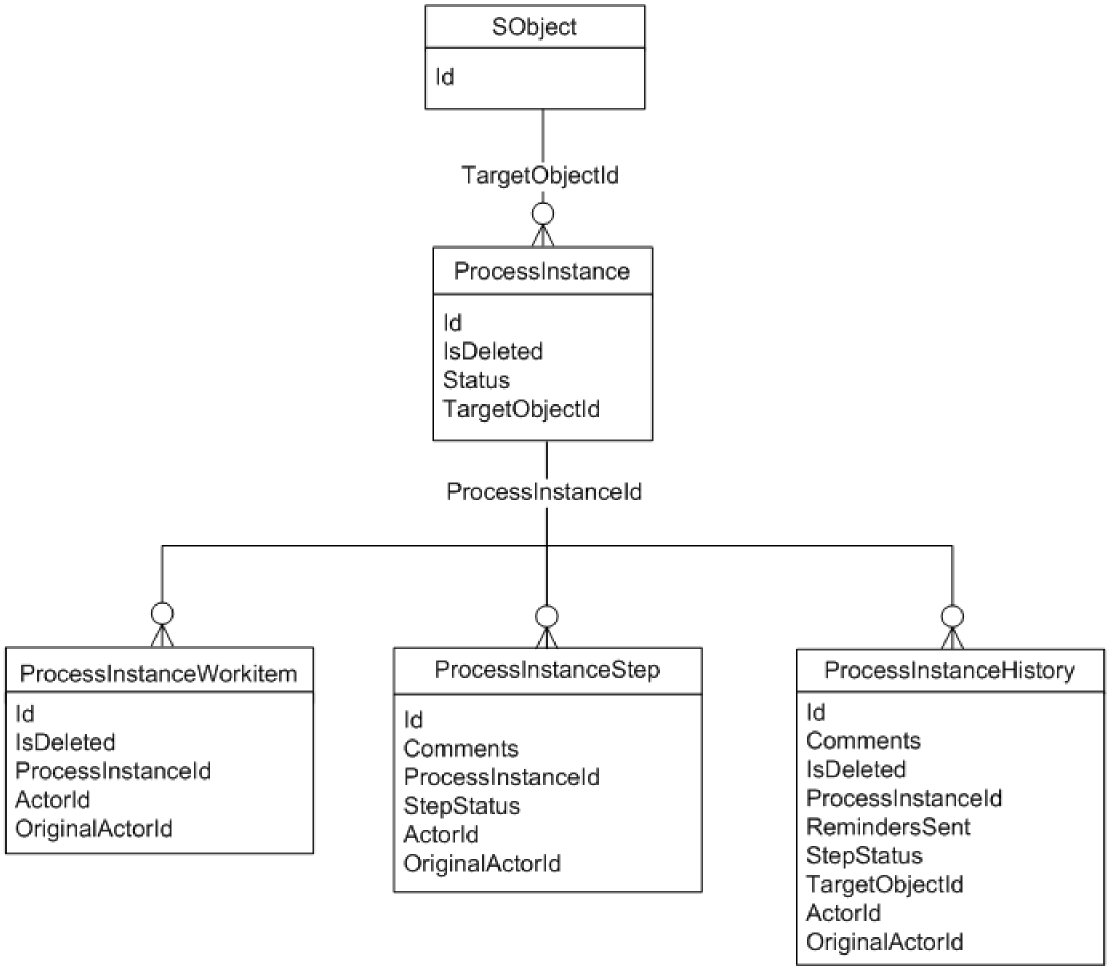
Territory Management



See Also:

- *Standard Objects*
- *Data Model*

Process Objects



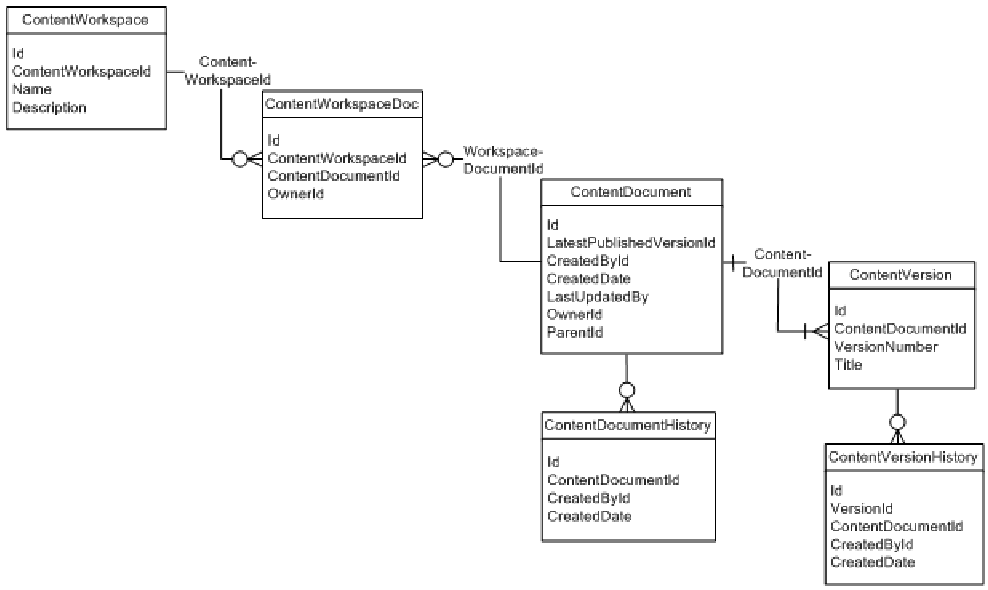
See Also:

Standard Objects

Data Model

Core Data Types Used in API Calls

Content Objects



See Also:

Standard Objects

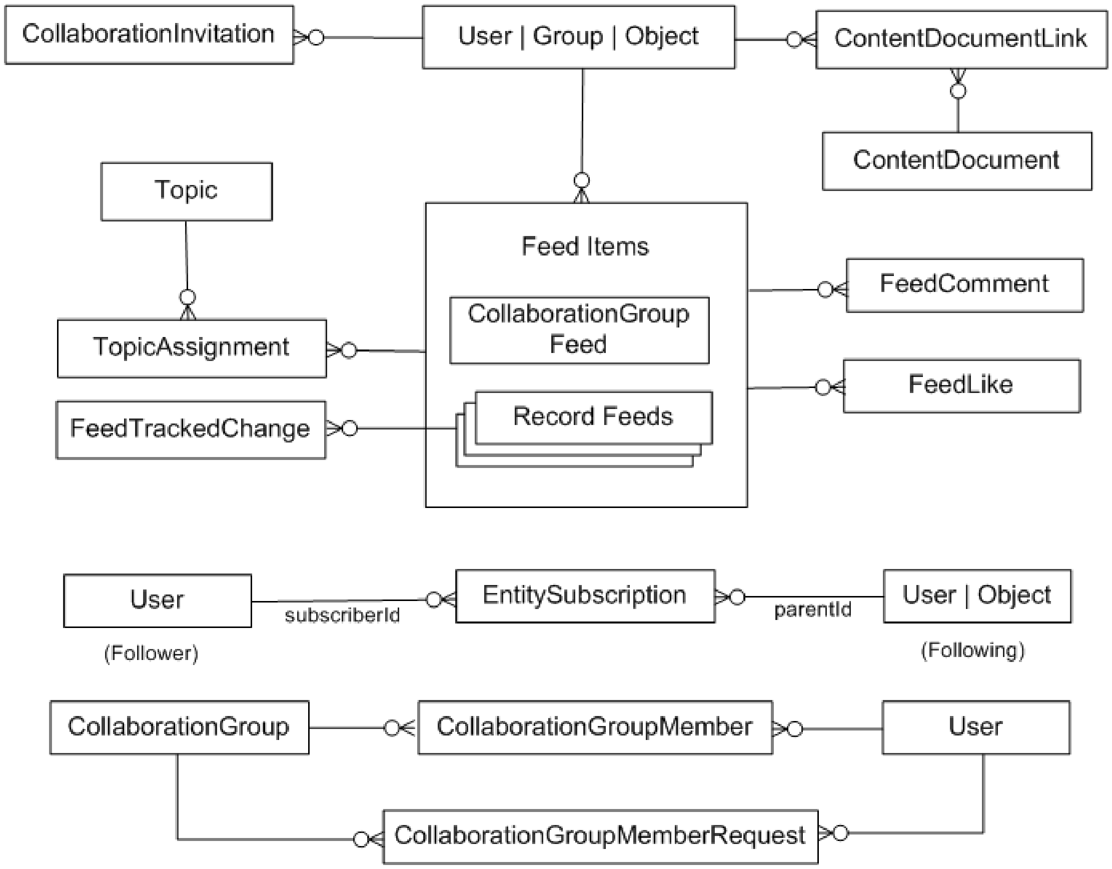
Data Model

Chatter Objects

Diagram showing the relationships between the Chatter objects

The following diagram shows the relationships between the major Chatter objects.

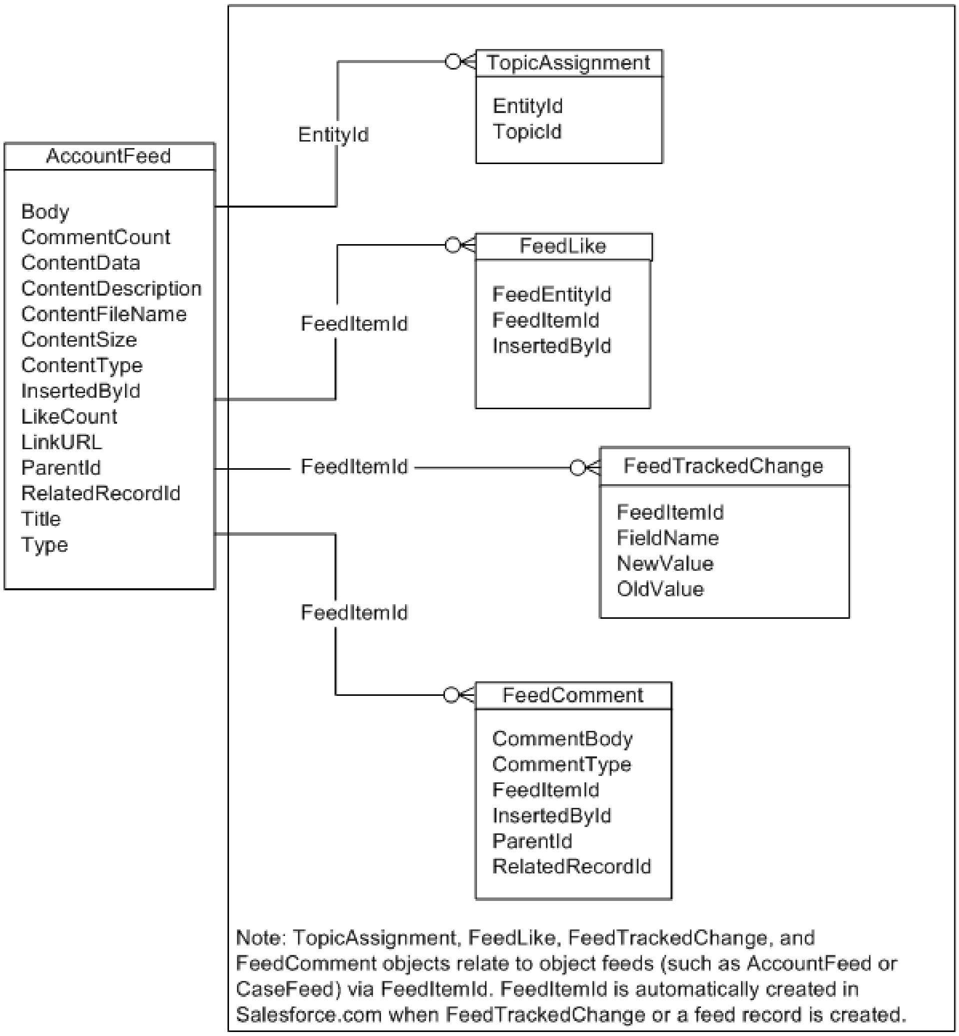
- A feed item is an entry in the feed, such as a change to a record that's being followed, an updated post, or a user status change.
- All feed items have a `ParentId`, which is either:
  - ◊ the record that is changed
  - ◊ the user who received the post



See Also:

Standard Objects
Data Model

Chatter Feed Objects



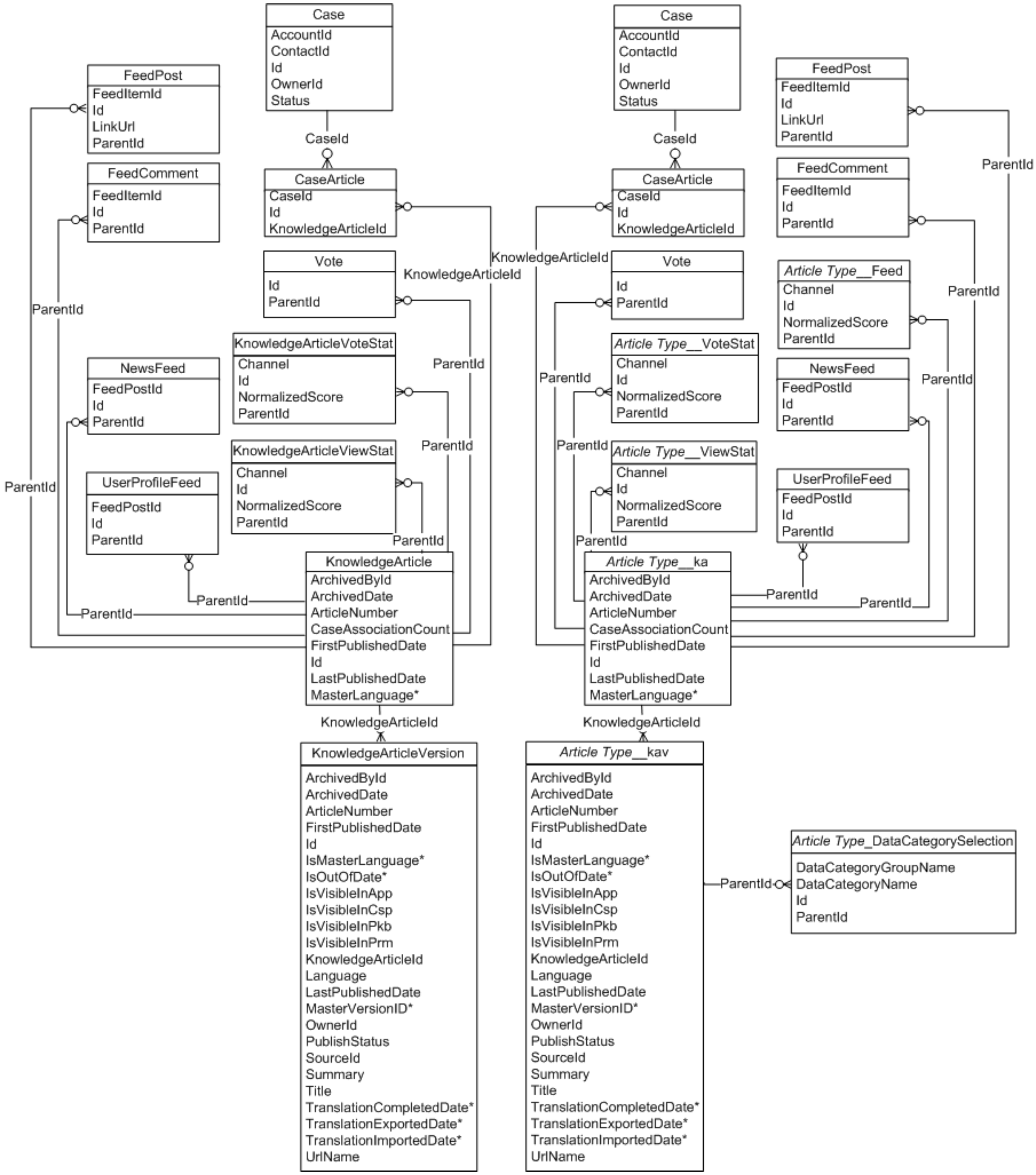
Standard Objects with Feeds

The following standard objects can have Chatter feeds: Account, Asset, Case, CollaborationGroup, Contact, ContentDocument, Contract, Dashboard, DashboardComponent, Entitlement, Event, KnowledgeArticle, Lead, Opportunity, Product2, Quote, Report, ServiceContract, Site, Solution, Task, Topic, and User.

See Also:

Standard Objects
Data Model

Salesforce Knowledge Objects



See Also:

[Standard Objects]()
[Data Model]()

Chapter 8

Standard Objects

This section provides a list of standard objects and their standard fields. Some fields may not be listed for some objects.

To see the system fields for each object, see System Fields.

To verify the complete list of fields for an object, you can use a describe call from the API, or inspect with an appropriate tool, for example, inspecting the WSDL or using a schema viewer.

Object	Description
AcceptedEventRelation	Represents invitees with the status `Accepted` for a given event.
Account	An individual account, which is an organization involved with your business (such as customers, competitors, and partners).
AccountContactRole	The role that a given Contact plays on an Account.
AccountFeed	Represents a single feed item on an account record detail page. This object is available in API version 18.0 and later.
AccountHistory	Represents the history of changes to the values in the fields of an account. This object is available in versions 11.0 and later.
AccountOwnerSharingRule	A rule that grants access to an account to users other than the owner.
AccountPartner	A relationship between two Account objects, such as partnerships or subsidiaries.
AccountShare	A sharing entry on an Account.
AccountTag	Associates a word or short phrase with an Account.
AccountTeamMember	A User who is a member of an Account team.
AccountTerritoryAssignmentRule	A rule that assigns accounts to territories.
AccountTerritoryAssignmentRuleItem	A field-specific criteria row for an AccountTerritoryAssignmentRule.
AccountTerritorySharingRule	Rules for sharing an account within a territory.
ActivityHistory	Information about tasks and events related to an object.
AdditionalNumber	An additional phone number for a CallCenter.
AllowedEmailDomain	Represents an allowed email domain for users in your organization. You can define a whitelist to restrict the email domains allowed in a user's `Email` field. This object is available in API version 29.0 and later.
ApexClass	Represents an Apex class.

Object	Description
ApexComponent	A Visualforce custom component that can be used in a Visualforce page alongside standard components such as `<apex:relatedList>` and `<apex:dataTable>`.
ApexLog	A debug log, containing information about a transaction, including information about Apex, Visualforce, workflow and validation rules. This object is available in API version 19.0 and later.
ApexPage	A Visualforce page, containing Visualforce markup, HTML, Javascript, and other Web-enabled code.
ApexTestQueueItem	Represents a single Apex class in the Apex job queue. This object is available in API version 23.0 and later.
ApexTestResult	Represents the result of an Apex test method execution. This object is available in API version 23.0 and later.
ApexTrigger	Represents an Apex trigger.
Approval	An approval request for a Contract.
*`Article Type`*__DataCategorySelection	A data category selection represents a data category that classifies an article. This object can be used to associate an article with data categories from a data category group or to query the category selections for an article. This object is available in API version 19.0 and later.
*`Article Type`*__Feed	Represents a single feed item in the feed displayed on the detail page for an article. This object is available in API version 20.0 and later.
Asset	An asset (such as product previously sold and installed) owned by an Account or Contact.
AssetFeed	Represents a single feed item in the feed displayed on the detail page for an asset record. This object is available in API version 18.0 and later.
AssetTag	Associates a word or short phrase with an Asset.
AssignmentRule	An assignment rule associated with a Case or Lead.
AsyncApexJob	Represents an individual Apex sharing recalculation job, a batch Apex job, or method with the `future` annotation.
AttachedContentDocument	This read-only object contains all `ContentDocument` objects associated with an object.
Attachment	A file that a User has uploaded and attached to a parent object.
AuthSession	The AuthSession object represents an individual user session in your organization.
Bookmark	A link between two opportunities.
BrandTemplate	Letterhead for email templates.
BusinessHours	Specifies the business hours of your support organization. Escalation rules are run only during these hours.
BusinessProcess	A business process.

Object	Description
CallCenter	A single computer-telephony integration (CTI) system instance in an organization.
Campaign	A marketing campaign, such as a direct mail promotion, webinar, or trade show.
CampaignFeed	Represents a single feed item in the feed on a campaign record detail page. This object is available in API version 18.0 and later.
CampaignMember	The association between a Campaign and either a Lead or Contact.
CampaignMemberStatus	A status value associated with a Campaign.
CampaignOwnerSharingRule	Represents the rules for sharing a Campaign with User records other than the owner.
CampaignShare	Represents a list of access levels to a Campaign along with an explanation of the access level. For example, if you have access to a record because you own it, the Access Level value is `Full` and Reason for Access value is `Owner`.
CampaignTag	Associates a word or short phrase with a Campaign.
Case	A customer issue such as a customer's feedback, problem, or question.
CaseArticle	Represents the association between a Case and a KnowledgeArticle. This object is available in API version 20.0 and later.
CaseComment	A comment that provides additional information about the associated Case.
CaseContactRole	The role that a given Contact plays on a Case.
CaseFeed	Represents a single feed item in the feed displayed on the detail page for a case record. This object is available in API version 18.0 and later.
CaseHistory	Historical information about changes that have been made to the associated Case.
CaseMilestone	Represents a milestone (required step in a customer support process) on a Case. This object is available in API version 18.0 and later.
CaseOwnerSharingRule	A rule that grants access to a case to users other than the owner.
CaseShare	A sharing entry on a Case.
CaseSolution	The association between a particular Case and a particular Solution.
CaseStatus	The status of a Case, such as New, On hold, In Process, and so on.
CaseTag	Associates a word or short phrase with a Case
CaseTeamMember	Represents a case team member, who works with a team of other users to help resolve a case.
CaseTeamRole	Represents a case team role. Every case team member has a role on a case, such as "Customer Contact" or "Case Manager."
CaseTeamTemplate	Represents a predefined case team, which is a group of users that helps resolve a case.
CaseTeamTemplateMember	Represents a member on a predefined case team, which is a group of users that helps resolve cases.

Object	Description
CaseTeamTemplateRecord	The CaseTeamTemplateRecord object is a linking object between the Case and CaseTeamTemplate objects. To assign a predefined case team to a case (customer inquiry), create a CaseTeamTemplateRecord record and point the `ParentId` to the case and the `TeamTemplateId` to the predefined case team.
CategoryData	A logical grouping of Solution records.
CategoryNode	A tree of Solution categories.
CategoryNodeLocalization	The translated value of the label for a category.
ChatterActivity	ChatterActivity represents the number of posts and comments made by a user and the number of comments and likes on posts and comments received by the same user. This object is available in API version 23.0 and later.
ChatterAnswersActivity	Represents the reputation of a User in Chatter Answers communities. This object is available in API version 25.0 and later.
ChatterAnswersReputationLevel	Represents a reputation level within a Chatter Answers zone. This object is available in API version 26.0 and later.
ChatterConversation	Represents a private conversation in Chatter, consisting of messages that conversation members have sent or received. This object is available in API version 23.0 and later.
ChatterConversationMember	Represents a member of a private conversation in Chatter. A member has either sent messages to or received messages from other conversation participants. This object is available in API version 23.0 and later.
ChatterMessage	Represents a message sent as part of a private conversation in Chatter. This object is available in API version 23.0 and later.
CollaborationGroup	Represents a Chatter group. This object is available in API version 19.0 and later.
CollaborationGroupFeed	A group feed shows posts and comments about the group. This object is available in API version 19.0 and later.
CollaborationGroupMember	Represents a member of a Chatter group. This object is available in API version 19.0 and later.
CollaborationGroupMemberRequest	Represents a request to join a private Chatter group.
CollaborationInvitation	Represents an invitation to join Chatter, either directly or through a group. This object is available in API version 21.0 and later.
CombinedAttachment	This read-only object contains all notes, attachments, Google Docs, documents uploaded to libraries in Salesforce CRM Content, and files added to Chatter that are associated with a record.
Community (Zone)	Represents a zone within Ideas, Chatter Answers, or Answers. .
Contact	A contact, which is an individual associated with an Account.
ContactFeed	Represents a single feed item in the feed on a contact record detail page. This object is available in API version 18.0 and later.

Object	Description
ContactHistory	Represents the history of changes to the values in the fields of a contact. This object is available in versions 11.0 and later.
ContactOwnerSharingRule	Represents the rules for sharing a contact with User records other than the owner.
ContactShare	Represents a list of access levels to a Contact along with an explanation of the access level. For example, if you have access to a record because you own it, the Access Level value is `Full` and Reason for Access value is `Owner`.
ContactTag	Associates a word or short phrase with a Contact.
ContentDocument	Represents a document that has been uploaded to a library in Salesforce CRM Content.
ContentDocumentFeed	Represents a single feed item associated with ContentDocument.
ContentDocumentHistory	Represents the history of a document in Salesforce CRM Content.
ContentDocumentLink	Represents the link between a Salesforce CRM Content document or Chatter file and where it's shared. A file can be shared with other users, Chatter groups, records, and Salesforce CRM Content libraries.
ContentVersion	Represents a specific version of a document in Salesforce CRM Content.
ContentVersionHistory	Represents the history of a specific version of a document in Salesforce CRM Content.
ContentWorkspace	Represents a public library in Salesforce CRM Content.
ContentWorkspaceDoc	Represents a link between a document and a library in Salesforce CRM Content.
Contract	A contract (a business agreement) associated with an Account.
ContractContactRole	The role that a given Contact plays on a Contract.
ContractFeed	Represents a single feed item in the feed on the contract record detail page. This object is available in API version 18.0 and later.
ContractHistory	Information about changes to a contract.
ContractLineItem	Represents a Product2 in a ServiceContract (customer support agreement). This object is available in API version 18.0 and later.
ContractLineItemHistory	Represents the history of changes to the values in the fields on a ContractLineItem (items in a customer support agreement). This object is available in API version 18.0 and later.
ContractStatus	The status of a Contract, such as Draft, In Approval, Activated, Terminated, or Expired.
ContractTag	Associates a word or short phrase with a Contract.
CronTrigger	Represents a scheduled job, similar to a cron job on UNIX systems. This object is available in API version 17.0 and later.
CronJobDetail	Contains details about the associated scheduled job, such as the job's name and type.

Object	Description
CurrencyType	The currencies used by an organization for which the multicurrency feature is enabled.
CustomBrand	Represents a custom branding and color scheme. This object is available in API version 28.0 and later.
CustomBrandAsset	Represents a branding element in a custom branding scheme. For example, a color, logo image, header image, or footer text. This object is available in API version 28.0 and later.
*Custom Object*__Feed	Represents a single feed item on a custom object detail page. This object is available in API version 18.0 and later.
CustomPermission	Represents a permission that grants access to custom features. This object is available in API version 31.0 and later.
DandBCompany	Represents a D&B Company record, which is associated with an account. This object is available in API version 25.0 and later.
Dashboard	Represents a dashboard, which shows data from custom reports as visual components. Access is read-only. This object is available in API version 20.0 and later.
DashboardComponent	Represents a dashboard component, which can be a chart, metric, table, or gauge on a dashboard. Access is read-only. This object is available in API version 21.0 and later.
DashboardComponentFeed	Represents a single feed item in the feed displayed on a dashboard component. This object is available in API version 21.0 and later.
DashboardFeed	Represents a single feed item in the feed displayed on a dashboard. This object is available in API version 20.0 and later.
DashboardTag	Associates a word or short phrase with a Dashboard. This object is available in API version 20.0 and later.
DatacloudCompany	Represents the fields for Data.com company records. This object is available in API version 30.0 or later.
DatacloudContact	The fields and properties for Data.com contact records. This object is available in API version 30.0 or later.
DatacloudDandBCompany	Represents a set of read-only fields that are used to return D&B company data from Data.com API calls.
DatacloudOwnedEntity	Represents fields in the DatacloudOwnedEntity object. The DatacloudOwnedEntity object tracks user-purchased records. This object is available in API version 30.0 or later.
DatacloudPurchaseUsage	Represents an object used to identify and track Data.com record purchases. This object is available in API version 30.0 or later.
DatacloudSocialHandle	Returns normalized URLs with userids for different social media used by Data.com contacts. The DatacloudSocialHandle object is a child object of the DatacloudContact object.This object is available in API version 30.0 or later.

Object	Description
DatedConversionRate	The effective dated exchange rates used by an organization for which advanced currency management is enabled.
DeclinedEventRelation	Represents invitees with the status `Declined` for a given event.
Division	A logical segment of your organization's data. Available only if the organization has the Division permission enabled.
DivisionLocalization	The translated value of a label for a division.
Document	A file that a user has uploaded. Unlike Attachment objects, Documents are not attached to a parent object.
DocumentAttachmentMap	Maps the relationship between an EmailTemplate and its attachment, which is stored as a Document.
DocumentTag	Associates a word or short phrase with a Document.
EmailMessage	An email message related to Email-to-Case.
EmailServicesAddress	An email service address.
EmailServicesFunction	An email service.
EmailStatus	The status of an email sent via Salesforce.
EmailTemplate	A template for sending email via Salesforce.
Entitlement	Represents the customer support an Account or Contact is eligible to receive. Entitlements may be based on an Asset, Product2, or ServiceContract. This object is available in API version 18.0 and later.
EntitlementContact	Represents a Contact eligible to receive customer support via an Entitlement. This object is available in API version 18.0 and later.
EntitlementFeed	An entitlement feed shows recent changes to an entitlement record for any fields that are tracked in feeds, as well as comments and posts about the record. This object is available in API version 23.0 and later.
EntitlementHistory	Represents the changes to field values on an Entitlement. This object is available in API version 18.0 and later.
EntitlementTemplate	Represents predefined terms of customer support for a product (Product2). This object is available in API version 18.0 and later.
EntityHistory	Removed as of 8.0. Use the history object corresponding to the object you are working with.
EntitySubscription	Represents a subscription for a user following a record or another user. This object is available in versions 18.0 and later.
EnvironmentHubMember	Represents a member organization in the Environment Hub. This object is available in API version 29.0 and later.
Event	A calendar appointment event.
EventFeed	Represents a single feed item in the feed on an Event. This object is available in API version 20.0 and later.

Object	Description
EventRelation	Represents people (a user, a lead, or contacts) or a resource (such as a conference room) invited to an event. This object lets you add or remove invitees from an event and use the API to manage invitees' responses to invitations.
EventTag	Associates a word or short phrase with an Event.
EventWhoRelation	Represents the relationship between an event and a lead or contacts. This derived object is a filtered version of the EventRelation on page 639 object; that is, IsParent is `true` and IsWhat is `false`. It doesn't represent relationships to invitees or to accounts, opportunities, or other objects.
ExternalDataSource	Represents an external data source, such as SharePoint®. This object is available in API version 27 and later.
ExternalDataUserAuth	Stores authentication settings required for a specific user and external data source, such as SharePoint®. This object is available in API version 27 and later.
FeedComment	Represents a comment added to a feed by a user. Represents a comment added to a feed by a user. This object is available in API version 18.0 and later.
FeedItem	FeedItem represents an entry in the feed, such as changes in a record feed, including text posts, link posts, and content posts. This object is available in API version 21.0 and later.
FeedLike	Indicates that a user has liked a feed item. This object is available in API version 21.0 and later.
FeedPollChoice	Shows the choices for a poll posted in the feed. This object is available in API version 29.0 and later.
FeedPollVote	Shows how users voted on a poll posted in the feed. This object is available in API version 29.0 and later.
FeedPost	FeedPost represents the following types of changes in a record feed, such as AccountFeed: text posts, link posts, and content posts. This object is available in API version 18.0 through 21.0. FeedPost is no longer available in later versions. Starting with API version 21.0, use FeedItem to represent text posts, link posts, and content posts in feeds.
FeedTrackedChange	Represents an individual field change or set of field changes. This object is available in API version 18.0 and later.
FieldPermissions	Represents the enabled field permissions for the parent PermissionSet. This object is available in API version 24.0 and later.
FiscalYearSettings	Fiscal year settings.
Folder	A repository for a Document, MailmergeTemplate, email template, or report. Only one type of item can be contained in a particular Folder.
ForecastingAdjustment	This object represents an individual user's adjustment for a subordinate's forecast via a ForecastingItem. Available in API versions 26 and greater.

Object	Description
ForecastingFact	This is a read-only object linking a ForecastingItem with its opportunities, such as opportunities that share the same owner or forecast category and have a closing date within the period of the forecasting item. Available in API versions 26 and greater.
ForecastingItem	This is a read-only object used for individual forecast amounts. Users see amounts based on their perspectives and forecast roles. The amounts users see include one of the following when forecasting in revenue: `AmountWithoutAdjustments`, `AmountWithoutManagerAdjustment`, `ForecastAmount`, `OwnerOnlyAmount`. The amounts users see include one of the following when forecasting in quantity: `QuantityWithoutAdjustments`, `QuantityWithoutManagerAdjustments`, `ForecastQuantity`, `OwnerOnlyQuantity`.
ForecastingQuota	This object represents an individual user's quota for a specified time period.
ForecastShare	Represents a sharing entry of a forecast at a given role and territory.
Group	A set of User records.
GroupMember	A User or Group that is a member of a public group.
HashtagDefinition	HashtagDefinition represents hashtag (#) topics in public Chatter posts and comments. This object is available in API version 26.0 and later.
Holiday	Represents a period of time during which your customer support team is unavailable. Business hours and escalation rules associated with business hours are suspended during any holidays with which they are affiliated.
Idea	Represents an idea on which users are allowed to comment and vote, for example, a suggestion for an enhancement to an existing product or process.
IdeaComment	Represents a comment that a user has submitted in response to an idea.
IdeaTheme	Represents an invitation to community members to submit ideas that are focused on a specific topic. This object is available in API version 26 and later.
KnowledgeableUser	Represents a user identified as knowledgeable about a specific topic, and ranks them relative to other knowledgeable users. This object is available in API version 31.0 and later.
KnowledgeArticle	Provides read-only access to an article and the ability to delete the master article. This object is available in API version 19.0 and later.
KnowledgeArticleVersion	Provides a global view of standard article fields across all article types depending on their version. This object is available in API version 18.0 and later.
KnowledgeArticleVersionHistory	Enables read-only access to the full history of an article.This object is available in API version 25.0 and later.
KnowledgeArticleViewStat	Provides statistics on the number of views for the specified article across all article types. This object is available in API version 20.0 and later.

Object	Description
KnowledgeArticleVoteStat	Provides the weighted rating for the specified article on a scale of 1 to 5 across all article types. This object is available in API version 20.0 and later.
Lead	A lead, which is a prospect or potential Opportunity.
LeadFeed	Represents a single feed item in the feed displayed on the detail page for a lead record. This object is available in API version 18.0 and later.
LeadHistory	Represents the history of changes to the values in the fields of a lead.
LeadOwnerSharingRule	Rules that assign an owner to a lead.
LeadShare	A sharing entry on a Lead.
LeadStatus	The status of a Lead, such as Open, Qualified, or Converted.
LeadTag	Associates a word or short phrase with a Lead.
LimitAllocationPerApp	Represents a connected app quota for an API limit. This object is available in API version 30.0 and later.
LineitemOverride	Customizable forecast data for an opportunity line item.
LoginHistory	Represents the login history for all successful and failed login attempts for organizations and enabled portals.This object is available in API version 21.0 and later.
MailmergeTemplate	A mail merge template (a Microsoft Word document) used for performing mail merges for your organization.
MilestoneType	Represents a milestone (required step in a customer support process). This object is available in API version 18.0 and later.
Name	Non-queryable object that provides information about foreign key traversals when the foreign key has more than one parent.
Network	Represents a community, which is a customizable public or private space where employees, end-customers, and partners can collaborate on best practices and business processes. Communities give you the opportunity to share information, records, and files with coworkers and related external stakeholders all in one place. This object is available in API version 26.0 and later.
NetworkActivityAudit	Represents an audit trail of moderation actions in Communities. This object is available in API version 30.0 and later.
NetworkMember	Represents a member of a community. Members can be either users in your company or external users with portal profiles. This object is available in API version 26.0 and later.
NetworkMemberGroup	Represents a group of members in a community. Members can be either users in your internal organization or external users assigned portal profiles. An administrator adds members to a community by adding a profile or a permission set, and any user with the profile or permission set becomes a member of the community. This object is available in API version 26.0 and later. Prior to API version 27.0, this object was called NetworkProfile.

Object	Description
NetworkModeration	Represents a flag on an item in a community. This object is available in API version 30.0 and later.
NewsFeed	Represents a single feed item on a user's home tab. A Chatter feed shows recent changes to records that the user is following. NewsFeed is available in API version 18.0 through API version 26.0. In API version 27.0 and later, NewsFeed is no longer available in the SOAP API.
Note	A note, which is text associated with an Attachment, Contact, Contract, Opportunity, or custom object.
NoteAndAttachment	Information about the notes and attachments for an object.
NoteTag	Associates a word or short phrase with a Note.
ObjectPermissions	Represents the enabled object permissions for the parent PermissionSet. This object is available in API version 24.0 and later.
ObjectTerritory2AssignmentRule	Represents a territory assignment rule that's associated with an object, such as Account. ObjectTerritory2AssignmentRuleItem can only be created or deleted if the BooleanFilter field on its corresponding ObjectTerritory2AssignmentRule is `null`. Available only if Territory Management 2.0 has been enabled for your organization.
ObjectTerritory2AssignmentRuleItem	A single row of selection criteria for an ObjectTerritory2AssignmentRule object. ObjectTerritory2AssignmentRuleItem can only be created or deleted if the `BooleanFilter` field on its corresponding ObjectTerritory2AssignmentRule object is a `null` value. Available only if Territory Management 2.0 has been enabled for your organization.
ObjectTerritory2Association	Represents an association (by assignment) between a territory and an object record, such as an account. Available only if Territory Management 2.0 has been enabled for your organization.
OpenActivity	Information about the open tasks and events for an object.
Opportunity	An opportunity, which is a sale or pending deal.
OpportunityCompetitor	A competitor on an Opportunity.
OpportunityContactRole	The role that a Contact plays on an Opportunity.
OpportunityFeed	Represents a single feed item in the feed displayed on the detail page for an opportunity record. This object is available in API version 18.0 and later.
OpportunityFieldHistory	The history of changes to the field values of an Opportunity. This object is available in versions 13.0 and later.
OpportunityHistory	The stage history of an Opportunity.
OpportunityLineItem	An opportunity line item, which is a member of the list of Product2 records associated with an Opportunity, along with other information about those products on that opportunity.
OpportunityLineItemSchedule	Information about the quantity, revenue distribution, and delivery dates for a particular OpportunityLineItem.
OpportunityOverride	Customizable forecast data for an opportunity.

Object	Description
OpportunityOwnerSharingRule	A rule that grants access to an opportunity to users other than the owner.
OpportunityPartner	A partner relationship between an Account and an Opportunity.
OpportunityShare	A sharing entry on an Opportunity.
OpportunitySplit	A related list for an Opportunity, which lets you divide sales credit across opportunity team members.
OpportunitySplitType	A customized label for one of the two default split types: revenue splits, which must total 100%, and overlay splits, which can total any percentage.
OpportunityStage	The stage of an Opportunity in the sales pipeline, such as New Lead, Negotiating, Pending, Closed, and so on.
OpportunityTag	Associates a word or short phrase with an Opportunity.
OpportunityTeamMember	An individual User on the opportunity team of a particular Opportunity.
Organization	A business, company, or other organization.
OrgWideEmailAddress	Represents an organization-wide email address for user profiles.
OwnedContentDocument	Represents a file owned by a user. This object is available in API version 30.0 and later.
PackageLicense	Represents a license for an installed managed package.
Partner	The association between two particular accounts or between a particular Opportunity and an Account.
PartnerNetworkConnection	A connection in Salesforce to Salesforce.
PartnerNetworkRecordConnection	A record that is shared with a connection using Salesforce to Salesforce.
PartnerRole	A role for an account Partner, such as a consultant or supplier.
Period	A fiscal period.
PermissionSet	A set of permissions that's used to grant additional permissions to one or more users without changing their profile. This object is available in API version 22.0 and later.
PermissionSetAssignment	Represents the association between a User and a PermissionSet. This object is available in API version 22.0 and later.
Pricebook2	A price book that contains the list of products (Product2 records) that your organization sells.
Pricebook2History	Represents historical information about changes that have been made to the standard fields of the associated Pricebook2, or to any custom fields with history tracking enabled. This object is available in API version 31.0 and later.
PricebookEntry	A product entry (an association between a Pricebook2 and Product2) in a price book.
ProcessDefinition	Represents the definition of a single approval process.

Object	Description
ProcessInstance	Represents an instance of a single, end-to-end approval process. Use this and the node, step, and workitem process instance objects to create approval history reports.
ProcessInstanceHistory	This read-only object shows all steps and pending approval requests associated with an approval process (ProcessInstance).
ProcessInstanceNode	Represents a step in an instance of an approval process. Compare to ProcessNode, which describes the step in a process definition. Use this object to retrieve approval history.
ProcessInstanceStep	Represents one work item in an approval process (ProcessInstance).
ProcessInstanceWorkitem	Represents a user's pending approval request.
ProcessNode	Describes a step in a process definition. Compare to ProcessInstanceNode, which describes the step in a running process.
Product2	A product that your organization sells. A product is member of the list of items in a Pricebook2.
Product2Feed	Represents a single feed item in the feed displayed on the detail page for a product record. This object is available in API version 18.0 and later.
ProductEntitlementTemplate	Represents predefined terms of customer support (Entitlement) that users can add to products (Product2).
Profile	A profile, which defines a set of user permissions for performing different operations, such as querying, adding, updating, or deleting information.
ProfileSkill	Represents a profile skill, which describes a user's professional knowledge. This is a global record for the organization, and users are associated through the ProfileSkillUser object.
ProfileSkillEndorsement	Represents a detail relationship of ProfileSkillUser. An endorsement of a profile skill shows approval and support of another user's publicly declared skill.
ProfileSkillEndorsementHistory	Represents the history of changes to the fields of a ProfileSkillEndorsement.
ProfileSkillFeed	Represents a single feed item in the feed displayed on the detail page for a profile skill.
ProfileSkillHistory	Represents the history of changes to the fields of a ProfileSkill.
ProfileSkillShare	Represents a sharing entry on a ProfileSkill.
ProfileSkillUser	Represents a detail relationship of User. The object connects profile skills with users.
ProfileSkillUserHistory	Represents the history of changes to the fields of a ProfileSkillUser.
PushTopic	Represents a query that is the basis for notifying listeners of changes to records in an organization.
QuantityForecast	A quantity-based customizable forecast.
QuantityForecastHistory	Historical information about a quantity-based customizable forecast.
Question	Represents a question in a community that users can view and reply to.

Object	Description
QuestionDataCategorySelection	A data category selection represents a data category that classifies a question. This object can be used to associate a question with a data category from a data category group or to query the categorization for a question. It is available in API version 19.0 and later.
QuestionReportAbuse	Represents a user-reported abuse on a Question in a Chatter Answers community. This object is available in API version 24.0 and later.
QuestionSubscription	Represents a subscription for a user following a Question. This object is available in API version 24.0 and later.
QueueSobject	Associates sObject records with a specified queue.
Quote	Represents a quote, which is a record showing proposed prices for products and services. Quotes can be created from and synced with opportunities, and emailed as PDFs to customer. Available in API version 18.0 and later.
QuoteDocument	Represents a Quote in document format. Available in API version 18.0 and later.
QuoteLineItem	Represents a quote line item, which is a member of the list of Product2 products associated with a Quote, along with other information about those line items on that quote. Available in API version 18.0 and later.
RecentlyViewed	Represents a record that the current user has recently viewed or referenced (by viewing a related record).
RecordType	A record type.
RecordTypeLocalization	The translated value of a label for a record type.
Reply	Represents a reply that a user has submitted to a question in an answers community.
ReplyReportAbuse	Represents a user-reported abuse on a Reply in a Chatter Answers community. This object is available in API version 24.0 and later.
Report	Represents a report, a set of data that meets certain criteria, displayed in an organized way. Access is read-only. This object is available in API version 20.0 and later.
ReportFeed	Represents a single feed item in the feed displayed on a report. This object is available in API version 20.0 and later.
ReportTag	Associates a word or short phrase with a Report. This object is available in API version 20.0 and later.
RevenueForecast	A revenue-based customizable forecast.
RevenueForecastHistory	Historical information about a revenue-based customizable forecast.
RuleTerritory2Association	Represents a record-assignment rule and its association to an object, such as Account. Available only if Territory Management 2.0 has been enabled for your organization.
Scontrol	A custom s-control, which is custom content that is hosted by the system but executed by the client application.
ScontrolLocalization	The translated value of the field label for an s-control.

Object	Description
SelfServiceUser	A Contact who has been enabled to use your organization's Self-Service portal, where he or she can obtain online support.
ServiceContract	Represents a customer support contract (business agreement). This object is available in API version 18.0 and later.
ServiceContractFeed	Represents a single feed item in the feed displayed on the detail page for a service contract record. A service contract feed shows recent changes to a service contract for any fields that are tracked in feeds, as well as comments and posts about the record.This object is available in API version 23.0 and later.
ServiceContractHistory	Represents the history of changes to the values in the fields on a ServiceContract (customer support agreement). This object is available in API version 18.0 and later.
ServiceContractOwnerSharingRule	Represents the rules for sharing a ServiceContract (customer service agreement) with users other than the owner. This object is available in API version 18.0 and later.
ServiceContractShare	Represents a sharing entry on a ServiceContract (customer support agreement). This object is available in API version 18.0 and later.
SetupEntityAccess	Represents the enabled setup entity access settings (such as for Apex classes) for the parent PermissionSet. This object is available in API version 25.0 and later.
SignupRequest	Represents a request for a new Trialforce signup. This object is available in API version 27.0 and later.
Site	Represents a public website that is integrated with an Organization. This object is available in API version 18.0 and later. To access this object, Force.com Sites or Site.com must be enabled for your organization.
SiteHistory	Represents the history of changes to the values in the fields of a site.This object is available in API version 18.0 and later. To access this object, Force.com Sites or Site.com must be enabled for your organization.
SlaProcess	Represents an entitlement process associated with an Entitlement. This object is available in API version 19.0 and later.
Solution	A detailed description of a customer issue and the resolution of that issue.
SolutionFeed	Represents a single feed item in the feed displayed on the detail page for a solution record. This object is available in API version 18.0 and later.
SolutionHistory	History of changes to a solution.
SolutionStatus	The status of a Solution, such as Draft, Reviewed, and so on.
SolutionTag	Associates a word or short phrase with a Solution.
StaticResource	A static resource that can be used in Visualforce markup.
StreamingChannel	Represents a channel that is the basis for notifying listeners of generic Streaming API events. This is available from API version 29.0 or later.
TagDefinition	Defines the attributes of child Tag objects.

Object	Description
Task	An activity or to-do item to perform or that has been performed.
TaskFeed	Represents a single feed item in the feed on an Task. This object is available in API version 20.0 and later.
TaskPriority	The priority (importance) of a Task, such as High, Normal, or Low.
TaskRelation	Represents the relationship between a task and a lead, contacts, and other objects related to the task.
TaskStatus	The status of a Task, such as Not Started, Completed, or Closed.
TaskTag	Associates a word or short phrase with a Task.
TaskWhoRelation	TaskWhoRelation represents the relationship between a task and a lead or contacts. This derived object is a filtered version of the TaskRelation on page 1241 object; that is, IsParent is `true` and IsWhat is `false`. It doesn't represent relationships to accounts, opportunities, or other objects.
Territory	A territory to which users and accounts are assigned.
Territory2	Represents a sales territory. Available only if Territory Management 2.0 has been enabled for your organization.
Territory2Model	Represents a territory model. Available only if Territory Management 2.0 has been enabled for your organization.
Territory2ModelHistory	Represents the history of changes to the values in the fields on a territory model. Available only if Territory Management 2.0 has been enabled for your organization.
Territory2Type	Represents a category for territories (Territory2). Every Territory2 must have a Territory2Type. Available only if Territory Management 2.0 has been enabled for your organization.
Topic	Represents a topic on a Chatter post or record. This object is available in API version 28.0 and later.
TopicAssignment	Represents the assignment of a topic to a specific feed item or record. This object is available in API version 28.0 and later.
TopicFeed	Represents a single feed tracked change for a topic. This object is available in API version 29.0 and later.
UndecidedEventRelation	Represents invitees with the status `Not Responded` for a given event.
User	A user in your organization.
UserAccountTeamMember	A single User on the default account team of another user.
UserConfigTransferButton	Represents the association between a Live Agent configuration and a live chat button. This association allows users associated with a specific configuration to transfer chats to a button queue.
UserConfigTransferSkill	Represents the association between a Live Agent configuration and a skill. This association allows users associated with a specific configuration to transfer chats to agents who have that skill.

Object	Description
UserFeed	Represents a single feed item in the feed displayed on a Chatter user profile feed. A user profile feed shows changes to a user record for fields that are tracked in feeds, as well as posts and comments about the record. This object is available in API version 18.0 and later.
UserLicense	A user license in your organization.
UserLogin	Represents the settings that affect a user's ability to log into an organization. This object is available in API version 29.0 and later.
UserMembershipSharingRule	Represents the rules for sharing user records from a source group to a target group. A user record contains details about a user. Users who are members of the source group can be shared with members of the target group.
UserPackageLicense	Represents a license for an installed managed package, assigned to a specific user.
UserPreference	A functional preference for a user in your organization.
UserProfile	Represents a Chatter user profile.
UserProfileFeed	Represents a user profile feed, which tracks all actions by a user on records that can be tracked in a feed. This feed is displayed on the user profile page. UserProfileFeed is available in API version 18.0 through API 26.0. In API version 27.0 and later, UserProfileFeed is no longer available in the SOAP API.
UserRecordAccess	Represents a user's access to a set of records. This object is available in API version 24.0 and later.
UserRole	A role in your organization.
UserShare	Represents a sharing entry on a user record. This object is available in API version 26.0 and later.
UserTeamMember	A single User on the default opportunity team of another user.
UserTerritory	A single user who has been assigned to a territory.
UserTerritory2Association	Represents an association (by assignment) between a territory and a user record. Available only if Territory Management 2.0 has been enabled for your organization.
Vote	Represents a vote that a user has made on an Idea or a Reply.
WebLink	A Web link to an URL or Scontrol.
WebLinkLocalization	The translated value of the field label for a custom link to a URL or s-control.
WorkAccess	Used to grant or restrict user access to give badge definitions. Each badge definition record must have one WorkAccess record.
WorkAccessShare	WorkAccessShare is used to control Givers of WorkBadgeDefinition records.
WorkBadge	A WorkBadge record is created for each recipient of a WorkBadgeDefinition. Contains information about who the badge was given to and which badge was given.

Object	Description
WorkBadgeDefinition	The attributes of a badge including the badge name, description, and image. Each WorkBadge record must have a lookup to a WorkBadgeDefinition as badge attributes, such as badge name, badge description, and badge image, are derived from the WorkBadgeDefinition object.
WorkBadgeDefinitionHistory	Represents the history of changes to the values in the fields of a WorkBadgeDefinition object.
WorkBadgeDefinitionShare	Represents a sharing entry on a WorkBadgeDefinition object.
WorkCoaching	Represents a single coaching relationship between two users. One of the users is defined as the coach and the other is defined as a coachee. WorkCoaching is feed-enabled so there is a private feed available to the coach and coachee.
WorkCoachingFeed	Represents a single feed item in the feed on the detail page for a coaching record. A coaching feed shows recent changes to a lead record for any fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to coaching in Salesforce.
WorkCoachingHistory	Represents the history of changes to the values in the fields of a WorkCoaching object.
WorkCoachingShare	Represents a sharing entry on a WorkCoaching object.
WorkFeedback	Answer to a specific question that the person was asked via a request for feedback. Also used to store offered feedback without linking it to a particular question.
WorkFeedbackQuestion	Represents a question within a set of questions of either a free form text type, or a multiple choice.
WorkFeedbackQuestionHistory	Represents the history of changes to the values in the fields of a WorkFeedbackQuestion.
WorkFeedbackQuestionSet	Represents a set of questions that are being asked. The question set is used to link all the individual requests where different recipients were asked the same set of questions on the same subject.
WorkFeedbackQuestionSetShare	Represents a sharing entry on a WorkFeedbackQuestionSet.
WorkFeedbackQuestionShare	Represents a sharing entry on a WorkFeedbackQuestion.
WorkFeedbackRequest	In the feedback application, WorkFeedbackRequest represents a single feedback request on a subject or topic (question) to a single recipient. In the case of offered feedback, WorkFeedbackRequest represents feedback that is offered about a subject. In the performance application, WorkFeedbackRequest represents a request for feedback on a set of questions from a question set, on a subject—for the recipient to complete and submit.
WorkFeedbackRequestHistory	Represents the history of changes to the values in the fields of a WorkFeedbackRequest.
WorkFeedbackRequestShare	Represents a sharing entry on a WorkFeedbackRequest.
WorkFeedbackShare	Represents a sharing entry on a WorkFeedback object.

Object	Description
WorkGoal	The WorkGoal object represents the components of a goal such as its description and associated metrics.
WorkGoalCollaborator	The WorkGoalCollaborator object represents collaborators on a WorkGoal object. Note that this doesn't include WorkGoal followers, which is handled by Chatter Feed Follow functionality.
WorkGoalCollaboratorHistory	This read-only object contains historical information about changes that were made to the WorkGoalCollaborator object.
WorkGoalFeed	Represents a single feed item in the feed displayed on the goal page for a Goal record. The goal feed shows changes to a goal for fields that are tracked in feeds, posts, and comments about the goal, as well as updates on metrics.
WorkGoalHistory	This read-only object contains historical information about changes that have been made to the WorkGoal object.
WorkGoalLink	Represents the relationship between two goals (many to many relationship).
WorkGoalShare	Represents a sharing entry on a WorkGoal object.
WorkPerformanceCycle	WorkPerformanceCycle represents feedback that is gathered for the purpose of assessing the performance of a specific set of employees.
WorkPerformanceCycleFeed	Represents a single feed item in the feed that is displayed on a Work.com Performance detail page.
WorkPerformanceCycleHistory	Represents the history of changes to the values in the fields of a WorkPerformanceCycle.
WorkPerformanceCycleShare	Represents a sharing entry on a WorkPerformanceCycle object.
WorkReward	Used to store reward codes tied to a Reward Fund. Reward Funds must have at least one WorkReward record.
WorkRewardFund	Represents a Reward Fund and describes the Reward Fund attributes.
WorkRewardFundHistory	Represents the history of changes to the values in the fields of a WorkRewardFund object.
WorkRewardFundShare	Share records for WorkRewardFund.
WorkRewardFundType	Represents the type of WorkRewardFund object.
WorkRewardFundTypeHistory	Represents the history of changes to the values in the fields of a WorkRewardFundType object.
WorkRewardFundTypeShare	Represents a sharing entry on a WorkRewardFundType.
WorkRewardHistory	Represents the history of changes to the fields of a WorkReward.
WorkRewardShare	Share records for WorkReward object.
WorkThanks	Represents the source and message of a thanks post.
WorkThanksShare	Share records for WorkThanks object.

Note: The Product and Pricebook objects are no longer available and have been removed. Requests containing them are refused, and responses do not contain them. Use Product2 and Pricebook2 instead.

Note: Force.com includes the following standard objects: AcceptedEventRelation, Account, AccountContactRole, AccountFeed, AccountHistory, AccountPartner, AccountShare, ActivityHistory, AdditionalNumber, AggregateResult, Announcement, ApexClass, ApexComponent, ApexPage, ApexTestQueueItem, ApexTestResult, ApexTrigger, AppMenuItem, AssignmentRule, AsyncApexJob, AttachedContentDocument, Attachment, AuthSession, BrandTemplate, BusinessHours, BusinessProcess, CallCenter, CategoryNode, ChatterActivity, ClientBrowser, CollaborationGroup, CollaborationGroupFeed, CollaborationGroupMember, CollaborationGroupMemberRequest, CollaborationInvitation, CombinedAttachment, Community, Contact, ContactFeed, ContactHistory, ContactShare, ContentDocument, ContentDocumentFeed, ContentDocumentHistory, ContentDocumentLink, ContentVersion, ContentVersionHistory, CronJobDetail, CronTrigger, Dashboard, DashboardComponent, DashboardComponentFeed, DashboardFeed, DeclinedEventRelation, Document, DocumentAttachmentMap, Domain, DomainSite, EmailServicesAddress, EmailServicesFunction, EmailStatus, EmailTemplate, EntitySubscription, Event, EventFeed, EventRelation, FeedComment, FeedItem, FeedLike, FeedTrackedChange, FieldPermissions, FiscalYearSettings, Folder, ForecastShare, Group, GroupMember, HashtagDefinition, Holiday, Idea, IdeaComment, LoginIp, MailmergeTemplate, MobileDeviceRegistrar, Name, Note, NoteAndAttachment, ObjectPermissions, OpenActivity, OrgWideEmailAddress, OwnedContentDocument, Period, PermissionSet, PermissionSetAssignment, PermissionSetLicense, PermissionSetLicenseAssign, ProcessDefinition, ProcessInstance, ProcessInstanceHistory, ProcessInstanceNode, ProcessInstanceStep, ProcessInstanceWorkitem, ProcessNode, Profile, PushTopic, QueueSobject, RecentlyViewed, RecordType, Report, ReportFeed, Scontrol, SetupEntityAccess, Site, SiteFeed, SiteHistory, StaticResource, Task, TaskFeed, TaskPriority, TaskStatus, Topic, TopicAssignment, TopicFeed, UndecidedEventRelation, User, UserFeed, UserLicense, UserPreference, UserProfile, UserRecordAccess, UserRole, UserShare, Vote.

AcceptedEventRelation

Represents invitees with the status `Accepted` for a given event.

This object is available in API versions 29.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`EventId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the event.
`RelationId`	**Type** reference

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** Indicates the ID of the invitee.
`RespondedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Indicates the most recent date and time when the invitee accepted an invitation to the event.
`Response`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Indicates the content of the response field. Label is `Comment`.
`Type`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Indicates whether the invitee is a user, lead or contact, or resource.

Usage

Query invitees who have accepted an invitation to an event

```
SELECT eventId, type, response FROM AcceptedEventRelation WHERE eventid='00UTD000000ZH5LA'
```

See Also:

DeclinedEventRelation
UndecidedEventRelation

Account

Represents an individual account, which is an organization or person involved with your business (such as customers, competitors, and partners).

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `merge()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can access their own accounts and any account shared with them.

Fields

Field Name	Details
`AccountNumber`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Account number assigned to this account (not the unique, system-generated ID assigned during creation). Maximum size is 40 characters.
`AccountSource`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The source of the account record. For example, `Advertisement`, `Data.com`, or `Trade Show`. The source is selected from a picklist of available values, which are set by an administrator. Each picklist value can have up to 40 characters.
`AnnualRevenue`	**Type** currency **Properties** Create, Filter, Nillable, Sort, Update **Description** Estimated annual revenue of the account.
`BillingAddress` (beta)	**Type** address

Field Name	Details
	Properties Filter, Nillable **Description** The compound form of the billing address. Read-only. See Address Compound Fields for details on compound address fields.
`BillingCity`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details for the billing address of this account. Maximum size is 40 characters.
`BillingCountry`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details for the billing address of this account. Maximum size is 80 characters.
`BillingCountryCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO country code for the account's billing address.
`BillingLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `BillingLongitude` to specify the precise geolocation of a billing address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`BillingLongitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update

Field Name	Details
	Description Used with `BillingLatitude` to specify the precise geolocation of a billing address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`BillingPostalCode`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details for the billing address of this account. Maximum size is 20 characters.
`BillingState`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details for the billing address of this account. Maximum size is 80 characters.
`BillingStateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO state code for the account's billing address.
`BillingStreet`	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Street address for the billing address of this account.
`CleanStatus`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update

Field Name	Details
	Description Indicates the record's clean status as compared with Data.com. Values are: `In Sync`, `Different`, `Reviewed`, `Not Found`, `Inactive`, `Not Compared`, `Select Match`, or `Skipped`.
`ConnectionReceivedId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that shared this record with your organization. This field is only available if you have enabled Salesforce to Salesforce.
`ConnectionSentId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that you shared this record with. This field is only available if you have enabled Salesforce to Salesforce. Beginning with API version 15.0, the `ConnectionSentId` field is no longer supported. The `ConnectionSentId` field is still visible, but the value is null. You can use the new PartnerNetworkRecordConnection object to forward records to connections.
`Description`	**Type** textarea **Properties** Create, Nillable, Update **Description** Text description of the account. Limited to 32,000 KB.
`DunsNumber`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The Data Universal Numbering System (D-U-N-S) number is a unique, nine-digit number assigned to every business location in the Dun & Bradstreet database that has a unique, separate, and distinct operation. D-U-N-S numbers are used by industries and organizations around the world as a global standard for business identification and tracking. Maximum size is 9 characters.

Field Name	Details
	Note: This field is only available to organizations that use Data.com Prospector or Data.com Clean.
`Fax`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Fax number for the account.
`Industry`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** An industry associated with this account. Maximum size is 40 characters.
`IsCustomerPortal`	**Type** boolean **Properties** Defaulted on create, Filter, Update **Description** Indicates whether the account has at least one contact enabled to use the organization's Customer Portal (`true`) or not (`false`). This field is available if Customer Portal is enabled OR Communities is enabled and you have Customer Portal licenses. If you change this field's value from `true` to `false`, you can disable up to 100 Customer Portal users associated with the account and permanently delete all of the account's Customer Portal roles and groups. You can't restore deleted Customer Portal roles and groups. This field can be updated in API version 16.0 and later. **Tip:** We recommend that you only update up to 50 contacts simultaneously when changing the accounts on contacts enabled for a Customer Portal or partner portal. We also recommend that you make this update during times outside of your organization's business hours.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter

Field Name	Details
	Description Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsPartner`	**Type** boolean **Properties** Defaulted on create, Filter, Update **Description** Indicates whether the account has at least one contact enabled to use the organization's partner portal (`true`) or not (`false`). This field is available if partner relationship management (partner portal) is enabled OR Communities is enabled and you have partner portal licenses. If you change this field's value from `true` to `false`, you can disable up to 15 partner portal users associated with the account and permanently delete all of the account's partner portal roles and groups. You can't restore deleted partner portal roles and groups. Disabling a partner portal user in the Salesforce user interface or the API does not change this field's value from `true` to `false`. Even if this field's value is `false`, you can enable a contact on an account as a partner portal user via the API. This field can be updated in API version 16.0 and later. **Tip:** We recommend that you only update up to 50 contacts simultaneously when changing the accounts on contacts enabled for a Customer Portal or partner portal. We also recommend that you make this update during times outside of your organization's business hours.
`IsPersonAccount`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Read only. Label is **Is Person Account**. Indicates whether this account has a record type of Person Account (`true`) or not (`false`).
`Jigsaw`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description References the ID of a company in Data.com. If an account has a value in this field, it means that the account was imported from Data.com. If the field value is `null`, the account was not imported from Data.com. Maximum size is 20 characters. Available in API version 22.0 and later. Label is **Data.com Key**. **Important:** The `Jigsaw` field is exposed in the API to support troubleshooting for import errors and reimporting of corrected data. Do not modify the value in the `Jigsaw` field.
`LastActivityDate`	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Value is one of the following, whichever is the most recent: • Due date of the most recent event logged against the record. • Due date of the most recently closed task associated with the record.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`MasterRecordId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description If this object was deleted as the result of a merge, this field contains the ID of the record that was kept. If this object was deleted for any other reason, or has not been deleted, the value is `null`.
`NaicsCode`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The six-digit North American Industry Classification System (NAICS) code is the standard used by business and government to classify business establishments into industries, according to their economic activity for the purpose of collecting, analyzing, and publishing statistical data related to the U.S. business economy. Maximum size is 8 characters. **Note:** This field is only available to organizations that use Data.com Prospector or Data.com Clean.
`NaicsDesc`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** A brief description of an organization’s line of business, based on its NAICS code. Maximum size is 120 characters. **Note:** This field is only available to organizations that use Data.com Prospector or Data.com Clean.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Label is **Account Name**. Name of the account. Maximum size is 255 characters. If the account has a record type of Person Account: • This value is the concatenation of the `FirstName`, `MiddleName` (beta), `LastName`, and `Suffix` (beta) of the associated person contact. • You can't modify this value.

Field Name	Details
NumberOfEmployees	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Label is **Employees**. Number of employees working at the company represented by this account. Maximum size is eight digits.
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The ID of the user who currently owns this account. Default value is the user logged in to the API to perform the create. If you have set up account teams in your organization, updating this field has different consequences depending on your version of the API: • For API version 12.0 and later, sharing records are kept, as they are for all objects. • For API version before 12.0, sharing records are deleted. • For API version 16.0 and later, users must have the “Transfer Record” permission in order to update (transfer) account ownership using this field.
Ownership	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Ownership type for the account, for example Private, Public, or Subsidiary.
ParentId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the parent object, if any.
Phone	**Type** phone

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** Phone number for this account. Maximum size is 40 characters.
PhotoUrl	**Type** url **Properties** Filter, Group, Nillable, Sort **Description** Path to be combined with the URL of a Salesforce instance (for example, https://na1.salesforce.com/) to generate a URL to request the social network profile image associated with the account. Generated URL returns an HTTP redirect (code 302) to the social network profile image for the account. Blank if Social Accounts and Contacts isn't enabled for the organization or if Social Accounts and Contacts is disabled for the requesting user.
Rating	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The account's prospect rating, for example Hot, Warm, or Cold.
RecordTypeId	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the record type assigned to this object.
Salutation	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Honorific added to the name for use in letters, etc.

Field Name	Details
`ShippingAddress` (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the shipping address. Read-only. See Address Compound Fields for details on compound address fields.
`ShippingCity`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details of the shipping address for this account. City maximum size is 40 characters
`ShippingCountry`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details of the shipping address for this account. Country maximum size is 80 characters.
`ShippingCountryCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO country code for the account's shipping address.
`ShippingLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `ShippingLongitude` to specify the precise geolocation of a shipping address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.

Field Name	Details
`ShippingLongitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `ShippingLatitude` to specify the precise geolocation of an address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`ShippingPostalCode`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details of the shipping address for this account. Postal code maximum size is 20 characters.
`ShippingState`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details of the shipping address for this account. State maximum size is 80 characters.
`ShippingStateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO state code for the account's shipping address.
`ShippingStreet`	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The street address of the shipping address for this account. Maximum of 255 characters.

Field Name	Details
`Sic`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Standard Industrial Classification code of the company's main business categorization, for example, 57340 for Electronics. Maximum of 20 characters.
`SicDesc`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** A brief description of an organization's line of business, based on its SIC code. Maximum length is 80 characters.
`Site`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Name of the account's location, for example `Headquarters` or `London`. Label is **Account Site**. Maximum of 80 characters.
`TickerSymbol`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The stock market symbol for this account. Maximum of 20 characters.
`Tradestyle`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** A name, different from its legal name, that an organization may use for conducting business. Similar to "Doing business as" or "DBA". Maximum length is 255 characters.

Field Name	Details
	Note: This field is only available to organizations that use Data.com Prospector or Data.com Clean.
`Type`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Type of account, for example, Customer, Competitor, or Partner.
`Website`	**Type** url **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The website of this account. Maximum of 255 characters.
`YearStarted`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The date when an organization was legally established. Maximum length is 4 characters. **Note:** This field is only available to organizations that use Data.com Prospector or Data.com Clean.

IsPersonAccount Fields

These fields are the subset of person account fields that are contained in the child person contact record of each person account. If the `IsPersonAccount` field has the value `false`, the following fields have a null value and can't be modified. If `true`, the fields contain the value indicated in the Description column in the following table and can be modified.

Person accounts are not enabled by default.

Field Name	Details
`FirstName`	**Type** string

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** First name of the person for a person account. Maximum size is 40 characters.
`LastName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Last name of the person for a person account. Required if the record type is a person account record type. Maximum size is 80 characters.
`MiddleName` (beta)	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Middle name of the person for a person account. Maximum size is 40 characters. Contact salesforce.com Customer Support to enable this field.
`PersonAssistantName`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** The person account's assistant name. Label is **Assistant**. Maximum size is 40 characters.
`PersonAssistantPhone`	**Type** phone **Properties** Create, Filter, Nillable, Update **Description** The person account's assistant phone. Label is **Asst. Phone**. Maximum size is 40 characters.

Field Name	Details

`PersonBirthDate`	**Type**
	date
	
	Properties
	Create, Filter, Nillable, Update
	
	Description
	The assistant name. Label is **Birthdate**.
	
	Note: The year portion of the `PersonBirthDate` field is ignored in filter criteria, including report filters, list view filters, and SOQL queries. For example, the following SOQL query returns person accounts with birthdays later in the year than today:
	
	`SELECT FirstName, LastName, PersonBirthDate`
	`FROM Account`
	`WHERE Birthdate > TODAY`

`PersonContactId`	**Type** reference **Properties** Filter, Nillable, Update **Description** The ID for the contact associated with this person account. Label is **Contact ID**.
`PersonDepartment`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** The department. Label is **Department**. Maximum size is 80 characters.
`PersonEmail`	**Type** email **Properties** Create, Filter, Nillable, Update **Description** Email address for thisperson account. Label is **Email**.

Field Name	Details
PersonEmailBouncedDate	**Type** dateTime **Properties** Create, Filter, Nillable, Update **Description** If bounce management is activated and an email sent to the person account bounces, the date and time the bounce occurred.
PersonEmailBouncedDate	**Type** string **Properties** Create, Filter, Nillable, Update **Description** If bounce management is activated and an email sent to the person account bounces, the reason the bounce occurred
PersonHasOptedOutOfEmail	**Type** boolean **Properties** Create, Filter, Nillable, Update **Description** Indicates whether the person account has opted out of email (`true`) or not (`false`). Label is **Email Opt Out**.
PersonHomePhone	**Type** phone **Properties** Create, Filter, Nillable, Update **Description** The home phone number for this person account. Label is **Home Phone**.
PersonLastCURequestDate	**Type** dateTime **Properties** Create, Filter, Nillable, Update

Field Name	Details
	Description The last date that this person account was requested. Label is **Last Stay-in-Touch Request Date**.
`PersonLastCUUpdateDate`	**Type** dateTime **Properties** Create, Filter, Nillable, Update **Description** The last date a person account was updated. Label is **Last Stay-in-Touch Save Date**.
`PersonLeadSource`	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** The person account's lead source. Label is **Lead Source**.
`PersonMailingAddress` (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the person account mailing address. Read-only. See Address Compound Fields for details on compound address fields.
`PersonMailingCity`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** Details about the person account's mailing city. Labels are **Mailing City**, **Mailing Country**, **Postal Code**, and **State**. Maximum size for city and country is 40 characters. Maximum size for postal code and state is 20 characters.
`PersonMailingLatitude` (beta)	**Type** double

Field Name	Details
	Properties Create, Filter, Nillable, Sort, Update **Description** Used with `PersonMailingLongitude` to specify the precise geolocation of a person account's mailing address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`PersonMailingLongitude` (beta)	**Type** double **Properties** Create, Filter, , Nillable, Sort, Update **Description** Used with `PersonMailingLatitude` to specify the precise geolocation of a person account's mailing address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`PersonMailingStreet`	**Type** textarea **Properties** Create, Filter, Nillable, Update **Description** The mailing street address for this person account. Label is **Mailing Street**. Maximum size is 255 characters.
`PersonMobilePhone`	**Type** phone **Properties** Create, Filter, Nillable, Update **Description** The mobile phone number for this person account. Label is **Mobile**.
• `PersonOtherCity` • `PersonOtherCountry` • `PersonOtherPostalCode` • `PersonOtherState`	**Type** string **Properties** Create, Filter, Nillable, Update

Field Name	Details
	Description Details about the alternate address for this person account. Labels are **Other City**, **Other Country**, **Other Zip/Postal Code**, and **Other State**.
• `PersonOtherCountryCode` • `PersonOtherStateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO country or state code for the alternate address of the person account.
`PersonOtherLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `PersonOtherLongitude` to specify the precise geolocation of a person account's alternate address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`PersonOtherLongitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `PersonOtherLatitude` to specify the precise geolocation of a person account's alternate address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`PersonOtherPhone`	**Type** phone **Properties** Create, Filter, Nillable, Update **Description** The alternate phone number for this person account. Label is **Other Phone**.
`PersonOtherStreet`	**Type** textarea

Field Name	Details
	Properties Create, Filter, Nillable, Update **Description** The person account's alternate street address. Label is **Other Street**.
`PersonTitle`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** The person account's title. Label is **Title**. Maximum size is 80 characters.
`Suffix` (beta)	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Name suffix of the person for a person account. Maximum size is 40 characters. Contact salesforce.com Customer Support to enable this field.

Note: If you are importing Account data into Salesforce and need to set the value for an audit field, such as `CreatedDate`, contact salesforce.com. Audit fields are automatically updated during API operations unless you request to set these fields yourself.

Usage

Use this object to query and manage accounts in your organization. Client applications can create, update, delete, or query Attachment records associated with an account via the API.

Client applications can also create or update account objects by converting a Lead via the `convertLead()` call.

If the values in the IsPersonAccount Fields are not null, you can't change `IsPersonAccount` to `false`, or an error occurs.

See Also:

AccountShare
AccountTeamMember
Person Account Record Types
AccountHistory

AccountCleanInfo

Stores the metadata Data.com Clean uses to determine an account record's clean status. Helps you automate the cleaning or related processing of account records. AccountCleanInfo includes a number of bit vector fields.

- `CleanedBy` indicates who (a user) or what (a Clean job) cleaned the account record.
- `IsDifferent` indicates whether or not a field on the account record has a value that differs from the corresponding field on the matched Data.com record.
- `IsFlaggedWrong` indicates whether or not a field on the account record has a value that is flagged as wrong to Data.com.
- `IsReviewed` indicates whether or not a field on the account record is in a `Reviewed` state, which means that the value was reviewed by a user but not accepted.

Their individual bits are defined here.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`

Fields

Field Name	Details
`AccountId`	**Type** reference **Properties** Filter, Group, Sort **Description** The unique, system-generated ID assigned when the account record was created.
`AccountSite`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Information about the account's location, such as single location, headquarters, or branch.
`Address` (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the address. Read-only. See Address Compound Fields for details on compound address fields.

Field Name	Details
`AnnualRevenue`	**Type** currency **Properties** Filter, Nillable, Sort **Description** Estimated annual revenue of the account.
`City`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the account.
`CleanedByJob`	**Type** boolean **Properties** Filter **Description** Indicates whether the account record was cleaned by a Data.com Clean job (`true`) or not (`false`).
`CleanedByUser`	**Type** boolean **Properties** Filter **Description** Indicates whether the account record was cleaned by a Salesforce user (`true`) or not (`false`).
`CompanyName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The name of the company.
`CompanyStatusDataDotCom`	**Type** picklist

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** The status of the company per Data.com. Values are: `Company is In Business per Data.com` or `Company is Out of Business per Data.com`.
`Country`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the account.
`DandBCompanyDunsNumber`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The D-U-N-S Number on the D&B Company record (if any) that is linked to the account.
`DataDotComId`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The ID Data.com maintains for the company.
`Description`	**Type** textarea **Properties** Nillable **Description** A description of the account.
`DunsNumber`	**Type** string

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** The Data Universal Numbering System (D-U-N-S) number is a unique, nine-digit number assigned to every business location in the Dun & Bradstreet database that has a unique, separate, and distinct operation. D-U-N-S numbers are used by industries and organizations around the world as a global standard for business identification and tracking.
`DunsRightMatchConfidence`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The account's DUNSRight confidence code.
`DunsRightMatchGrade`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The account's DUNSRight match grade.
`Fax`	**Type** phone **Properties** Filter, Group, Nillable, Sort **Description** The account's fax number.
`Industry`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** The industry the account belongs to.
`IsDifferentAccountSite`	**Type** boolean

Field Name	Details
	Properties Filter **Description** Indicates whether the account's `AccountSite` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentAnnualRevenue`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `AnnualRevenue` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentCity`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `City` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentCompanyName`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `AccountName` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentCountry`	**Type** boolean **Properties** Filter

Field Name	Details
	Description Indicates whether the account's `Country` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentCountryCode`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `Country Code` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentDandBCompanyDunsNumber`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `DandBCompanyID` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentDescription`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `Description` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentDunsNumber`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `DunsNumber` field value is different from the D-U-N-S Number on its matched Data.com record (`true`) or not (`false`).

Field Name	Details
IsDifferentFax	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `Fax` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentIndustry	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `Industry` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentNaicsCode	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `NaicsCode` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentNaicsDescription	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `NaicsDescription` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentNumberOfEmployees	**Type** boolean **Properties** Filter

Field Name	Details
	Description Indicates whether the account's `NumberOf Employees` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentOwnership`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `Ownership` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentPhone`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `Phone` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentPostalCode`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `PostalCode` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentSic`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `Sic` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).

Field Name	Details
IsDifferentSicDescription	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `SicDescription` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentState	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `State` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentStateCode	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `State Code` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentStreet	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `State` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentTickerSymbol	**Type** boolean **Properties** Filter

Field Name	Details
	Description Indicates whether the account's `TickerSymbol` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentTradestyle`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `Tradestyle` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentWebsite`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `Website` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentYearStarted`	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `YearStarted` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsFlaggedWrongAccountSite`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `AccountSite` field value is flagged as wrong to Data.com (`true`) or not (`false`).

Field Name	Details
`IsFlaggedWrongAddress`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Address` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongAnnualRevenue`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `AnnualRevenue` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongCompanyName`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `CompanyName` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongDescription`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Description` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongDunsNumber`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `DunsNumber` field value is flagged as wrong to Data.com (`true`) or not (`false`).

Field Name	Details
IsFlaggedWrongFax	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Fax` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongIndustry	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Industry` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongNaicsCode	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `NaicsCode` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongNaicsDescription	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `NaicsDescription` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongNumberOfEmployees	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `NumberOfEmployees` field value is flagged as wrong to Data.com (`true`) or not (`false`).

Field Name	Details
`IsFlaggedWrongOwnership`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Ownership` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongPhone`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Phone` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongSic`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Sic` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongSicDescription`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `SicDescription` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongTickerSymbol`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `TickerSymbol` field value is flagged as wrong to Data.com (`true`) or not (`false`).

Field Name	Details
IsFlaggedWrongTradestyle	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Tradestyle` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongWebsite	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Website` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongYearStarted	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `YearStarted` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsInactive	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the account has been reported to Data.com as `Inactive` (`true`) or not (`false`).
IsReviewedAccountSite	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `AccountSite` field value is in a `Reviewed` state (`true`) or not (`false`).

Field Name	Details
IsReviewedAddress	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Address` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedAnnualRevenue	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `AnnualRevenue` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedCompanyName	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `CompanyName` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedDandBCompanyDunsNumber	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `DandBCompanyID` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedDescription	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's `Description` field value is in a `Reviewed` state (`true`) or not (`false`).

Field Name	Details
IsReviewedDunsNumber	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account’s `DunsNumber` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedFax	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account’s `Fax` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedIndustry	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account’s `Industry` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedNaicsCode	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account’s `NaicsCode` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedNaicsDescription	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account’s `NaicsDescription` field value is in a `Reviewed` state (`true`) or not (`false`).

Field Name	Details
IsReviewedNumberOfEmployees	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's NumberOfEmployees field value is in a Reviewed state (true) or not (false).
IsReviewedOwnership	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's Ownership field value is in a Reviewed state (true) or not (false).
IsReviewedPhone	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's Phone field value is in a Reviewed state (true) or not (false).
IsReviewedSic	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's Sic field value is in a Reviewed state (true) or not (false).
IsReviewedSicDescription	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account's SicDescription field value is in a Reviewed state (true) or not (false).

Field Name	Details
IsReviewedTickerSymbol	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account’s `TickerSymbol` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedTradestyle	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account’s `Tradestyle` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedWebsite	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account’s `Website` field value is in a `Reviewed` state (`true`) or not (`false`).
IsReviewedYearStarted	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the account’s `YearStarted` field value is in a `Reviewed` state (`true`) or not (`false`).
LastMatchedDate	**Type** dateTime **Properties** Filter, Sort **Description** The date the account record was last matched and linked to a Data.com record.

Field Name	Details
LastStatusChangedById	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of who or what last changed the record's `Clean Status` field value: a Salesforce user or a Clean job.
LastStatusChangedDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date on which the record's `Clean Status` field value was last changed.
`Latitude` (beta)	**Type** double **Properties** Filter, Nillable, Sort **Description** Used with `Longitude` to specify the precise geolocation of a billing address. Data not currently provided.
`Longitude` (beta)	**Type** double **Properties** Filter, Nillable, Sort **Description** Used with `Latitude` to specify the precise geolocation of a billing address. Data not currently provided.
NaicsCode	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The six-digit North American Industry Classification System (NAICS) code is the standard used by business and government to classify business

Field Name	Details
	establishments into industries, according to their economic activity for the purpose of collecting, analyzing, and publishing statistical data related to the U.S. business economy.
`NaicsDescription`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on its NAICS code.
`Name`	**Type** string **Properties** Filter, Group, Sort, Update **Description** Field label is **Account Clean Info Name**. The name of the account. Maximum size is 255 characters.
`NumberOfEmployees`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of employees working at the account.
`Ownership`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Ownership type for the account, for example Private, Public, or Subsidiary.
`Phone`	**Type** phone **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description The phone number for the account.
PostalCode	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the account.
Sic	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Standard Industrial Classification code of the company's main business categorization, for example, 57340 for Electronics.
SicDescription	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on its SIC code.
State	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the account.
Street	**Type** textarea **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the account.

Field Name	Details
TickerSymbol	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The stock market symbol for the account.
Tradestyle	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A name, different from its legal name, that an organization can use for conducting business. Similar to "Doing business as" (DBA).
Website	**Type** url **Properties** Filter, Group, Nillable, Sort **Description** The website of the account.
YearStarted	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The year the company was established or the year when current ownership or management assumed control of the company.

Usage

Create triggers that read AccountCleanInfo fields to help automate the cleaning or related processing of account records. For example:

- Keep account records' status `InSync` if the only difference from matched records is the `Phone` format (for example, *(415) 353-8000* on the account record versus *415 353 8000* on the matched Data.com record).

```
trigger AccountPhoneTrigger on Account (before update) {

   for (Account account: Trigger.new) {
       Account oldAccount = Trigger.oldMap.get(account.ID);
       if (account.CleanStatus == 'Different') {
```

```
            List cleanInfo = [Select Id, IsDifferentPhone, IsReviewedPhone, Phone from
AccountCleanInfo where AccountId = :account.Id];
            if (cleanInfo.size() > 0 && cleanInfo[0].IsDifferentPhone &&
cleanInfo[0].Phone.StartsWith('+')) {
                // if Data.com phone number is marked Different but starts with '+',
ignore this
                // and set the status to "Reviewed"
                AccountCleanInfo cleanInfoToUpdate = new AccountCleanInfo();
                cleanInfoToUpdate.Id = cleanInfo[0].Id;
                cleanInfoToUpdate.IsReviewedPhone = true;
                update cleanInfoToUpdate;
                account.CleanStatus = 'Reviewed';
            }
        }
    }
}
```

- Create a customized set of `Industry` field values for accounts. Use triggers to map values from fields on imported or cleaned records onto a standard set of values.
- Read the `CleanStatus` field value on the Account object. If that value is `Different`, but a Salesforce record has no street address value, update the record's status to `Not Compared`.

AccountContactRole

Represents the role that a Contact plays on an Account.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`AccountId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Account.
`ContactId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update

Field	Details
	Description Required. ID of the Contact associated with this account.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsPrimary`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Specifies whether the Contact plays the primary role on the Account (`true`) or not (`false`). Note that each account has only one primary contact role. Label is **Primary**. Default value is `false`.
`Role`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Name of the role played by the Contact on this Account, such as Decision Maker, Approver, Buyer, and so on. Must be unique—there can't be multiple records in which the `AccountId`, `ContactId`, and `Role` values are identical. Different contacts can play the same role on the same account. A contact can play different roles on the same account.

Usage

Use this object to define the role that a Contact plays on a given Account within the context of a specific Opportunity.

See Also:

Account
Contact

AccountFeed

Represents a single feed item on an account record detail page.

An account feed shows changes to an account record for fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to accounts in Salesforce. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the Account object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of AccountFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the

Field	Details
	`ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`.The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort

Field	Details
	Description Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.

Field	Details
LastModifiedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, LastModifiedDate is the same as CreatedDate. If a FeedComment is inserted on that feed item, then LastModifiedDate becomes the CreatedDate for that FeedComment. Deleting the FeedComment does not change the LastModifiedDate. Ordering by LastModifiedDate DESC sorts the feed by both the most recent feed item or comment.
LikeCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of a LinkPost.
NetworkScope	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. NetworkScope can have the following values: • NetworkId—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community.

Field	Details
	• `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the account record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on

Field	Details
	cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist

Field	Details
	Properties Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for an account record.

See Also:

Account
EntitySubscription
NewsFeed
UserProfileFeed

AccountHistory

Represents the history of changes to the values in the fields of an account. This object is available in versions 11.0 and later.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort

Field	Details
	Description The name of the field that was changed.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`AccountId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Account. Label is **Account ID**.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.

Usage

Use this object to identify changes to an account.

This object respects field level security on the parent object.

See Also:

Account

AccountOwnerSharingRule

Represents the rules for sharing an account with a User other than the owner.

Note: Contact salesforce.com customer support to enable access to this object for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`AccountAccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Update **Description** A value that represents the type of sharing being allowed. The possible values are: • `Read` • `Edit` • `All` (This value isn't valid for creating or updating.)
`CaseAccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of access granted to the target Group for all child cases. The possible values are: • `None` • `Read` • `Edit`
`ContactAccessLevel`	**Type** picklist

Field	Details
	Properties Filter, Group, Restricted picklist, Sort **Description** A value that represents the type of access granted to the target Group, UserRole, or User for any associated contacts. The possible values are: • `None` • `Read` • `Edit` **Note:** When `DefaultContactAccess` is set to `Controlled by Parent`, you can't create or update this field.
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
`DeveloperName`	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Rule Name** in the user interface. This field is available in API version 24.0 and later. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`GroupId`	**Type** reference

Field	Details
	Properties Create, Filter, Group, Sort **Description** The ID representing the source group. An Account owned by a User in the source Group triggers the rule to give access.
`OpportunityAccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of access granted to the target Group for any associated Opportunity. The possible values are: • `None` • `Read` • `Edit`
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to **Label** on the user interface.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the User or Group being granted access.

Usage

Use this object to manage the sharing rules for accounts. General sharing and territory management-related sharing use this object. For example, the following code creates an account owner sharing rule between two public groups, which can also contain portal users.

```
AccountOwnerSharingRule rule = new AccountOwnerSharingRule();
rule.setName("RuleName"); // Set the sharing rule name
rule.setDeveloperName("RuleDeveloperName"); // Set the sharing rule developer name
rule.setGroupId("00Gx00000000000"); // Set the group of users to share records from
```

```
rule.setUserOrGroupId("00Gx00000000001"); // Set the group of users to share records to
rule.setAccountAccessLevel("Edit");
rule.setOpportunityAccessLevel("Read");
rule.setCaseAccessLevel("None");
connection.create(rule);
```

See Also:

Account

AccountShare

AccountPartner

This read-only object represents a partner relationship between two Account objects. It is automatically created when a Partner object is created for a partner relationship between two accounts.

Note: This object is completely distinct and independent of Account records that have been enabled for the partner portal.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`AccountFromId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the main Account in the partner relationship.
`AccountToId`	**Type** reference **Properties** Filter, Group, Nillable, Sort, **Description** ID of the partner Account in the partner relationship.
`IsDeleted`	**Type** boolean

Field	Details
	Properties Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsPrimary`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the AccountPartner is the main account's primary partner (`true`) or not (`false`).
`OpportunityId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the opportunity in a partner relationship with the related account.
`Role`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** The UserRole that the partner Account has on the main Account. For example, `Consultant` or `Distributor`.

Creating an Account-Account Partner Relationship

When you create a partner relationship between two accounts (when you create a Partner object and specify the `AccountFromId`), the API automatically creates two AccountPartner objects, one for the forward relationship and one for the reverse. For example, if you create a Partner object with "Acme, Inc." as the `AccountFromId` and "Acme Consulting" as the `AccountToId`, the API automatically creates two AccountPartner objects:

- The forward relationship AccountPartner with "Acme, Inc." as the `AccountFromId` and "Acme Consulting" as the `AccountToId`.
- The reverse relationship AccountPartner with "Acme Consulting" as the `AccountFromId` and "Acme, Inc." as the `AccountToId`.
- The value of the `Role` field in the reverse relationship AccountPartner is set to the PartnerRole object `ReverseRole` value associated with the value of the `Role` field in the forward relationship AccountPartner.

This mapping allows the API to manage the objects and their relationship efficiently.

See Also:

Partner
OpportunityPartner

AccountShare

Represents a sharing entry on an Account.

Supported Calls

`describeSObjects()`, `create()`, `delete()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`AccountAccessLevel`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Level of access that the User or Group has to the Account. The possible values are: • `Read` • `Edit` • `All` (This value isn't valid for create or update calls.) This field must be set to an access level that is at least equal to the organization's default Account access level. In addition, either this field, the `OpportunityAccessLevel` field, or the `CaseAccessLevel` field must be set higher than the organization's default access level.
`AccountId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Account associated with this sharing entry. This field can't be updated.
`CaseAccessLevel`	**Type** picklist

Field	Details
	Properties Filter, Group, Restricted picklist, Sort **Description** Level of access that the User or Group has to cases associated with the account. The possible values are: • `None` • `Read` • `Edit` This field must be set to an access level that is at least equal to the organization's default `CaseAccessLevel`. This field can't be updated via the API if the `AccountAccessLevel` field is set to `All`. You can't update this field for the associated account owner via the API. You must update the account owner's `CaseAccessLevel` via the Salesforce user interface.
`ContactAccessLevel`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Level of access that the User or Group has to contacts associated with the account. The possible values are: • `None` • `Read` • `Edit` This field must be set to an access level that is at least equal to the organization's default `ContactAccessLevel`. This field can't be updated via the API if the `ContactAccessLevel` field is set to "Controlled by Parent," you can't update this field for the associated account owner using the API. You must update the account owner's `ContactAccessLevel` via the Salesforce user interface.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`OpportunityAccessLevel`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort

Field	Details
	Description Level of access that the User or Group has to opportunities associated with the Account. The possible values are: • `None` • `Read` • `Edit` This field must be set to an access level that is at least equal to the organization's default opportunity access level. This field can't be updated via the API if the `AccountAccessLevel` field is set to `All`. You can't use the API to update this field for the associated Account owner. You must update the Account owner's opportunityAccessLevel via the Salesforce user interface.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. There are many possible values, including: • `Manual Sharing`—The User or Group has access because a User with "All" access manually shared the Account with them. • `Owner`—The User is the owner of the Account or is in a Role above the Account owner in the role hierarchy. • `Sales Team`—The User or Group has team access (is an AccountTeamMember). • `Sharing Rule`—The User or Group has access via an Account sharing rule.
`UserOrGroupId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the User or Group that has been given access to the Account. This field can't be updated.

Usage

This object allows you to determine which users and groups can view or edit Account records owned by other users.

If you attempt to create an AccountShare record that matches an existing record, the request updates any modified fields and returns the existing record.

For example, the following code finds all accounts owned by a user and manually shares them to a portal user.

```
QueryResult result = conn.query("SELECT Id FROM Account WHERE OwnerId = '005D0000001LPFB'");
// Create a new AccountShare object
```

```
List<AccountShare> shares = new ArrayList<AccountShare>();
for (SObject rec : result.getRecords()) {
     AccountShare share = new AccountShare();
    share.setAccountId(rec.getId());
    //Set the portal user Id to share the accounts with
    share.setUserOrGroupId("003D000000QA8Tl");
    share.setAccountAccessLevel("Edit");
    share.setOpportunityAccessLevel("Read");
    share.setCaseAccessLevel("Edit");
    shares.add(share);
}
conn.create(shares.toArray(new AccountShare[shares.size()]));
```

This code shares the accounts that the user owns at the time, but not those that are owned at a later time. For these types of shares, use an owner-based sharing rule such as AccountOwnerSharingRule.

See Also:

Account
CaseShare
LeadShare
OpportunityShare

AccountTag

Associates a word or short phrase with an Account.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the

Field Name	Details
	same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.
`Type`	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

AccountTag stores the relationship between its parent TagDefinition and the Account being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

AccountTeamMember

Represents a User who is a member of an Account team. See also UserAccountTeamMember, which represents a User who is on the default account team of another user.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- This object is available only for Enterprise, Unlimited, and Performance Edition users who have enabled the account team functionality.
- Customer Portal users can't access this object.

Fields

Field Name	Details
`AccountAccessLevel`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Level of access that the User or Group has to the Account. The possible values are: • `None` • `Read` • `Edit` • `All` This field must be set to an access level that is at least equal to the organization's default Account access level. In addition, the Account`AccountAccessLevel`, `OpportunityAccessLevel`, or `CaseAccessLevel` field must be set higher than the organization's default access level.
`AccountId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Account to which this user is a team member. Must be a valid account ID.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**. **Note:** Any AccountTeamMember record that is deleted is not moved to the Recycle Bin, and cannot be undeleted, unless the record was cascade-deleted when deleting a related Account. For directly deleted AccountTeamMember records, don't use the isDeleted field to detect deleted records in SOQL queries or `queryAll()` calls. Instead, use `getDeleted()`.
`TeamMemberRole`	**Type** picklist

Field Name	Details
	Properties Create, Filter, Nillable, Update **Description** Role associated with this team member. One of the valid team member roles defined for your organization. Label is **Team Role**.
`UserId`	**Type** reference **Properties** Create, Filter **Description** Required. ID of the User who is a member of this account team. Must be a valid User ID.

Usage

Use this object to manage the team members of a particular Account and to specify team member roles for those users on that account.

See Also:

Account

AccountTerritoryAssignmentRule

An account assignment rule that assigns accounts to territories based on account fields. Only available if territory management has been enabled for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`BooleanFilter`	**Type** string **Properties** Create, Filter, Nillable, Update

Field	Details
	Description Advanced filter conditions that were specified for the rule in the online application. For example, "(1 AND 2) OR 3."
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Update **Description** Indicates whether the rule is active (`true`) or inactive (`false`). Via the API, active rules run automatically when new accounts are created and existing accounts are edited. The exception is when the `IsExcludedFromRealign` field on an account is `true`, which prevents account assignment rules from evaluating that account.
`IsInherited`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Update **Description** Indicates whether the rule is an inherited rule (`true`) or a local rule (`false`). An inherited rule also acts upon territories below it in the territory hierarchy. A local rule is created at the immediate territory and only impacts the immediate territory.
`Name`	**Type** string **Properties** Create, Filter, Update **Description** A name for the rule. Limit is 80 characters.
`TerritoryId`	**Type** reference **Properties** Create, Filter, Update **Description** ID of the territory where accounts that satisfy this rule are assigned.

Usage

A territory will not have any accounts (with the exception of manually assigned accounts) unless at least one account assignment rule is active for the territory.

See Also:

AccountTerritoryAssignmentRuleItem
Territory
UserTerritory

AccountTerritoryAssignmentRuleItem

A row of selection criteria for an AccountTerritoryAssignmentRule object. Only available if territory management has been enabled for your organization. AccountTerritoryAssignmentRuleItem can be created or deleted if the `BooleanFilter` field on its corresponding AccountTerritoryAssignmentRule object is a null value.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`Field`	**Type** picklist **Properties** Create, Filter, Group, Sort, Update **Description** The standard or custom account field to use as a criteria.
`Operation`	**Type** picklist **Properties** Create, Filter, Group, Sort, Update **Description** The criteria to apply, such as “equals” or “starts with.”
`RuleID`	**Type** reference

Field	Details
	Properties Create, Filter, Update **Description** ID of the associated AccountTerritoryAssignmentRule.
`SortOrder`	**Type** int **Properties** Create, Filter, Update **Description** The order in which this row is evaluated compared to other AccountTerritoryAssignmentRuleItem objects for the given AccountTerritoryAssignmentRule.
`Value`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** The field value(s) to evaluate, such as `94105` if the `Field` is `Billing Zip/Postal Code`.

Usage

- Both standard and custom account fields can be used as criteria for account assignment rules.
- A territory will not have any accounts (with the exception of manually assigned accounts) unless at least one account assignment rule is active for the territory.

See Also:

AccountTerritoryAssignmentRule
Territory
UserTerritory

AccountTerritorySharingRule

Represents the rules for sharing an Account within a Territory.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`AccountAccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of sharing being allowed. The possible values are: • `Read` • `Edit` • `All`
`CaseAccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of access granted to the target group for all child cases of the account. The possible values are: • `None` • `Read` • `Edit`
`ContactAccessLevel`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A value that represents the type of access granted to the target group for all related contacts on the account. The possible values are: • `None` • `Read` • `Edit` **Note:** This field is read only.

Field	Details
Description	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
DeveloperName	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Rule Name** in the user interface. This field is available in API version 24.0 and later. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
GroupId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the source group. Accounts owned by users in the source territory trigger the rule to give access.
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to **Label** on the user interface.

Field	Details
`OpportunityAccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of access granted to the target group for all opportunities associated with the account. The possible values are: • `None` • `Read` • `Edit`
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the user or group being given access, or, if a territory ID, the users assigned to that territory.

Usage

Use this object to manage the sharing rules for a particular object. General sharing and Territory-related sharing use this object.

See Also:

Account
AccountShare

ActivityHistory

This read-only object is displayed in a related list of closed activities—past events and closed tasks—related to an object. It includes activities for all contacts related to the object. ActivityHistory fields for phone calls are only available if your organization uses Salesforce CRM Call Center.

Supported Calls

`describeSObjects()`

Fields

Field	Details
AccountId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the related account, which is determined as follows: • The account associated with the `WhatId`, if it exists; or • The account associated with the `WhoId`, if it exists; otherwise • `null` For information on IDs, see ID Field Type.
ActivityDate	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Indicates one of the following: • The due date of a task • The due date of an event if `IsAllDayEvent` is set to `true` This field has a time stamp that is always set to midnight in the Universal Time Coordinated (UTC) time zone. The time stamp doesn't represent the time of the activity; don't attempt to alter it to accommodate time zone differences. Label is `Date`.
ActivityType	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Represents one of the following values: `Call`, `Meeting`, or `Other`.
CallDisposition	**Type** string **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Represents the result of a given call, for example, “we'll call back,” or “call unsuccessful.” Limit is 255 characters.
`CallDurationInSeconds`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Duration of the call in seconds.
`CallObject`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Name of a call center. Limit is 255 characters.
`CallType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of call being answered: Inbound, Internal, or Outbound.
`ConnectionReceivedId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the PartnerNetworkConnection that shared this record with your organization. This field is available only if your organization has enabled Salesforce to Salesforce and only in API versions 28.0 and later.
`ConnectionSentId`	**Type** reference

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** Indicates the ID of the PartnerNetworkConnection that your organization shared this record with. This field is available only if your organization has enabled Salesforce to Salesforce, and only in API versions 28.0 and later. The value is always `null`. You can use the PartnerNetworkRecordConnection object to forward records to connections.
`Description`	**Type** textarea **Properties** Nillable **Description** Contains a description of the event or task.
`Division`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A logical segment of your organization's data. Available only if the organization has the Division permission enabled.
`DurationInMinutes`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Indicates the duration of the event or task.
`EndDateTime`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Indicates the end date and time of the event or task. Available in versions 27.0 and later. This field is optional, depending on the following:

Field	Details
	• If `IsAllDayEvent` is true, you can supply a value for either `DurationInMinutes` or `EndDateTime`. Supplying values in both fields is allowed if the values add up to equal time spans. If both fields are `null`, the duration defaults to one day. • If `IsAllDayEvent` is false, a value must be supplied for either `DurationInMinutes` or `EndDateTime`. Supplying values in both fields is allowed if the values add up to the same amount of time.
`IsAllDayEvent`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** If the value of this field is set to `true`, then the activity is an event and the `ActivityDate` defines the date of the event. If the value of this field is set to `false`, then the activity may be a task or an event.
`IsClosed`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether a task is closed; value is always `true`. This field is set indirectly by setting the `Status` field on the task—each picklist value has a corresponding `IsClosed` value. Label is `Closed`.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the activity has been moved to the Recycle Bin (`true`) or not (`false`). Label is `Deleted`.
`IsHighPriority`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates a high-priority task. This field is derived from the `Priority` field.
`IsOnlineMeeting`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the activity represents an online meeting (`true`) or not (`false`). **Note:** This field is not available in API version 16.0 or later.
`IsReminderSet`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether a reminder is set for an activity (`true`) or not (`false`).
`IsTask`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** If the value of this field is set to `true`, then the activity is a task. If the value is set to `false`, then the activity is an event. Label is `Task`.
`IsVisibleInSelfService`	**Type** boolean **Properties** Defaulted on create, Filter **Description** If the value of this field is set to `true`, then the activity can be viewed in the self-service portal.

Field	Details
`Location`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** If the activity is an event, then this field contains the location of the event. If the activity is a task, the value is `null`.
`OwnerId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the user who owns the activity.
`PrimaryAccountId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Contains the `AccountId` value from the activity record. Available in API versions 30.0 and later to organizations that have enabled Shared Activities.
`PrimaryWhoId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Contains the `AccountId` value from the activity record. Available in API versions 30.0 and later to organizations that have enabled Shared Activities.
`Priority`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Indicates the priority of a task, such as high, normal, or low.

Field	Details
ReminderDateTime	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Represents the time when the reminder is scheduled to fire, if `IsReminderSet` is set to `true`. If `IsReminderSet` is set to `false`, then the user may have deselected the reminder checkbox in the Salesforce user interface, or the reminder has already fired at the time indicated by the value.
Status	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Indicates the current status of a task, such as in progress or complete. Each predefined status field sets a value for `IsClosed`. To obtain picklist values, query TaskStatus.
Subject	**Type** combobox **Properties** Filter, Nillable, Sort **Description** Contains the subject of the task or event.
WhatId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the related object (Campaign, Case, Opportunity, or custom object). Label is `Opportunity/Account ID`.
WhoId	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Indicates the ID of the related lead or contact. If the ID in this field refers to a lead, then the `WhatId` field must be `null`. If your organization uses Shared Activities, when you query activities in API version 30.0 or later, then the returned value of the `WhoId` field matches the value in the queried object, not necessarily in the activity record itself. Label is `Contact/Lead ID`.

Usage

Query activities related to an object

1. Optionally, issue a describe call against the object whose activities you wish to query, to get a suggestion of the correct SOQL to use.
2. Issue a SOQL relationship query with a main clause that references the object, and an inner clause that references the activity history, for example:

```
SELECT (SELECT ActivityDate, Description FROM ActivityHistories)
FROM Account
WHERE Name Like 'XYZ%'
```

or

```
SELECT (SELECT ActivityDate, Description FROM OpenActivities)
FROM Account
WHERE Name Like 'XYZ%'
```

The user interface enforces sharing rules, filtering out related-list items that a user doesn't have permission to see.

The following restrictions on users who don't have "View All Data" permission help prevent performance issues:

- In the main clause of the relationship query, you can reference only one record. For example, you can't filter on all records where the account name starts with 'A'; instead, you must reference a single account record.
- You can't use `WHERE` clauses.
- You must specify a limit of 499 or fewer on the number of rows returned in the list.
- You must sort on `ActivityDate` in descending order and `LastModifiedDate` in descending order; you can display nulls last. For example: `ORDER BY ActivityDate DESC NULLS LAST, LastModifiedDate DESC`.

See Also:

Task

AdditionalNumber

This object represents an optional additional number for a call center. This additional number is visible in the call center's phone directory.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`CallCenterId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** System field that contains the ID of the user who created the call center associated with this additional number. If value is null, this additional number is displayed in every call center's phone directory.
`Description`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Description of the additional number, such as Conference Room B. Limit: 255 characters.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The name of the additional number. Limit: 80 characters.
`Phone`	**Type** phone **Properties** Create, Filter, Nillable, Group, Sort, Update **Description** The phone number that corresponds to this additional number.

Usage

Create an additional number for a call center directory. Use this object if the number is not easily categorized as a User, Contact, Lead, Account, or the other object. Examples include phone queues or conference rooms.

AllowedEmailDomain

Represents an allowed email domain for users in your organization. You can define a whitelist to restrict the email domains allowed in a user's `Email` field. This object is available in API version 29.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

You must have the “Manage Users” user permission to use this object.

Note: If you don't see this object, contact your salesforce.com representative to enable it.

Fields

Field	Details
`Domain`	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update **Description** An allowed email domain for users.

Announcement

Represents a Chatter group announcement. This object is available in API version 30.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
`ExpirationDate`	**Type** dateTime **Properties** Create, Filter, Sort, Update

Field Name	Details
	Description Required. The date on which the announcement expires. Announcements display on the group UI until 11:59 p.m. local time on the selected date.
`FeedItemId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. The ID of the FeedItem that contains the content of the announcement. Announcements are stored as text posts.
`ParentId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the parent CollaborationGroup that the announcement belongs to. An announcement can belong only to a single Chatter group.

Usage

Group owners, managers, and users with the “Modify All Data” permission can use the Announcement object to create, edit, and delete group announcements. Creating a group announcement is a three-step process:

1. Use the FeedItem object to create a text post with the announcement’s content. Use the CollaborationGroup record you want to post the announcement to as the parent of this feed item.
2. Next, use the feed item ID and an expiration date to create the announcement record.
3. Finally, update the `AnnouncementId` field in the CollaborationGroup record with the ID of the announcement you created.

To delete the group announcement, simply delete the `AnnouncementId` value in the CollaborationGroup record. To restore a group announcement, update the `AnnouncementId` field for a group with the announcement’s ID. The expiration date for the announcement should be in the future and the feed item used to create the announcement should be parented by the same group.

ApexClass

Represents an Apex class.

Note: Although Apex classes and triggers have the Create and Update field properties, a runtime exception occurs if you try to create or update them using the API. Instead, use the Force.com Migration Tool, the Salesforce user interface, or the Force.com IDE to create or update Apex classes or triggers.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`ApiVersion`	**Type** double **Properties** Create, Filter, Sort, Update **Description** The API version for this class. Every class has an API version specified at creation.
`Body`	**Type** textarea **Properties** Create, Nillable, Update **Description** The Apex class definition. Limit: 100,000 characters.
`bodyCrc`	**Type** double **Properties** Create, Defaulted on create, Filter, Nillable, Sort, Update **Description** The CRC (cyclic redundancy check) of the class or trigger file.
`IsValid`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether any dependent metadata has changed since the class was last compiled (`true`) or not (`false`).
`LengthWithoutComments`	**Type** int

Field	Details
	Properties Create, Filter, Group, Sort, Update **Description** Length of the class without comments.
`Name`	**Type** string **Properties** Group, Sort, Create, Filter, Update **Description** Name of the class. Limit: 255 characters
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.
`Status`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The current status of the Apex class. The following string values are valid: • `Active`—The class is active.

Field	Details
	• `Deleted`—The class is marked for deletion. This is useful for managed packages, because it allows a class to be deleted when a managed package is updated. **Note:** The ApexTrigger `Status` field includes an `Inactive` option, but it is only supported for ApexTrigger. For more information, see the *Force.comMetadata API Developer's Guide*.

Usage

For information, see the *Force.com Apex Code Developer's Guide*.

See Also:

ApexTrigger
Developer's Guide: Force.com Apex Code Developer's Guide

ApexComponent

Represents a definition for a custom component that can be used in a Visualforce page alongside standard components such as `<apex:relatedList>` and `<apex:dataTable>`. For information, see the *Visualforce Developers Guide*.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`ApiVersion`	**Type** double **Properties** Create, Filter, Sort, Update **Description** The API version for this custom component. Every custom component has an API version specified at creation. If the API version is less than 15.0 and `ApiVersion` is not specified, `ApiVersion` defaults to 15.0.
`ControllerKey`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The identifier for the controller associated with this custom component:

Field	Details
	• If the `ControllerType` parameter is set to `Standard` or `StandardSet`, this value is the name of the sObject that defines the controller. • If the `ControllerType` parameter is set to `Custom`, this value is the name of the Apex class that defines the controller.
`ControllerType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The type of controller associated with this Visualforce custom component. Possible values include: • `Not Specified`, for custom components defined without a value for the `controller` attribute on the `<apex:component>` tag • `Standard`, a value that can't be used with custom components or errors may occur • `StandardSet`, a value that can't be used with custom components or errors may occur • `Custom`, for components that have a value for the `controller` attribute on the `<apex:component>` tag
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Description of the Visualforce custom component.
`Markup`	**Type** textarea **Properties** Create, Update **Description** The Visualforce markup, HTML, Javascript, and any other Web-enabled code that defines the content of the custom component.
`MasterLabel`	**Type** string **Properties** Create, Filter, Group, Sort, Update

Field	Details
	Description The text used to identify the Visualforce custom component in the Setup area of Salesforce. The Label for this field is **Label**.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of this Visualforce custom component.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.

Usage

Use custom components to encapsulate a common design pattern and then reuse that pattern several times in one or more Visualforce pages. All users who can view Visualforce pages can view custom components, but the “Customize Application” permission is required to create or update custom components.

See Also:

ApexPage

StaticResource

Developer's Guide: Visualforce Developer's Guide

ApexLog

Represents an Apex debug log.

This object is available in API version 19.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`Application`	**Type** string **Properties** Filter, Group, Sort **Description** This value depends on the client type that triggered the log. • For API clients, this value is the client ID • For browser clients, this value is `Browser`
`DurationMilliseconds`	**Type** int **Properties** Filter, Group, Sort **Description** Duration of the transaction in milliseconds.
`Location`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies the location of the origin of the log. Values are: • `Monitoring`—log is generated as part of debug log monitoring. These types of logs are maintained until the user or the system overwrites them. • `SystemLog`—log is generated from the Developer Console. These types of logs are only maintained for 60 minutes or until the user clears them.
`LogLength`	**Type** int

Field	Details
	Properties Filter, Group, Sort **Description** Length of the log in bytes.
`LogUserId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the user whose actions triggered the debug log.
`Operation`	**Type** string **Properties** Filter, Group, Sort **Description** Name of the operation that triggered the debug log, such as `APEXSOAP`, `Apex Sharing Recalculation`, and so on.
`Request`	**Type** string **Properties** Filter, Group, Sort **Description** Request type. Values are: • `API`—request came from the API • `Application`—request came from the Salesforce user interface
`StartTime`	**Type** dateTime **Properties** Filter, Sort **Description** Start time of the transaction.
`Status`	**Type** string

Field	Details
	Properties Filter, Group, Sort **Description** Status of the transaction. This value is either `Success`, or the text of an unhandled Apex exception.

Usage

You can read information about this object, as well as delete it: you can't update or insert. For information, see the *Force.com Apex Code Developer's Guide*.

See Also:

ApexClass
ApexTrigger
Developer's Guide: Force.com Apex Code Developer's Guide

ApexPage

Represents a single Visualforce page.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`ApiVersion`	**Type** double **Properties** Create, Filter, Sort, Update **Description** The API version for this page. Every page has an API version specified at creation. If the API version is less than 15.0 and `ApiVersion` is not specified, `ApiVersion` defaults to 15.0.
`ControllerKey`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description The identifier for the controller associated with this page: • If the `ControllerType` parameter is set to `Standard` or `StandardSet`, this value is the name of the sObject that defines the controller. • If the `ControllerType` parameter is set to `Custom`, this value is the name of the Apex class that defines the controller.
`ControllerType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The type of controller associated with this Visualforce page. Possible values include: • `Not Specified`, for pages defined with neither a `standardController` nor a `controller` attribute on the `<apex:page>` tag • `Standard`, for pages defined with the `standardController` attribute on the `<apex:page>` tag • `StandardSet`, for pages defined using the `standardController` and `recordSetVar` attribute on the `<apex:page>` tag • `Custom`, for pages defined with the `controller` attribute on the `<apex:page>` tag
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Description of the Visualforce page.
`isAvailableInTouch`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates if Visualforce tabs associated with the Visualforce page can be used in the Salesforce1 app (`true`) or not (`false`). (Use of this field for Salesforce Touch is deprecated.) This field is available in API version 27.0 and later. Standard object tabs that are overridden with a Visualforce page aren't supported in Salesforce1, even if you set this field for the page. The default Salesforce1 page for the object is displayed instead of the Visualforce page.

Field	Details
`IsConfirmationTokenRequired`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether `GET` requests for the page require a CSRF confirmation token (`true`) or not (`false`). This field is available in API version 28.0 and later. If you change this field's value from `false` to `true`, links to the page require a CSRF token to be added to them, or the page will be inaccessible.
`Markup`	**Type** textarea **Properties** Create, Update **Description** The Visualforce markup, HTML, Javascript, and any other Web-enabled code that defines the content of the page.
`MasterLabel`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The text used to identify the Visualforce page in the Setup area of Salesforce. The Label is **Label**.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of this Visualforce page.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.

Usage

Use Visualforce pages to add custom content that extends the base Salesforce application functionality. All users in Visualforce-enabled organizations can view Visualforce pages, but the “Customize Application” permission is required to create or update them.

For information, see the *Visualforce Developers Guide*.

See Also:

- *ApexComponent*
- *StaticResource*
- *Developer's Guide: Visualforce Developer's Guide*

ApexTestQueueItem

Note: The API for asynchronous test runs is a Beta release.

Represents a single Apex class in the Apex job queue. This object is available in API version 23.0 and later.

Supported Calls

`create()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Description
`ApexClassId`	**Type** reference

Field Name	Description
	Properties Create, Filter, Group, Sort **Description** The Apex class whose tests are to be executed. This field can't be updated.
`ExtendedStatus`	**Type** string **Properties** Filter, Nillable, Sort **Description** The pass rate of the test run. For example: “(4/6)”. This means that four out of a total of six tests passed. If the class fails to execute, this field contains the cause of the failure.
`ParentJobId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Read-only. Points to the AsyncApexJob that represents the entire test run. If you insert multiple Apex test queue items in a single bulk operation, the queue items will share the same parent job. This means that a test run can consist of the execution of the tests of several classes if all the test queue items are inserted in the same bulk operation.
`Status`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update **Description** The status of the job. Valid values are: • `Holding`[1] • `Queued` • `Preparing` • `Processing`

Field Name	Description
	• `Aborted` • `Completed` • `Failed` [1] This status is for the Apex Flex Queue pilot.

Usage

Insert an `ApexTestQueueItem` object to place its corresponding Apex class in the Apex job queue for execution. The Apex job executes the test methods in the class.

To abort a class that is in the Apex job queue, perform an update operation on the `ApexTestQueueItem` object and set its `Status` field to Aborted.

If you insert multiple Apex test queue items in a single bulk operation, the queue items will share the same parent job. This means that a test run can consist of the execution of the tests of several classes if all the test queue items are inserted in the same bulk operation.

ApexTestResult

Note: The API for asynchronous test runs is a Beta release.

Represents the result of an Apex test method execution. This object is available in API version 23.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ApexClassId`	**Type** reference **Properties** Filter, Group, Sort **Description** The Apex class whose test methods were executed.
`ApexLogId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description Points to the ApexLog for this test method execution if debug logging is enabled; otherwise, `null`.
`AsyncApexJobId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Read-only. Points to the AsyncApexJob that represents the entire test run. This field points to the same object as `ApexTestQueueItem.ParentJobId`.
`Message`	**Type** string **Properties** Filter, Nillable, Sort **Description** The exception error message if a test failure occurs; otherwise, `null`.
`MethodName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The test method name.
`Outcome`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The result of the test method execution. Can be one of these values: • Pass • Fail

Field Name	Details
	• CompileFail
QueueItemId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Points to the ApexTestQueueItem which is the class that this test method is part of.
StackTrace	**Type** string **Properties** Filter, Nillable, Sort **Description** The Apex stack trace if the test failed; otherwise, `null`.
TestTimestamp	**Type** dateTime **Properties** Filter, Sort **Description** The start time of the test method.

Usage

You can query the fields of the `ApexTestResult` record that corresponds to a test method executed as part of an Apex class execution.

Each test method execution is represented by a single `ApexTestResult` record. For example, if an Apex test class contains six test methods, six `ApexTestResult` records are created. These records are in addition to the `ApexTestQueueItem` record that represents the Apex class.

ApexTrigger

Represents an Apex trigger.

Note: Although Apex classes and triggers have the Create and Update field properties, a runtime exception occurs if you try to create or update them using the API. Instead, use the Force.com Migration Tool, the Salesforce user interface, or the Force.com IDE to create or update Apex classes or triggers.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`ApiVersion`	**Type** double **Properties** Create, Filter, Sort, Update **Description** The API version for this trigger. Every trigger has an API version specified at creation.
`Body`	**Type** textarea **Properties** Create, Nillable, Update **Description** The Apex trigger definition. Limit: 32,000 characters.
`bodyCrc`	**Type** double **Properties** Create, Defaulted on create, Filter, Nillable, Sort, Update **Description** The CRC (cyclic redundancy check) of the class or trigger file.
`IsValid`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether any dependent metadata has changed since the trigger was last compiled (`true`) or not (`false`).
`LengthWithoutComments`	**Type** int

Field	Details
	Properties Create, Filter, Group, Sort, Update **Description** Length of the trigger without comments
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Name of the trigger. Limit: 255 characters
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.
`Status`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The current status of the Apex trigger. The following string values are valid: • `Active`—The trigger is active. • `Inactive`—The trigger is inactive, but not deleted.

Field	Details
	• `Deleted`—The trigger is marked for deletion. This is useful for managed packages, because it allows a class to be deleted when a managed package is updated. **Note:** `Inactive` is not valid for ApexClass. For more information, see the *Force.com Metadata API Developer's Guide*.
`TableEnumOrId`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Specifies the object associated with the trigger, such as Account or Contact.
`UsageAfterDelete`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether the trigger is an `after delete` trigger (`true`) or not (`false`).
`UsageAfterInsert`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether the trigger is an `after insert` trigger (`true`) or not (`false`).
`UsageAfterUndelete`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether the trigger is an `after undelete` trigger (`true`) or not (`false`).
`UsageAfterUpdate`	**Type** boolean **Properties** Create, Filter, Update

Field	Details
	Description Specifies whether the trigger is an `after update` trigger (`true`) or not (`false`).
`UsageBeforeDelete`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether the trigger is a `before delete` trigger (`true`) or not (`false`).
`UsageBeforeInsert`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether the trigger is a `before insert` trigger (`true`) or not (`false`).
`UsageBeforeUpdate`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether the trigger is a `before update` trigger (`true`) or not (`false`).
`UsageIsBulk`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether the trigger is defined as a bulk trigger (`true`) or not (`false`). **Note:** This field is not used for Apex triggers saved using Salesforce.com API version 10.0 or higher: all triggers starting with that version are automatically considered bulk, and this field will always return `true`.

Usage

For more information about Apex triggers, see the *Force.com Apex Code Developer's Guide*.

See Also:

ApexClass
Developer's Guide: Force.com Apex Code Developer's Guide

AppMenuItem

Represents an item in the App Picker menu.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`CanvasAccessMethod`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The access method for the canvas app. Values can be: • OAuth Webflow (GET) • Signed Request (POST)
`CanvasEnabled`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates if the app menu item is a canvas app (`true`) or not (`false`).
`CanvasReferenceId`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The canvas app unique identifier.

Field	Details
`CanvasSelectedLocations`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The selected locations for the canvas app which define where the canvas app can appear in the user interface. For example: `Chatter,ChatterFeed,Publisher,ServiceDesk`
`CanvasUrl`	**Type** url **Properties** Filter, Group, Nillable, Sort **Description** The URL of the canvas app.
`Description`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A description of this menu item.
`IconUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The icon for the menu item's application.
`InfoUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The URL for more information about the application.

Field	Details
IsUsingAdminAuthorization	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** If `true`, the app is pre-authorized for certain users by the administrator.
Label	**Type** string **Properties** Group, Nillable **Description** The app's name.
LogoUrl	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The logo for the menu item's application. The default is a cloud.
MobileStartUrl	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The location mobile users are directed to after they've authenticated. This is only used with connected apps.
Name	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The API name of the item.
NamespacePrefix	**Type** string

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects. This field can't be accessed unless the logged-in user has the "Customize Application" permission.
`SortOrder`	**Type** int **Properties** Filter, Group, idLookup, Sort **Description** The index value that controls where this item appears in the menu. For example, a menu item with a sort order of 5 will appear between items with sort order values of 3 and 9.
`StartUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The location users are directed to after they've authenticated. For a connected app, this is the location specified by the `StartUrl`. Otherwise it's the application's default start page.
`Type`	**Type** string **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field	Details
	Description The type of application represented by this item. The types are: • ConnectedApplication • ServiceProvider • TabSet

Usage

Use this read-only object to view an entry in the Force.com App Menu. You can create a SOQL query to retrieve all items, even those a user does not see from the user interface.

There are many ways you can use AppMenuItem. Here are some examples:

- Build your own App Launcher or App Menu in Salesforce. Create a custom page showing all the apps you have access to and that lets you run them using single sign-on.
- Build your own App Launcher or App Menu on a tablet or mobile app. You can have your own app for launching applications on a variety of mobile devices.
- Build an app launcher into your company's intranet. There's no need to have it run on Salesforce because there's a Web service API to let you build an app launcher.

Approval

Represents an approval request for a Contract.

Note: This object exists for backwards compatibility. This object is specific to approvals on the Contract object. It isn't equal to or involved in the approval processes represented by the ProcessInstance, which is more powerful.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`ApproveComment`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** Text entered by the user when they approved or rejected this approval request. Required. Limit: 4,000 characters.

Field	Details
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`OwnerId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** Required. ID of the User being asked to approve or reject the approval request. Must be a valid User ID. Required.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Contract associated with this approval request. Must be a valid contract ID.
`RequestComment`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** Text entered by the User who created the approval request. Optional. This field can't be updated after the Approval has been created. Limit: 4,000 characters.
`Status`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. Status of this approval request. One of the following picklist values: • `Pending`—Specified only when the Approval request is created (`create()` call)

Field	Details
	• `Approved`—Specified only when the Approval request is approved (`update()` call) • `Rejected`—Specified when the Approval request is rejected (`update()` call) or when it is created (`create()` call) and immediately rejected for archival/historical purposes.

Usage

This object allows client applications to programmatically handle approval requests for a Contract. Initially, to request a Contract approval, a client application might create a new Approval request record, specifying the `ParentId`, `OwnerId` (user approving or rejecting the request), `Status` (Pending), and (optionally) `RequestComment` fields. Note that when a client application creates the first approval request, if the value of the Contract `Status` field is Draft, then the Approval `Status` for this record is automatically changed to In Approval Process (see ContractStatus for more information).

A client application might subsequently update an existing Approval request, specifying the `Status` (Approved or `Rejected`) and an `ApproveComment` (required); the `RequestComment` field can't be updated. Updating an Approval record (either to approve or reject) requires the client application to be logged in with “Approve Contract” permission. To update an Approval request, its `Status` must be Pending—a client application can't update an Approval that has already been Approved or Rejected. To re-submit an approval request for a given Contract, a client application must create a new, separate Approval record and repeat the approval process.

Once a Contract has been approved (not rejected), the Contract `LastApprovedDate` field is automatically updated, however the Contract `Status` field isn't updated, it keeps the value InApproval.

An approved Contract must be activated explicitly. Client applications can activate a Contract by setting the value in its `Status` field to Activated, or a User can activate a Contract via the Salesforce user interface.

A Contract can have multiple approval requests in various states (Pending, Approved, and Rejected). In addition, one User can have multiple approval requests associated with the same Contract.

Client applications can't explicitly deleteApproval records. Approval records are deleted automatically if the parent Contract is deleted.

See Also:

Standard and Custom Object Basics

*`Article Type`*__DataCategorySelection

A data category selection represents a data category that classifies an article. This object can be used to associate an article with data categories from a data category group or to query the category selections for an article.

The object name is variable and has a syntax of *`Article Type`*__DataCategorySelection, where *`Article Type`* is the `Object Name` for the article type associated with the article. For example, `Offer__DataCategorySelection` represents the association between the `Offer` article type and its data categories. Every article is associated with an article type.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `getDeleted()`, `retrieve()`

Special Access Rules

Users can only access, create or delete data category selection visible to their role, permission set, or profile. If a user has partial visibility on an article's categorization, only the visible categories are returned.

Fields

Field Name	Details
DataCategoryGroupName	**Type** DataCategoryGroupReference **Properties** Create **Description** Unique name of the data category group which has categories associated with the article.
DataCategoryName	**Type** DataCategoryGroupReference **Properties** Create **Description** Unique name of the data category associated with the article.
ParentId	**Type** reference **Properties** Create, Filter **Description** ID of the article associated with the data category selection.

Usage

Every article in Salesforce Knowledge can be categorized. A data category selection represents a category that has been selected to classify an article. You can use the *Article Type*__DataCategorySelection object to query and manage article categorization in your organization. Client applications can create a categorization for an article with a Draft status. They can also delete and query article categorizations.

Note: When using *Article Type*__DataCategorySelection to classify an article, you can't select both a category (for example USA) and one of its descendants (California) or ascendant categories (North America). In this case, only the first category is selected.

Answer communities use QuestionDataCategorySelection to classify questions.

SOQL Sample

The following SOQL query returns the data category selections used to classify the article whose ID is `ka0D000000005ApIAI`.

```
SELECT Id,DataCategoryName, ParentId
FROM Offer__DataCategorySelection WHERE ParentId='ka0D000000005ApIAI'
```

This clause only returns category unique names. To retrieve category labels use the following clause:

```
SELECT Id,toLabel(DataCategoryName), ParentId
FROM Offer__DataCategorySelection WHERE ParentId='ka0D000000005ApIAI'
```

Tip: You can also use relationship queries to retrieve categorizations from an article type.

See Also:

QuestionDataCategorySelection
Data Categories
Articles

*Article Type*__Feed

Represents a single feed item in the feed displayed on the detail page for an article. An article feed shows recent changes to an article record for any fields that are tracked in feeds, as well as posts and comments about the article. It is a useful way to stay up-to-date with changes made to articles in Salesforce Knowledge. This object is available in API version 20.0 and later. The object name is variable and has a syntax of `Article Type`__Feed, where `Article Type` is the `Object Name` for the article type associated with the article. For example, `Offer__Feed` represents a feed item on an article based on the `Offer` article type.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Body`	**Type** textarea **Properties** Create, Filter, Nillable, Sort **Description** The content of the FeedItem. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.

Field	Details
ContentData	**Type** base64 **Properties** Create, Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
ContentDescription	**Type** textarea **Properties** Create, Nillable, Sort **Description** The description of the file specified in `ContentData`.
ContentFileName	**Type** string **Properties** Create, Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
ContentSize	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
ContentType	**Type** string **Properties** Group, Nillable, Sort

Field	Details
	Description Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDate DESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.

Field	Details
LastModifiedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, LastModifiedDate is the same as CreatedDate. If a FeedComment is inserted on that feed item, then LastModifiedDate becomes the CreatedDate for that FeedComment. Deleting the FeedComment does not change the LastModifiedDate. Ordering by LastModifiedDate DESC sorts the feed by both the most recent feed item or comment.
LikeCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
LinkUrl	**Type** url **Properties** Create, Filter, Nillable, Sort **Description** The URL of a LinkPost.
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the article that is tracked in the feed. The feed is displayed on the detail page for this record.
RelatedRecordId	**Type** reference

Field	Details
	Properties Create, Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as a FeedItem object of `Type` `ContentPost`.
`Title`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort **Description** The title of the FeedItem. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Create,Filter, Group **Description** The type of FeedItem. With the exception of `ContentPost`, `LinkPost`, and `TextPost`, FeedItem types should not be created directly from the API. • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold.

Field	Details
	• `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case. **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.

Usage

Use this object to track changes for an article. You can only delete a feed if you created it, or if you have the "Modify All Data" permission or "Modify All" permission on the KnowledgeArticle object.

See Also:

KnowledgeArticle
EntitySubscription
NewsFeed
UserProfileFeed

Asset

Represents an item of commercial value owned by an Account or Contact, for example, a product previously sold and installed.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`AccountId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Required. ID of the Account associated with this asset. Must be a valid account ID. Required if `ContactId` is not specified.
`ContactId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Required if `AccountId` is not specified. ID of the Contact associated with this asset. Must be a valid contact ID that has an account parent (but does not need to match the asset's `AccountId`).
`Description`	**Type** textarea **Properties** Create, Nillable, Update **Description** Description of this asset.
`InstallDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description Date on which this asset was installed.
`IsCompetitorProduct`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this Asset represents a product sold by a competitor (`true`) or not (`false`). Default value is `false`. Label is **Competitor Asset**.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of the asset. Label is **Asset Name**.
`Price`	**Type** currency **Properties** Create, Filter, Nillable, Sort, Update **Description** Price paid for this asset.
`Product2Id`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the Product2 associated with this asset. Must be a valid Product2 ID. Optional.
`PurchaseDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Date on which this asset was purchased.

Field	Details
Quantity	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Quantity purchased or installed.
SerialNumber	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Serial number for this asset.
Status	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Customizable picklist of values. The default picklist includes the following values: • Purchased • Shipped • Installed • Registered • Obsolete
UsageEndDate	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Date when usage for this asset ends or expires.

Usage

Use this object to track assets previously sold into customer accounts. With asset tracking, a client application can quickly determine which products were previously sold or are currently installed at a specific account.

For example, your organization might want to renew and up-sell opportunities on products sold in the past. Similarly, your organization might want to track competitive products that exist in a customer environment that could potentially be replaced or swapped out.

Asset tracking is also useful for product support, providing detailed information to assist with product-specific support issues. For example, the `PurchaseDate` or `SerialNumber` could indicate whether a given product has certain maintenance requirements, including product recalls. Similarly, the `UsageEndDate` might indicate when the asset was removed from service or when a license or warranty expires.

If an application creates a new Asset record, it must at least specify a `Name` and either an `AccountId`, `ContactId`, or both.

See Also:

Standard and Custom Object Basics

AssetFeed

Represents a single feed item in the feed displayed on the detail page for an asset record. An asset feed shows recent changes to an asset record for any fields that are tracked in feeds, as well as comments and posts about the record. It is a useful way to stay up-to-date with changes to assets. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the Asset object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field Name	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of the AssetFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort

Field Name	Details
	Description The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Setting AssetFeed automatically sets `Type` to `ContentPost`.

Field Name	Details
ContentSize	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
ContentType	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
FeedPostId	**Type** reference **Properties** Filter. Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. Use FeedItem instead. The ID of the associated FeedPost. A FeedPost represents the following types of changes in an AssetFeed: status updates, changes to tracked fields, text posts, link posts, and content posts.
InsertedById	**Type** reference **Properties** Group, Nillable, Sort **Description** The ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into the feed, `InsertedById` is set to the ID of the logged-in user.
LastModifiedDate	**Type** dateTime

Field Name	Details
	Properties Defaulted on create, Filter, Sort **Description** This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on the feed item, then `LastModifiedDate` becomes the `CreatedDate` for the FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of the `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities.

Field Name	Details
	Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the asset record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion article associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as aern AssetFeed object of `Type ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the AssetFeed. When the `Type` is `LinkPost`, the `LinkUrl` is the URL, and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field Name	Details
	Description The type of AssetFeed item: • ActivityEvent—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • AdvancedTextPost—created when a user posts a group announcement. • AnnouncementPost—Not used. • ApprovalPost—generated when a user submits an approval. • BasicTemplateFeedItem—Not used. • CanvasPost—a post made by a canvas app posts on a feed. • CollaborationGroupCreated—generated when a user creates a public group. • CollaborationGroupUnarchived—Not used. • ContentPost—a post with an attached file. • CreatedRecordEvent—generated when a user creates a record from the publisher. • DashboardComponentAlert—generated when a dashboard metric or gauge exceeds a user-defined threshold. • DashboardComponentSnapshot—created when a user posts a dashboard snapshot on a feed. • LinkPost—a post with an attached URL. • PollPost—a poll posted on a feed. • ProfileSkillPost—generated when a skill is added to a user's Chatter profile. • QuestionPost—generated when a user posts a question. • ReplyPost—generated when Chatter Answers posts a reply. • RypplePost—generated when a user creates a Thanks badge in Work.com. • TextPost—a direct text entry on a feed. • TrackedChange—a change or group of changes to a tracked field. • UserStatus—automatically generated when a user adds a post. Deprecated. The following values appear in the Type picklist for all feed objects but apply only to CaseFeed: • AttachArticleEvent—generated event when a user attaches an article to a case. • CallLogPost—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • CaseCommentPost—generated event when a user adds a case comment for a case object.

Field Name	Details
	• `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case.
`Visibility`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for an asset record.

See Also:

Asset
Product2
FeedItem

AssetTag

Associates a word or short phrase with an Asset.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.
`Type`	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

AssetTag stores the relationship between its parent TagDefinition and the Asset being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

AssignmentRule

Represents an assignment rule associated with a Case or Lead.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

- This object is read only. Assignment rules are created, configured, and deleted in the user interface.
- Customer Portal users can't access this object.

Fields

Field	Details
`Active`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this assignment rule is active (`true`) or not (`false`).
`Name`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Name of this assignment rule.
`SobjectType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Type of assignment rule—Case or Lead.

Usage

Before creating or updating a new Case or Lead, a client application can query (by name) the AssignmentRule to obtain the ID of the assignment rule to use, and then assign that ID to the `assignmentRuleId` field of the AssignmentRuleHeader. The AssignmentRuleHeader can be set using either SOAP API or REST API.

See Also:

Standard and Custom Object Basics

AsyncApexJob

Represents an individual Apex sharing recalculation job, a batch Apex job, or method with the `future` annotation.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ApexClassID`	**Type** reference, **Properties** Filter, Group, Sort **Description** The ID of the Apex class executing the job. Label is `Class ID`.
`CompletedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date and time when the job was completed.
`ExtendedStatus`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** If one or more errors occurred during the batch processing, this contains a short description of the first error. A more detailed description of that error, along with any subsequent errors, is emailed to the user who started the running batch class. This field is available in API version 19.0 and later.

Field Name	Details
JobItemsProcessed	**Type** int **Properties** Filter, Group, Sort **Description** Number of job items processed. Label is `Batches Processed`.
JobType	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The type of job being processed. Valid values are: • `Future` • `SharingRecalculation` • `ScheduledApex` • `BatchApex` • `BatchApexWorker` • `TestRequest` • `TestWorker` • `ApexToken`
MethodName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The name of the Apex method being executed. Label is `Apex Method`.
NumberOfErrors	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Total number of batches with a failure. A batch is considered transactional, so any unhandled exceptions constitute an entire failure of the batch. Label is `Failures`.
Status	**Type** picklist

Field Name	Details
	Properties Filter, Group, Restricted picklist, Sort **Description** The status of the job. Valid values are: • `Holding`[1] • `Queued` • `Preparing` • `Processing` • `Aborted` • `Completed` • `Failed` [1] This status is for the Apex Flex Queue pilot.
`TotalJobItems`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Total number of batches processed. Each batch contains a set of records. Label is `Total Batches`.

Usage

Use this object to query Apex batch jobs in your organization.

AttachedContentDocument

This read-only object contains all `ContentDocument` objects associated with an object.

Supported Calls

`describeSObjects()`

Fields

Field Name	Details
`ContentDocumentId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description ID of the attached `ContentDocument`.
`ContentSize`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Size of the document in bytes.
`ContentUrl`	**Type** url **Properties** Filter, Group, Nillable, Sort **Description** URL for links and Google Docs. This field is set only for links and Google Docs, and is one of the fields that determine the `FileType`. This field is available in API version 31.0 and later.
`FileExtension`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** File extension of the attached `ContentDocument`. This field is available in API version 31.0 and later.
`FileType`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Type of document, determined by the file extension.

Field Name	Details
LinkedEntityId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the record the `ContentDocument` is attached to.
Title	**Type** string **Properties** Filter, Group, Sort **Description** Title of the attached `ContentDocument`.

Usage

Use this object to list all `ContentDocument` objects attached to an object via a feed post.

To retrieve `ContentDocument` objects, issue a describe call on an object, which returns a query result for each activity since the record was created. You can't directly query this object.

Attachment

Represents a file that a User has uploaded and attached to a parent object.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
Body	**Type** base64 **Properties** Create, Update **Description** Required. Encoded file data.

Field	Details
BodyLength	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Size of the file (in bytes).
ConnectionReceivedId	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that shared this record with your organization. This field is only available if you have enabled Salesforce to Salesforce.
ConnectionSentId	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that you shared this record with. This field is only available if you have enabled Salesforce to Salesforce. Beginning with API version 15.0, the `ConnectionSentId` field is no longer supported. The `ConnectionSentId` field is still visible, but the value is null. You can use the new PartnerNetworkRecordConnection object to forward records to connections.
ContentType	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The content type of the attachment. If the `Don't allow HTML uploads as attachments or document records` security setting is enabled for your organization, you cannot upload files with the following file extensions: `.htm`, `.html`, `.htt`, `.htx`, `.mhtm`, `.mhtml`, `.shtm`, `.shtml`, `.acgi`, `.svg`.
Description	**Type** textarea

Field	Details
	Properties Create, Filter, Nillable, Sort, Update **Description** Description of the attachment. This field is available in API version 18.0 and later.
`IsPartnerShared`	**Type** boolean **Properties** Defaulted on create, Filter, Update **Description** Indicates whether this record is shared with a connection using Salesforce to Salesforce. Label is `Is Shared With Partner`.
`IsPrivate`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this record is viewable only by the owner and administrators (`true`) or viewable by all otherwise-allowed users (`false`). During a create or update call, it is possible to mark an Attachment record as private even if you are not the owner. This can result in a situation in which you can no longer access the record that you just inserted or updated. Label is **Private**. Attachments on tasks or events can't be marked private.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of the attached file. Label is **File Name**.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the User who owns the attachment. This field was required previous to release 9.0. Beginning with release 9.0, it can be null on create.

Field	Details
	The owner of an attachment on a task or event must be the same as the owner of the task or event.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the parent object of the attachment. The following objects are supported as parents of attachments: • Account • Asset • Campaign • Case • Contact • Contract • Custom objects • EmailMessage • EmailTemplate • Event • Lead • Opportunity • Product2 • Solution • Task

Usage

The API sends and receives the binary file attachment data encoded as a base64Binary data type. Prior to creating a record, client applications must encode the binary attachment data as base64. Upon receiving a response, client applications must decode the base64 data to binary (this conversion is usually handled for you by the SOAP client).

The create call restricts these files to a maximum size of 25 MB. For a file attached to a Solution, the limit is 1.5 MB. The maximum email attachment size is 3 MB.

The API supports attachments on email in create, delete, or update calls. The query call does not return attachments parented by email, unless the user performing the query has the “Modify All Data” permission.

Note:

- Attachment records are not searched during text searches.
- When issued by an administrator, the query results include Attachment records from the Recycle Bin.
- When issued by a non-administrator, the `queryAll()` call results do not include Attachment records from the Recycle Bin.

Access to fields depends on the method being used:

- All of the fields are accessible using the `describeSObjects()` and `query()` calls. With the `create()` call, you can insert the `Name`, `ParentId`, `Body`, `IsPrivate`, and `OwnerId` fields.
- To modify existing records, the `update()` call gives you access to change the `Name`, `Body`, `IsPrivate`, and `OwnerId` fields.
- You can access all of the fields using a `query()` call. However, you can't receive the `Body` field for multiple records in a single `query()` call. If your query returns the `Body` field, your client application must ensure that only one row with one Attachment is returned; otherwise, an error occurs. A more effective approach is to return IDs (but not Attachment records in the `Body` field) from a `query()` call and then pass them into `retrieve()` calls that return the `Body` field.
- For information about accessing the attachments of archived activities, see Archived Activities.

See Also:

Note

AuthSession

The AuthSession object represents an individual user session in your organization. This object is available in versions 29.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`LoginType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of login, for example, Application.
`NumSecondsValid`	**Type** int **Properties** Filter, Group, Sort **Description** The number of seconds before the session expires, starting from the last update time.
`SessionSecurityLevel`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field Name	Details
	Description Standard or High, depending upon the authentication method used.
SessionType	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of session. Common ones are UI, Content, API, and Visualforce.
SourceIp	**Type** string **Properties** Filter, Group, Sort **Description** IP address of the end user's device from which the session started. This can be an IPv4 or IPv6 address.
UserType	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The kind of user for this session. Types include Standard, Partner, Customer Portal Manager, High Volume Portal, and CSN Only.
UsersId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The user's Salesforce user ID.

Usage

The AuthSession object exposes session data and enables read and delete operations on that data. For example, use this object to create a report showing who is signed in to your organization, or to create a tool to delete a session, ending that user's session. For a user, only their own sessions are available, while administrators can see all sessions.

You can't change user sessions with this object; only read and delete them.

Bookmark

Represents a link between opportunities that share common information.

This object is available to organizations with the Similar Opportunities feature enabled.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`ID`	**Type** ID **Properties** Defaulted on create, Filter **Description** ID of the bookmark. Label is **Bookmark ID**.
`FromId`	**Type** ID **Properties** Filter **Description** The originating opportunity. Label is **Bookmarked From ID**
`ToId`	**Type** ID **Properties** Filter **Description** The opportunity to which the originating opportunity is linked. Label is **Bookmarked To ID**.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.

Usage

The Bookmark object works with the Opportunity object only.

Use this read-only object to query the bookmarks between opportunities in your organization. In the online application, users can search for opportunities that share attributes with their opportunity. The user can then bookmark the appropriate opportunities for future reference.

BrandTemplate

Letterhead for HTML EmailTemplate.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`Description`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** Description of the letterhead. Limited to 1000 characters.
`DeveloperName`	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Label is **Letterhead Unique Name**. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.

Field	Details
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the letterhead is available for use (`true`) or not (`false`). Label is **Active**.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label of the template as it appears in the user interface. Limited to 255 characters. Label is **Brand Template Name**.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects. This field can't be accessed unless the logged-in user has the “Customize Application” permission.
`Value`	**Type** textarea **Properties** Create, Update

Field	Details
	Description The contents of the letterhead, in HTML, including any logos.

Usage

Use this object to brand EmailTemplate records with your letterhead. You can also set a brand template to active or inactive. For example, if you have five different marketing brands, you can maintain each different brand in one template, and assign to the appropriate EmailTemplate.

See Also:

EmailTemplate

BusinessHours

This object is used to specify the business hours of your support organization. Escalation rules are run only during these hours. If business hours are associated with any Holiday records, then business hours and escalation rules associated with business hours are suspended during the dates and times specified as holidays.

Supported Calls

`create()`, `getUpdated()`, `query()`, `retrieve()`, `update()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`BusinessHoursId`	**Type** reference **Properties** Filter, Group, Nillable,Sort **Description** ID of the BusinessHours associated with the SlaProcess.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the business hours is active (`true`) or not active (`false`).

Field	Details
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The name of the business hours.
`IsDefault`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the business hours are set as the default business hours (`true`) or not (`false`).
`FridayEndTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business closes.
`FridayStartTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business opens.
`MondayEndTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business closes.
`MondayStartTime`	**Type** time

Field	Details
	Properties Create, Filter, Nillable, Sort, Update **Description** Time that business opens.
`SaturdayEndTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business closes.
`SaturdayStartTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business opens.
`SundayEndTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business closes.
`SundayStartTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business opens.
`ThursdayEndTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update

Field	Details
	Description Time that business closes.
`ThursdayStartTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business opens.
`TimeZoneSidKey`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The time zone of the business hours.
`TuesdayEndTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business closes.
`TuesdayStartTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business opens.
`WednesdayEndTime`	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business closes.

Field	Details
WednesdayStartTime	**Type** time **Properties** Create, Filter, Nillable, Sort, Update **Description** Time that business opens.

Usage

Use this object to specify the business hours at which your support team operates. Escalation rules only run during the business hours with which they are associated. To set business hours to 24-hours a day, set the times from midnight to midnight (00:00:00 ~ 00:00:00) on each day.

By default, business hours are set from 12:00 AM to 12:00 AM in the default time zone specified in your organization's profile.

See Also:

Standard and Custom Object Basics

BusinessProcess

Represents a business process.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users cannot access this object.

Fields

Field	Details
Description	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Description of this business process. Limit: 255 characters.
IsActive	**Type** boolean

Field	Details
	Properties Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this business process can be presented to users in the Salesforce user interface (`true`) or not (`false`) when creating a new record type or changing the business process of an existing record type.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of this business process. Limit: 80 characters.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.
`TableEnumOrId`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** Required. One of the following values: Case, Opportunity, or Solution. Label is **Entity Enumeration Or ID**.

Usage

Use the BusinessProcess object to offer different subsets of picklist values to different users for the LeadStatus, CaseStatus, and OpportunityStage fields. Similar to a RecordType, a BusinessProcess identifies the type of a row in a Case, Lead, or Opportunity and implies a subset of picklist values for these three fields. The values for the remaining picklist fields are driven off of RecordType.

See Also:

Standard and Custom Object Basics

CallCenter

This object represents a call center, which is a logical representation of a single computer-telephony integration (CTI) system instance in an organization.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`AdapterURL`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort **Description** An optional field that specifies the location of where the CTI adapter is hosted. For example, `http://localhost:11000`. This field is available for call centers using CTI Toolkit version 4.0 and API version 23.0 or later.
`CustomSettings`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies settings in the call center definition file, such as whether the call center uses the Open CTI, and SoftPhone properties, such as height in pixels. This field is available for Open CTI and in API version 25.0 or later.

Field	Details
Id	**Type** ID **Properties** Defaulted on create, Filter **Description** System field that uniquely identifies this call center. Label is **Call Center ID**. This ID is created automatically when the call center is created.
InternalName	**Type** string **Properties** Create, Filter, Group, Sort **Description** The internal name of the call center. Limit is 80 characters.
Name	**Type** string **Properties** Create, Filter, Group, Sort **Description** The name of the call center. Limit is 80 characters.
Version	**Type** double **Properties** Create, Filter, Nillable, Sort **Description** The version of the CTI Developer's Toolkit used to create the call center (for versions 2.0 and later). This field is available in API version 18.0 and later.

Usage

Create a call center or query an existing call center.

Campaign

Represents and tracks a marketing campaign, such as a direct mail promotion, webinar, or trade show.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`ActualCost`	**Type** currency **Properties** Create, Filter, Nillable, Sort, Update **Description** Amount of money spent to run the campaign.
`AmountAllOpportunities`	**Type** currency **Properties** Filter, Sort **Description** Amount of money in all opportunities associated with the campaign, including closed/won opportunities. Label is **Total Value Opportunities**.
`AmountWonOpportunities`	**Type** currency **Properties** Filter, Sort **Description** Amount of money in closed or won opportunities associated with the campaign. Label is **Total Value Won Opportunities**.
`BudgetedCost`	**Type** currency **Properties** Create, Filter, Nillable, Sort, Update

Field	Details
	Description Amount of money budgeted for the campaign.
`CampaignMemberRecordTypeId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The record type ID for CampaignMember records associated with the campaign.
`CurrencyIsoCode`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Nillable, Restricted picklist, Update **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.
`Description`	**Type** textarea **Properties** Create, Nillable, Update **Description** Description of the campaign. Limit: 32 KB. Only the first 255 characters display in reports.
`EndDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Ending date for the campaign. Responses received after this date are still counted.
`ExpectedResponse`	**Type** percent **Properties** Create, Filter, Nillable, Sort, Update **Description** Percentage of responses you expect to receive for the campaign.

Field	Details
ExpectedRevenue	**Type** currency **Properties** Create, Filter, Nillable, Sort, Update **Description** Amount of money you expect to generate from the campaign.
HierarchyActualCost	**Type** currency **Properties** Filter, Nillable, Sort **Description** Calculated field for the total amount of money spent to run the campaigns in a campaign hierarchy. Label is **Total Actual Cost in Hierarchy**.
HierarchyBudgetedCost	**Type** currency **Properties** Filter, Nillable, Sort **Description** Calculated field for the total amount of money budgeted for the campaigns in a campaign hierarchy. Label is**Total Budgeted Cost in Hierarchy**.
HierarchyExpectedRevenue	**Type** currency **Properties** Filter, Nillable, Sort **Description** Calculated field for the total amount of money you expect to generate from the campaigns in a campaign hierarchy. Label is **Total Expected Revenue in Hierarchy**.
HierarchyNumberSent	**Type** int **Properties** Filter **Description** Calculated field for the total number of individuals targeted by the campaigns in a campaign hierarchy. For example, the number of email messagess sent. Label is **Total Num Sent in Hierarchy**.

Field	Details
IsActive	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this campaign is active (`true`) or not (`false`). Default value is `false`. Label is **Active**.
LastActivityDate	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Value is one of the following, whichever is the most recent: • Due date of the most recent event logged against the record. • Due date of the most recently closed task associated with the record.
LastReferencedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
LastViewedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update

Field	Details
	Description Required. Name of the campaign. Limit: is 80 characters.
`NumberOfContacts`	**Type** int **Properties** Filter, Group, Sort **Description** Number of contacts associated with the campaign. Label is **Total Contacts**.
`NumberOfConvertedLeads`	**Type** int **Properties** Filter, Group, Sort **Description** Number of leads that were converted to an account and contact due to the marketing efforts in the campaign. Label is **Converted Leads**.
`NumberOfLeads`	**Type** int **Properties** Filter, Group, Sort **Description** Number of leads associated with the campaign. Label is **Total Leads**.
`NumberOfOpportunities`	**Type** int **Properties** Filter, Group, Sort **Description** Number of opportunities associated with the campaign. Label is **Num Total Opportunities**.
`NumberOfResponses`	**Type** int **Properties** Filter, Group, Sort

Field	Details
	Description Number of contacts and unconverted leads with a Member Status equivalent to “Responded” for the campaign. Label is **Total Responses**.
`NumberOfWonOpportunities`	**Type** int **Properties** Filter, Group, Sort **Description** Number of closed or won opportunities associated with the campaign. Label is **Num Won Opportunities**.
`NumberSent`	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Number of individuals targeted by the campaign. For example, the number of emails sent. Label is **Num Sent**.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the user who owns this campaign. Default value is the user logging in to the API to perform the create.
`ParentCampaign`	**Type** reference **Properties** Create, Filter Nillable, Update **Description** The campaign above the selected campaign in the campaign hierarchy.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description ID of the parent Campaign record, if any.
`RecordTypeId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the record type assigned to this object.
`StartDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Starting date for the campaign.
`Status`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Status of the campaign, for example, Planned, In Progress. Limit: 40 characters.
`TotalAmountAllOpportunities`	**Type** currency **Properties** Filter **Description** Calculated field for total amount of all opportunities associated with the campaign hierarchy, including closed/won opportunities. Label is **Total Value Opportunities in Hierarchy**.
`TotalAmountAllWonOpportunities`	**Type** currency **Properties** Filter

Field	Details
	Description Calculated field for amount of all closed/won opportunities associated with the campaign hierarchy. Label is **Total Value Won Opportunities in Hierarchy**.
`TotalNumberofContacts`	**Type** int **Properties** Filter **Description** Calculated field for number of contacts associated with the campaign hierarchy. Label is **Total Contacts in Hierarchy**.
`TotalNumberofConvertedLeads`	**Type** int **Properties** Filter **Description** Calculated field for the total number of leads associated with the campaign hierarchy that were converted into accounts, contacts, and opportunities. Label is **Total Converted Leads in Hierarchy**.
`TotalNumberofLeads`	**Type** int **Properties** Filter **Description** Calculated field for total number of leads associated with the campaign hierarchy. This number also includes converted leads. Label is **Total Leads in Hierarchy**.
`TotalNumberofOpportunties`	**Type** int **Properties** Filter **Description** Calculated field for the total number of opportunities associated with the campaign hierarchy. Label is **Total Opportunities in Hierarchy**.
`TotalNumberofResponses`	**Type** int

Field	Details
	Properties Filter **Description** Calculated field for number of contacts and unconverted leads that have a `Member Status` equivalent to “Responded” for the campaign hierarchy. Label is **Total Responses in Hierarchy**.
`TotalNumberofWonOpportunities`	**Type** int **Properties** Filter **Description** Calculated field for the total number of won opportunities associated with the campaign hierarchy. Label is **Total Won Opportunities in Hierarchy**.
`Type`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Type of campaign, for example, Direct Mail or Referral Program. Limit: 40 characters.

Usage

Client applications can create, update, delete, and query Attachment records associated with a campaign via the API.

The Campaign object is defined only for those organizations that have the marketing feature enabled and valid marketing licenses. In addition, it is accessible only to those users that are enabled as marketing users. If the organization does not have the marketing feature or valid marketing licenses, this object does not appear in the `describeGlobal()` call, and you can’t use `describeSObjects()` or `query()` with the Campaign object.

Note: The main constituent of a campaign is a CampaignMember. You will commonly need to update campaigns with CampaignMember.

See Also:

Standard and Custom Object Basics

CampaignFeed

Represents a single feed item in the feed on a campaign record detail page.

A campaign feed shows recent changes to a campaign record for any fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to campaigns. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the Campaign object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of CampaignFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64

Field	Details
	Properties Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.

Field	Details
CreatedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
FeedPostId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
InsertedById	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
LastModifiedDate	**Type** dateTime

Field	Details
	Properties Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`:

Field	Details
	• Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the campaign record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging).

Field	Details
	For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field	Details
	Description Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a campaign record.

See Also:

Campaign
EntitySubscription
NewsFeed
UserProfileFeed

CampaignMember

Represents the association between a Campaign and either a Lead or Contact.

Supported Calls

For API version 15.0 and earlier:`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

For API version 16.0 and later: `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`CampaignId`	**Type** reference

Field	Details
	Properties Create, Filter, Group, Sort **Description** Required. ID of the Campaign to which this Lead or Contact is associated.
`ContactId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** Required. ID of the Contact who is associated with a Campaign.
`CurrencyIsoCode`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Nillable, Restricted picklist, Update **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.
`FirstRespondedDate`	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Indicates the date on which the campaign member was first given a responded status.
`HasResponded`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the campaign member has responded to the campaign (`true`) or not (`false`). Label is **Responded**.
`LeadId`	**Type** reference **Properties** Create, Filter, Group, Nillable,Sort

Field	Details
	Description Required. ID of the Lead who is associated with a Campaign.
`RecordTypeId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the record type assigned to this object. To change the record type, modify the `CampaignMemberRecordTypeId` field on the associated Campaign.
`Status`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Controls the `HasResponded` flag on this object. You can't directly set the `HasResponded` flag, as it is read-only, but you can set it indirectly by setting this field in a create or update call. Each predefined value implies a `HasResponded` flag value. Each time you update this field, you implicitly update the `HasResponded` flag. In the Salesforce user interface, Marketing users can define valid status values for the `Status` picklist. They can choose one status as the default status. For each `Status` field value, they can also select which values should be counted as "Responded," meaning that the `HasResponded` flag will be set to `true` for those values. 40 character limit. **Note:** When creating or updating campaign members, use the text value for `Status` instead of the ID from the CampaignMemberStatus object.

Note: If you are importing CampaignMember data into Salesforce and need to set the value for an audit field, such as `CreatedDate`, contact salesforce.com. Audit fields are automatically updated during API operations unless you request to set these fields yourself..

Usage

Each record has a unique ID, and must contain either a `ContactId` or a `LeadId`, but can't contain both. Any attempt to create a single record with both results in a successful insert but only the `ContactId` will be inserted. However, you can create two separate records on a Campaign—one for the Lead and one for the Contact.

Standard fields from a Contact or Lead are associated with the CampaignMember object but you can't query them directly. To include a lead's `Phone` in your query, for example, query the field from the Lead object.

```
SELECT Id, (SELECT Phone FROM Lead)
FROM CampaignMember
```

This object is defined only for those organizations that have the marketing feature and valid marketing licenses. In addition, the object is accessible only to those users that are enabled as marketing users. If the organization does not have the marketing feature or valid marketing licenses, this object does not appear in the `describeGlobal()` call, and you can't use `describeSObjects()` or `query()` with the CampaignMember object.

In API version 16.0 and later, a `create()` call only creates a new record; in earlier versions, a `create()` call creates and updates records. The API determines whether a record exists with the specified `CampaignId` and either `ContactId` or `LeadId`.

Note: Only use a `ContactId` or `LeadId`, but not both, unless you want to track lead-based campaign members you convert to contacts.

If the record does not exist for the given `ContactId` or `LeadId`, then a new record is created. If the record exists, an error is returned and no update is made. To update an existing record, specify the ID of the CampaignMember record to update.

In API versions 15.0 and earlier, if you submit multiple records using a single create request, and if more than one record matches an existing record, only the first record submitted updates the existing record. If any of the submitted records match each other but do not match existing records, only the last record submitted is created.

The `upsert()` call is not supported in API version 16.0 and later. To use the `upsert()` call on this object, you must first delete all data in ID fields except the record ID.

To delete a record, specify the ID of the CampaignMember record to delete.

When creating or updating records, the `Status` field value specified in the call is verified as a valid status for the given Campaign:

- If the specified `Status` value is a valid status, the value is updated, and the `HasResponded` field is updated to either `true` or `false`, depending on the `Status` value association with `HasResponded`.
- If the specified `Status` value is not a valid status, the API assigns the default status to the `Status` field and updates the `HasResponded` field with the associated value. However, if the given Campaign does not have a default status, the API assigns the value specified in the call to the `Status` field, and the `HasResponded` field is set to `false`.

See Also:

Campaign
CampaignMemberStatus

CampaignMemberStatus

One or more member status values defined for a campaign.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

You can't delete a CampaignMemberStatus if that status is designated as the default status or if the status is currently used in a Campaign.

Fields

Field	Details
CampaignId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the campaign associated with this member status.
HasResponded	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this status is equivalent to "Responded" (`true`) or not (`false`).
IsDefault	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this status is the default status (`true`) or not (`false`).
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
Label	**Type** string **Properties** Create, Filter, Sort **Description** Label for the status in the picklist. Limited to 765 characters.

Field	Details
SortOrder	**Type** int **Properties** Create, Filter, Group, Sort, Update **Description** Order where this campaign member status appears in the picklist.

Usage

Use this object to create picklist items for the member status in a campaign.

This object is defined only for those organizations that have the marketing feature and valid marketing licenses. In addition, the object is accessible only to those users that are enabled as marketing users. If the organization does not have the marketing feature or valid marketing licenses, this object does not appear in a `describeGlobal()` call, and you can't use `describeSObjects()` or `query()` with the CampaignMember object.

See Also:

Campaign
CampaignMember

CampaignOwnerSharingRule

Represents the rules for sharing a campaign with User records other than the owner or anyone above the owner in the role hierarchy.

Note: Contact salesforce.com customer support to enable access to this object for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
CampaignAccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of access granted to the target Group, or UserRole. The possible values are:

Field	Details
	• `Read` • `Edit` • `All`
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
`DeveloperName`	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Rule Name** in the user interface. This field is available in API version 24.0 and later. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`GroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the source group. ACampaign owned by a User in the source Group triggers the rule to give access.
`Name`	**Type** string

Field	Details
	Properties Create, Filter, Group, Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to **Label** on the user interface.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the User or Group being granted access.

Usage

Use this object to manage the sharing rules for campaigns.

CampaignShare

Represents a sharing entry on a Campaign.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`CampaignId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Campaign associated with this sharing entry. This field can't be updated.
`CampaignAccessLevel`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort

Field	Details
	Description Level of access that the User or Group has to the Campaign. The possible values are: • `Read` • `Edit` • `All` (This value is not valid for creating or updating records.) This field must be set to an access level that is higher than the organization's default access level for Campaign.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. There are many possible values, including: • `Campaign Sharing Rule`—The User or Group has access via a Campaign sharing rule. • `Manual Sharing`Manual Sharing—The User or Group has access because a User with "`All`" access manually shared the Campaign with them. • `Owner`—The User is the owner of the Campaign or is in a Role above the Campaign owner in the role hierarchy.
`UserOrGroupId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the User or Group that has been given access to the Campaign. This field can't be updated.

Usage

This object allows you to determine which users and groups can view or edit Campaign records owned by other users.

CampaignTag

Associates a word or short phrase with a Campaign.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.
`Type`	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

CampaignTag stores the relationship between its parent TagDefinition and the Campaign being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

Case

Represents a case, which is a customer issue or problem.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`AccountId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the account associated with this case.
`CaseNumber`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** Assigned automatically when each case is inserted. It can't be set directly, and it can't be modified after the case is created.
`ClosedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date and time when the case was closed.
`CommunityId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description ID of the Community (Zone) associated with this case. This field is available in API version 24.0 and later.
`ConnectionReceivedId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that shared this record with your organization. This field is only available if you have enabled Salesforce to Salesforce.
`ConnectionSentId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that you shared this record with. This field is only available if you have enabled Salesforce to Salesforce. Beginning with API version 15.0, the `ConnectionSentId` field is no longer supported. The `ConnectionSentId` field is still visible, but the value is null. You can use the new PartnerNetworkRecordConnection object to forward records to connections.
`ContactId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the associated Contact.
`CreatorFullPhotoUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** URL of the user's profile photo from the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.

Field	Details
CreatorName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Name of the user who posted the question or reply. Only the first name of internal users (agents) appears to portal users in the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.
CreatorSmallPhotoUrl	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** URL of the user's thumbnail photo from the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.
Description	**Type** textarea **Properties** Create, Nillable, Update **Description** A text description of the case. Limit: 32 KB.
HasCommentsUnreadByOwner	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether a case has comments that have not yet been read by the owner (`true`) or not (`false`).
HasSelfServiceComments	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates whether a case has comments added by a Self-Service user (`true`) or not (`false`).
`IsClosed`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the case is closed (`true`) or open (`false`). This field is controlled by the `Status` field; it can't be set directly. Label is **Closed**.
`IsClosedOnCreate`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the case was closed at the same time that it was created (`true`) or not (`false`). This flag is read-only and is automatically set when a record is created. It can't be set to `true` unless the `IsClosed` flag is also `true`.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsEscalated`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the case has been escalated (`true`) or not. A case's escalated state does not affect how you can use a case, or whether you can query, delete, or update it. However, you can't set this flag via the API. Label is **Escalated**.
`IsSelfServiceClosed`	**Type** boolean

Field	Details
	Properties Defaulted on create, Filter, Group, Sort **Description** Indicates whether the case is closed for Self-Service users (`true`) or not (`false`).
`IsStopped`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether an entitlement process on a case is stopped (`true`) or not (`false`).
`IsVisibleInSelfService`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the case can be viewed in the Customer Service Portal, Partner Service Portal, and Self-Service Portal (`true`) or not (`false`). This field is applied for case visibility in the Partner Relationship Management, Customer Service Portal, and the earlier version of Self Service Portal. The field does not alter sharing and will not prevent usage of a direct URL to a case if a portal user has read or write access.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.

Field	Details
`Origin`	**Type** picklist **Properties** Create, Filter, Group, Nillable,Sort, Update **Description** The source of the case, such as `Email`, `Phone`, or `Web`. Label is **Case Origin**.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the contact who owns the case.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the parent case in the hierarchy. The label is **Parent Case**.
`Priority`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The importance or urgency of the case, such as `High`, `Medium`, or `Low`.
`QuestionId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The question in the answers community that is associated with the case. This field does not appear if you don't have an answers community enabled.
`Reason`	**Type** picklist

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The reason why the case was created, such as `Instructions not clear`, or `User didn't attend training`.
`RecordTypeId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the record type assigned to this object.
`SlaStartDate`	**Type** dateTime **Properties** Create, Filter, Nillable, Update **Description** Shows the time the case entered an entitlement process. You can update or reset the time if you have the "Edit" permission on cases. This field is available in API version 18.0 and later.
`Status`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Sort, Update **Description** The status of the case, such as "New," "Closed," or "Escalated." This field directly controls the `IsClosed` flag. Each predefined `Status` value implies an `IsClosed` flag value. For more information, see CaseStatus.
`StopStartDate`	**Type** dateTime **Properties** Filter, Nillable **Description** The date and time an entitlement process was stopped on the case. This field is available in API version 18.0 and later.

Field	Details
`Subject`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The subject of the case. Limit: 255 characters.
`SuppliedCompany`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The company name that was entered when the case was created. can't be updated after the case has been created. Label is **Company**.
`SuppliedEmail`	**Type** email **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The email address that was entered when the case was created. can't be updated after the case has been created. Label is **Email**. If your organization has an active auto-response rule, `SuppliedEmail` is required when creating a case via the API. Auto-response rules use the email in the contact specified by `ContactId`. If no email address is in the contact record, the email specified here is used.
`SuppliedName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The name that was entered when the case was created. can't be updated after the case has been created. Label is **Name**.
`SuppliedPhone`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description The phone number that was entered when the case was created. can't be updated after the case has been created. Label is **Phone**.
`Type`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The type of case, such as `Feature Request` or `Question`.

Note: If you are importing Case data and need to set the value for an audit field, such as `CreatedDate`, contact salesforce.com. Audit fields are automatically updated during API operations unless you request to set these fields yourself.

Usage

Use the Case object to manage cases for your organization. Client applications can query, update, and delete Attachment records associated with a case via the API.

Assignment Rules

When you query or update a case, your client application can have the case automatically assigned to one or more User records based on assignment rules that have been configured in the user interface. To use this feature, your client application needs to set either of the following options (but not both) in the AssignmentRuleHeader used in the create or update:

Field	Field Type	Details
`assignmentRuleId`	reference	ID of the assignment rule to use. Can be an inactive assignment rule. If unspecified and `useDefaultRule` is `true`, then the default assignment rule is used. To find the ID for a given assignment rule, query the AssignmentRule object (specifying `RuleType="caseAssignment"`), iterate through the returned AssignmentRule objects, find the one you want to use, retrieve its ID, and then specify its ID in this field in the AssignmentRuleHeader.
`useDefaultRule`	boolean	Specifies whether to use the default rule for rule-based assignment (`true`) or not (`false`). The default rule is assigned by users in the Salesforce user interface.

For a code example that shows setting the AssignmentRuleHeader for a Lead (which is similar to setting the AssignmentRuleHeader for a Case), see Lead.

Separating Accounts from Contacts in Cases

In releases before 8.0, the `AccountId` could not be specified, it was derived from the contact's account. This behavior will continue to be supported in future releases, but you can also now specify an `AccountId`. If you do not specify the `AccountId` during the creation of a case, the value will default to the contact's `AccountId`.

Note: When a record is updated, if the `ContactId` has not changed, then the `AccountId` is not regenerated. This prevents the API from overwriting a value previously changed in the Salesforce user interface. However, if an API call changes the `ContactId` and the `AccountId` field is empty, then the `AccountId` is generated using the contact's account.

Using `_case` with Java

Depending on the development tool you use, you may need to write your application using `_case` instead of `Case`. This is because `case` is a reserved word in Java.

See Also:

Account
CaseMilestone

CaseArticle

Represents the association between a Case and a KnowledgeArticle.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Access to this object is controlled by the parent Case and KnowledgeArticle. However, when querying, access is only controlled by the parent Case.

Customer Portal users can't access this object.

Fields

Field	Details
`ArticleLanguage`	**Type** picklist **Properties** Filter, Restricted picklist **Description** The language of the article associated with the case.
`ArticleVersionNumber`	**Type** int **Properties** Create, Group, Nillable **Description** The number assigned to a version of an article. This field is available in API version 24.0 and later.

Field	Details
CaseId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the Case associated with the KnowledgeArticle.
IsSharedByEmail	**Type** int **Properties** Create, Group, Nillable **Description** Indicates that the article has been shared with the customer through an email.
KnowledgeArticleId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the KnowledgeArticle associated with the Case.

Usage

This object lets you send emails with case-article associations and include case-article associations in Apex and Visualforce.

You can't update this object via the API. If you attempt to create a record that matches an existing record, the create request simply returns the existing record.

See Also:

Case
KnowledgeArticle

CaseComment

Represents a comment that provides additional information about the associated Case.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`CommentBody`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Text of the CaseComment. The maximum size of the comment body is 4,000 bytes. Label is **Body**.
`ConnectionReceivedId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that shared this record with your organization. This field is only available if you have enabled Salesforce to Salesforce.
`ConnectionSentId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that you shared this record with. This field is only available if you have enabled Salesforce to Salesforce. Beginning with API version 15.0, the `ConnectionSentId` field is no longer supported. The `ConnectionSentId` field is still visible, but the value is null. You can use the new PartnerNetworkRecordConnection object to forward records to connections.
`CreatorFullPhotoUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** URL of the user’s profile photo from the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.
`CreatorName`	**Type** string

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** Name of the user who posted the question or reply. Only the first name of internal users (agents) appears to portal users in the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.
`CreatorSmallPhotoUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** URL of the user's thumbnail photo from the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsPublished`	**Type** boolean **Properties** Defaulted on create, Create, Filter, Group, Sort, Update **Description** Indicates whether the CaseComment is visible to customers in the Self-Service portal (`true`) or not (`false`). Label is **Published**. This is the only CaseComment field that can be updated via the API.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort, **Description** Required. ID of the parent Case of the CaseComment.

Note: If you are importing CaseComment data and need to set the value for an audit field, such as `CreatedDate`, contact salesforce.com. Audit fields are automatically updated during API operations unless you request to set these fields yourself..

Usage

In the Salesforce user interface, comments are generally entered by a User working on a Case. All users have access to create and view CaseComment in the Salesforce user interface and when using the API. In the API, CaseComment records can't be modified after insertion unless the user has the “Modify All” object-level permission for Cases or the “Modify All Data” permission. If not, users can only update the `IsPublished` field, and can't delete CaseComment.

See Also:

Standard and Custom Object Basics

CaseContactRole

Represents the role that a given Contact plays on a Case.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`CasesId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the cases associated with this contact.
`ContactId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** Required. ID of the contact.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter

Field	Details
	Description Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`Role`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Name of the role played by the contact on this contract, such as Decision Maker, Approver, Buyer, and so on. Must be unique—there can't be multiple records in which the `ContractId`, `ContactId`, and `Role` values are identical. Different contacts can play the same role on the same contract. A contact can play different roles on the same contract.

Usage

Use this object to define the role that a given Case plays on a given Contact. For example, you can use this object to be able to see all contacts who are associated to a case, or, given a contact, be able to query all cases that they are associated with, even if they are not the primary contact on the case.

CaseFeed

Represents a single feed item in the feed displayed on the detail page for a case record. A case feed shows recent changes to a case record for any fields that are tracked in feeds, as well as comments and posts about the record. It is a useful way to stay up-to-date with changes to cases. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn’t create, you must have one of these permissions:

- “Modify All Data”
- “Modify All” on the Case object
- “Moderate Chatter”

Note: Users with the “Moderate Chatter” permission can delete only the feed items and comments they see.

Fields

Field	Details
Body	**Type** textarea **Properties** Nillable, Sort **Description** The content of the CaseFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
CommentCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
ConnectionId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
ContentData	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
ContentDescription	**Type** textarea

Field	Details
	Properties Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`FeedPostId`	**Type** reference **Properties** Filter. Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. Use FeedItem instead.

Field	Details
	The ID of the associated FeedPost. A FeedPost represents the following types of changes in a CaseFeed: status updates, changes to tracked fields, text posts, link posts, and content posts. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`InsertById`	**Type** reference **Properties** Group, Nillable, Sort **Description** The ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into the feed, `InsertById` is set to the ID of the logged-in user.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on the feed item, then `LastModifiedDate` becomes the `CreatedDate` for the FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkURL`	**Type** url **Properties** Nillable, Sort

Field	Details
	Description The URL of the `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the case record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordID`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion article associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing

Field	Details
	ContentVersion and post it to a feed as a CaseFeed object of `Type` `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the CaseFeed. When the `Type` is `LinkPost`, the `LinkURL` is the URL, and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of CaseFeed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed.

Field	Details
	• `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case.
`Visibility`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`.

Field	Details
	• On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a case record.

See Also:

Case
EntitySubscription
NewsFeed
FeedItem
UserProfileFeed

CaseHistory

Represents historical information about changes that have been made to the associated Case.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

This object is always read-only.

Fields

Field	Details
`CaseId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Case associated with this record.
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort

Field	Details
	Description Name of the case field that was modified, or a special value to indicate some other modification to the case. The possible values, in addition to the case field names, are: • **ownerAssignment**—The owner of the case was changed. • **ownerAccepted**—A user took ownership of a case from a queue. • **ownerEscalated**—The owner of the case was changed due to case escalation. • **external**—A user made the case visible to customers in the Customer Self-Service Portal.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** New value of the modified case field. Maximum of 255 characters.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** Previous value of the modified case field. Maximum of 255 characters.

Usage

Case history entries are indirectly created each time a case is modified.

Two rows are added to this record when foreign key fields change. One row contains the foreign key object names that display in the online application. For example, `Jane Doe` is recorded as the name of a Contact. The other row contains the actual foreign key ID that is only returned to and visible from the API.

This object respects field level security on the parent object.

See Also:

Standard and Custom Object Basics

CaseMilestone

Represents a milestone (required step in a customer support process) on a Case.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`, `update()`

Fields

Field	Details
`BusinessHoursId`	**Type** reference **Properties** Filter, Group, Nillable,Sort **Description** ID of the BusinessHours associated with the CaseMilestone.
`CaseId`	**Type** reference **Properties** Filter **Description** ID of the case.
`CompletionDate`	**Type** dateTime **Properties** Filter, Nillable, Update **Description** The date and time the milestone was completed.
`ElapsedTimeInDays`	**Type** double **Properties** Filter, Nillable

Field	Details
	Description The time required to complete a milestone in days.
`ElapsedTimeInHrs`	**Type** double **Properties** Filter, Nillable **Description** The time required to complete a milestone in hours.
`ElapsedTimeInMins`	**Type** int **Properties** Filter, Nillable **Description** The time required to complete a milestone in minutes.
`IsCompleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the milestone is completed (`true`) or not (`false`).
`IsViolated`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the milestone is violated (`true`) or not (`false`).
`MilestoneTypeId`	**Type** reference **Properties** Filter, Nillable **Description** The ID of the milestone on the case.

Field	Details
StartDate	**Type** dateTime **Properties** Filter, Nillable, Update **Description** The date and time the milestone started on the case.
TargetDate	**Type** dateTime **Properties** Filter **Description** The date and time the milestone must be completed.
TargetResponseInDays	**Type** double **Properties** Filter, Nillable **Description** The time to complete the milestone in days.
TargetResponseInHrs	**Type** double **Properties** Filter, Nillable **Description** The time to complete the milestone in hours.
TargetResponseInMins	**Type** int **Properties** Filter, Nillable **Description** The time to complete the milestone in minutes.
TimeSinceTargetInMins	**Type** text

Field	Details
	Properties Nillable **Description** The time elapsed since the milestone target. The format is minutes and seconds.
`TimeRemainingInMins`	**Type** text **Properties** Nillable **Description** Time remaining to reach the milestone target. The format is minutes and seconds.

Usage

This object lets you view a milestone on a case. It also lets you view if the milestone was completed and when it must be completed.

See Also:

Case
MilestoneType
SlaProcess

CaseOwnerSharingRule

Represents the rules for sharing a case with users other than the owner.

Note: Contact salesforce.com customer support to enable access to this object for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
CaseAccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of sharing being allowed. The possible values are: • `Read` • `Edit`
Description	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
DeveloperName	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Rule Name** in the user interface. This field is available in API version 24.0 and later. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
GroupId	**Type** reference

Field	Details
	Properties Create, Filter, Group, Sort **Description** The ID representing the source group. Cases owned by users in the source group trigger the rule to give access.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to **Label** on the user interface.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the target user or group. Target users or groups are given access.

Usage

Use this object to manage the sharing rules for cases. General sharing and territory management-related sharing use this object.

See Also:

Case

CaseShare

CaseShare

Represents a sharing entry on a Case.

Supported Calls

`describeSObjects()`, `create()`, `delete()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can’t access this object.

Fields

Field	Details
CaseAccessLevel	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Level of access that the User or Group has to the Case. The possible values are: • Read • Edit • All This value is not valid for creating or deleting records. This field must be set to an access level that is higher than the organization's default access level for cases.
CaseId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Case associated with this sharing entry. This field can't be updated.
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (true) or not (false). Label is **Deleted**.
RowCause	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Values may include: • Manual—The User or Group has access because a user with "All" access manually shared the Case with them. • Owner—The User is the owner of the Case or is in a Role above the Case owner in the role hierarchy.

Field	Details
	• `ImplicitChild`—User or Group has access to the Case on the Account associated with this Case. • `Rule`—The User or Group has access via a Case sharing rule.
`UserOrGroupId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the User or Group that has been given access to the Case. This field can't be updated.

Usage

This object allows you to determine which users and groups can view and edit Case records owned by other users.

If you attempt to create a new record that matches an existing record, request updates any modified fields and returns the existing record.

See Also:

AccountShare
LeadShare
OpportunityShare

CaseSolution

Represents the association between a Case and a Solution.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`CaseId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Case associated with the Solution.

Field	Details
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
SolutionId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Solution associated with the case.

Usage

You can't update this object via the API. If you attempt to create a record that matches an existing record, the request simply returns the existing record.

See Also:

CaseShare
SolutionStatus

CaseStatus

Represents the status of a Case, such as New, On Hold, or In Process.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
IsClosed	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates whether this case status value represents a closed Case (`true`) or not (`false`). Multiple case status values can represent a closed Case.
`IsDefault`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this is the default case status value (`true`) or not (`false`) in the picklist.
`MasterLabel`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Master label for this case status value. This display value is the internal label that does not get translated.
`SortOrder`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Number used to sort this value in the case status picklist. These numbers are not guaranteed to be sequential, as some previous case status values might have been deleted.

Usage

This object represents a value in the case status picklist. The case status picklist provides additional information about the status of a Case, such as whether a given `Status` value represents an open or closed case. Query the CaseStatus object to retrieve the set of values in the case status picklist, and then use that information while processing Case records to determine more information about a given case. For example, the application could test whether a given case is open or closed based on its `Status` value and the value of the `IsClosed` property in the associated CaseStatus object.

See Also:

Standard and Custom Object Basics

CaseTag

Associates a word or short phrase with a Case

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.
`Type`	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag.

Field Name	Details
	Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

CaseTag stores the relationship between its parent TagDefinition and the Case being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

CaseTeamMember

Represents a case team member, who works with a team of other users to help resolve a case.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`MemberId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the user or contact who is a member on a case team.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the case with which the case team member is associated.
`TeamRoleId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update

Field	Details
	Description The ID of the case team role with which the case team member is associated.
`TeamTemplateMemberId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the team member included in a predefined case team.

CaseTeamRole

Represents a case team role. Every case team member has a role on a case, such as “Customer Contact” or “Case Manager.”

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of access granted to the target Group for cases. The possible values are: • `None` • `Read` • `Edit`
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The name of the case team role.

Field	Details
PreferencesVisibleInCSP	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates whether or not the case team role is visible to Customer Portal users.

CaseTeamTemplate

Represents a predefined case team, which is a group of users that helps resolve a case.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
Description	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A text description of the predefined case team.
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The name of the predefined case team.

CaseTeamTemplateMember

Represents a member on a predefined case team, which is a group of users that helps resolve cases.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`MemberId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the user or contact who is a team member on a predefined case team.
`TeamRoleId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the predefined case team member's case team role.
`TeamTemplateId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the predefined case team's template.

CaseTeamTemplateRecord

The CaseTeamTemplateRecord object is a linking object between the Case and CaseTeamTemplate objects. To assign a predefined case team to a case (customer inquiry), create a CaseTeamTemplateRecord record and point the `ParentId` to the case and the `TeamTemplateId` to the predefined case team.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the case with which the case team template record is associated.
TeamTemplateId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the predefined case team with which the case team template record is associated.

CategoryData

Represents a logical grouping of Solution records.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
CategoryNodeId	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the CategoryNode associated with the solution.

Field	Details
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`RelatedSobjectId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the solution related to the category.

Usage

This object allows you to assign one or more categories to a Solution. It is an intermediate data table with two foreign keys that defines the relationship between a CategoryNode and a Solution record.

CategoryData has two foreign keys:

- The first foreign key, `CategoryNodeId`, refers to the ID of a CategoryNode.
- The other foreign key, `RelatedSobjectId`, refers to a Solution ID.

This is a many-to-many relationship, so there can be multiple rows returned with a `CategoryNodeId`. A Solution can be associated with multiple categories.

See Also:

Standard and Custom Object Basics

CategoryNode

Represents a tree of Solution categories.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- Customer Portal users can't access this object.
- Attempting to delete a CategoryNode that has children (referred by CategoryNode.Parent), or is referred to elsewhere, causes a failure.

Fields

Field	Details
MasterLabel	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label for the category node.
ParentId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the parent of this node, if any.
SortOrder	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Indicates the sort order of child CategoryNode objects.
SortStyle	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Indicates whether the sort order is alphabetical or custom.

Usage

A CategoryNode defines a category of solutions. In the user interface, you can edit category definitions from Setup by clicking **Customize** > **Solution** > **Solution Categories**.

See Also:

CategoryData
Solution

CategoryNodeLocalization

When the Translation Workbench is enabled for your organization, the CategoryNodeLocalization object provides the translation of the label of a solution category.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- Your organization must be using Professional, Enterprise, Developer, Unlimited, or Performance Edition and be enabled for the Translation Workbench.
- To view this object, you must have the "View Setup and Configuration" permission.

Fields

Field	Details
`CategoryNodeId`	**Type** reference **Properties** Create, Filter, Nillable **Description** The ID of the solution CategoryNode that is being translated.
`LanguageLocaleKey`	**Type** picklist **Properties** Create, Filter, Nillable, Restricted picklist **Description** This field is available in API version 16.0 and earlier. It is the same as the `Language` field.
`Language`	**Type** picklist **Properties** Create, Filter, Nillable, Restricted picklist **Description** This field is available in API version 17.0 and later. The combined language and locale ISO code, which controls the language for labels displayed in an application.

Field	Details
	This picklist contains the following fully-supported languages: • Chinese (Simplified): `zh_CN` • Chinese (Traditional): `zh_TW` • Danish: `da` • Dutch: `nl_NL` • English: `en_US` • Finnish: `fi` • French: `fr` • German: `de` • Italian: `it` • Japanese: `ja` • Korean: `ko` • Portuguese (Brazil): `pt_BR` • Russian: `ru` • Spanish: `es` • Spanish (Mexico): `es_MX`* * Spanish (Mexico) falls back to Spanish for customer-defined translations. • Swedish: `sv` • Thai: `th`* * Even though the Salesforce interface is fully translated to Thai, Help remains in English. The following end-user only languages are available. • Arabic: `ar` • Bulgarian: `bg` • Czech: `cs` • English (UK): `en_GB` • Greek: `el` • Hebrew: `iw` • Hungarian: `hu` • Indonesian: `in` • Norwegian: `no` • Polish: `pl` • Romanian: `ro` • Turkish: `tr` • Ukrainian: `uk` • Vietnamese: `vi` The following platform languages are available for organizations that use Salesforce exclusively as a platform. • Albanian: `sq` • Armenian: `hy` • Basque: `eu` • Bosnian: `bs` • Bengali: `bn`

Field	Details
	• Croatian: `hr` • English (Australia): `en_AU` • English (Canada): `en_CA` • English (India): `en_IN` • English (Malaysia): `en_MY` • English (Philippines): `en_PH` • Estonian: `et` • French (Canada): `fr_CA` • Georgian: `ka` • German (Austria): `de_AT` • German (Switzerland): `de_CH` • Hindi: `hi` • Icelandic: `is` • Irish: `ga` • Latvian: `lv` • Lithuanian: `lt` • Luxembourgish: `lb` • Macedonian: `mk` • Malay: `ms` • Maltese: `mt` • Romanian (Moldova): `ro_MD` • Montenegrin: `sh_ME` • Portuguese (European): `pt_PT` • Romansh: `rm` • Serbian (Cyrillic): `sr` • Serbian (Latin): `sh` • Slovak: `sk` • Slovenian: `sl` • Tagalog: `tl` • Tamil: `ta` • Urdu: `ur` • Welsh: `cy` The values in this field are not related to the default locale selection.
`NamespacePrefix`	**Type** string **Properties** Filter, Nillable **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values:

Field	Details
	• In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.
`Value`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** The actual translated label for the solution category. Label is **Translation**.

Usage

Use this object to translate the labels of your solution categories into a supported language. Users with the Translation Workbench enabled can view category node translations, but either the "Customize Application," "Manage Translation," or "Manage Categories" permission is required to create or update category node translations.

See Also:

ScontrolLocalization
WebLinkLocalization

ChatterActivity

ChatterActivity represents the number of posts and comments made by a user and the number of comments and likes on posts and comments received by the same user. This object is available in API version 23.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`CommentCount`	**Type** int **Properties** Filter, Group, Sort

Field Name	Details
	Description The number of FeedComments made by the ParentId.
`CommentReceivedCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments received by the ParentId.
`InfluenceRawRank`	**Type** int **Properties** Filter, Group, Sort **Description** Number indicating the ParentId's Chatter influence rank, which is calculated based on the ParentId's ChatterActivity statistics, relative to the other users in the organization. This field is available in API version 26.0 and later.
`LikeReceivedCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes received by the ParentId.
`NetworkId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Identifier of the community to which the ChatterActivity belongs. This field is available only if Salesforce Communities is enabled in your organization. This field is available in API version 26.0 and later.
`ParentId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description Identifier of the object type to which the ChatterActivity is related. In API version 31.0, the `ParentId` must be a `UserId` or SelfServiceUser ID.
`PostCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedItems made by the ParentId.

Usage

- Use this object to reference the Chatter activity statistics, which include the number of posts and comments made by a user and the number of comments and likes on posts and comments received by the same user.
- You can directly query for ChatterActivity.

```
SELECT Id, PostCount, LikeReceivedCount
FROM ChatterActivity
WHERE ParentId = UserId
```

Note: To query ChatterActivity, you must provide the `ParentId`. In API version 31.0, the `ParentId` must be a `UserId` or SelfServiceUser ID.

- A ChatterActivity record is created for users the first time they post or comment. Users who have never posted or commented don't have ChatterActivity records. If users make only one post and then delete it, they do have ChatterActivity records. In both cases, the user interface displays zeros for their Chatter activity.
- Use the `InfluenceRawRank` field to reference a user's Chatter influence rank. This field is available in API version 26.0 and later.

See Also:

FeedItem
FeedComment
FeedLike

ChatterAnswersActivity

Represents the reputation of a User in Chatter Answers communities. This object is available in API version 25.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`BestAnswerReceivedCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of best answers the User has received from other users.
`BestAnswerSelectedCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of best answers the User has selected.
`QuestionsCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of Question records posted by the User.
`QuestionSubscrCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of Question records the User has selected to follow.
`QuestionSubscrReceivedCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of users following Question records posted by the User.

Field Name	Details
`QuestionUpVotesCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of up votes the User has marked on Question records posted by other users.
`QuestionUpVotesReceivedCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of up votes the User has received from other users on the Question records he or she has posted.
`RepliesCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of Reply records posted by the User.
`ReplyDownVotesCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of down votes the User has marked on Reply records posted by other users.
`ReplyDownVotesReceivedCount`	**Type** int **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description The number of down votes the User has received from other users on the Reply records he or she has posted.
ReplyUpVotesCount	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of up votes the User has marked on the Reply records posted by other users.
ReplyUpVotesReceivedCount	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of up votes the User has received from other users on the Reply records he or she has posted.
ReportAbuseOnQuestionsCount	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of abuses that the User has reported on Question records posted by other users.
ReportAbuseOnRepliesCount	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of abuses that the User has reported on Reply records posted by other users.

Field Name	Details
`ReportAbuseReceivedOnQnCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of abuses reported by other users on the Question records posted by the User.
`ReportAbuseReceivedOnReCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** the number of abuses reported by other users on the Reply records posted by the User.
`UserId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The User ID associated with this reputation.
`CommunityId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID for the zone associated with this reputation.

Usage

Use this object to view metrics on User activity in Chatter Answers. For example, you can use the ChatterAnswersActivity object to view the number of Question records a user is following in Chatter Answers communities.

See Also:

Question
Reply
User

ChatterAnswersReputationLevel

Represents a reputation level within a Chatter Answers zone. This object is available in API version 26.0 and later.

Supported Calls

`create()`, `delete()`, `query()`, `retrieve()`, `update()`

Fields

Field	Details
`CommunityID`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the zone for which you're creating the reputation level.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Name of the reputation level.
`Value`	**Type** int **Properties** Create, Filter, Group, Sort, Update **Description** Minimum number of points for this level.

Usage

Use to create or edit reputation levels for the zone.

ChatterConversation

Represents a private conversation in Chatter, consisting of messages that conversation members have sent or received. This object is available in API version 23.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Id`	**Type** ID **Properties** Defaulted on create, Filter, Group, Sort **Description** ID of the conversation.

Usage

Use this object to identify private conversations in Chatter. Users can access this object if they have the “Manage Chatter Messages” permission. This object is read-only via the API and is provided only to allow administrators to view users' Chatter messages; for example, for compliance purposes.

See Also:

ChatterConversationMember
ChatterMessage

ChatterConversationMember

Represents a member of a private conversation in Chatter. A member has either sent messages to or received messages from other conversation participants. This object is available in API version 23.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
ConversationId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the associated ChatterConversation.
MemberId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the conversation member.

Usage

Use this object to view members of private conversations in Chatter. Users can access this object if they have the “Manage Chatter Messages” permission. This object is read-only via the API and is provided only to allow administrators to view users' Chatter messages; for example, for compliance purposes.

See Also:

ChatterConversation
ChatterMessage

ChatterMessage

Represents a message sent as part of a private conversation in Chatter. This object is available in API version 23.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
Body	**Type** textarea **Properties** None

Field Name	Details
	Description Text of the message.
ConversationId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the conversation that the ChatterMessage is associated with.
SenderId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the sender.
SentDate	**Type** dateTime **Properties** Filter, Sort **Description** Date the message was sent.

Usage

Use this object to view and delete messages sent or received via private conversations in Chatter. Users can access this object if they have the “Manage Chatter Messages” permission. This object is provided to allow administrators to view and delete users' Chatter messages; for example, for compliance purposes.

Messages are hard deleted. They aren’t sent to the Recycle Bin.

See Also:

ChatterConversation
ChatterConversationMember

ClientBrowser

Represents a cookie added to the browser upon login, and also includes information about the browser application where the cookie was inserted. This object is available in version 28.0 and later.

Supported Calls

`describeSObjects()`, `delete()`, `query()`, `retrieve()`

Fields

Field	Details
`FullUserAgent`	**Type** string **Properties** Filter, Nillable, Sort **Description** Detailed information about the client (browser). For example, `Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US; rv:1.9.0.1) Gecko/2008070208 Firefox/3.0.1`
`LastUpdate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Represents the last time the cookie was changed.
`ProxyInfo`	**Type** string **Properties** Filter, Nillable, Sort **Description** The browser's current proxy information.
`UsersId`	**Type** reference **Properties** Nillable, Group, Sort **Description** The ID of the user associated with this item.

Usage

At every login, the device the login request is from is checked against the known devices using ClientBrowser. A match means a cookie was found on the browser that matches an entry in the ClientBrowser table, so the device is known. No match means that no matching cookie was found, so the device is unknown, and the user is asked to confirm their identity.

CollaborationGroup

Represents a Chatter group. This object is available in API version 19.0 and later.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `query()`, `retrieve()`, `search()`, `update()`, `upsert()`

Fields

Field	Details
`AnnouncementId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Contains the ID of the Announcement last associated with the group. This field is available in API version 30.0 and later.
`CanHaveGuests`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** If set to `true`, indicates that a group allows customers. Chatter customers are users outside your company's email domains who can only see groups they're invited to and interact with members of those groups; they can't see any Salesforce information. This field is available starting in API version 23.0, but groups that allow customers are accessible from earlier API versions. However, when accessed from earlier API versions, groups that allow customers aren't distinguishable from private groups. We strongly recommend that you upgrade to the latest API version. If you must use an earlier version, name groups that allow customers to indicate that they include customers.
`CollaborationType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The type of Chatter group. Available values are: • `Public`—Anyone can see the group's updates, but only members can post updates. Anyone can join a public group.

Field	Details
	• `Private`—Only members can see and post updates. The group's owner or managers must add members.
`Description`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** Description of the group.
`FullPhotoUrl`	**Type** url **Properties** Filter, Nillable, Sort **Description** The URL for the group's profile photo. The URL is updated every time a photo is uploaded and reflects the most recent photo. The URL returned for an older photo is not guaranteed to return a photo if a newer photo has been uploaded. You should always query this field for the URL of the most recent photo. This field is available in API version 20.0 and later.
`GroupEmail`	**Type** email **Properties** Group, Nillable, Sort **Description** The email address for posting to the group. For private groups, only visible to members and users with "Modify All Data" or "View All Data" permissions. This field is available in API version 29.0 and later.
`HasPrivateFieldsAccess`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** If set to `true`, indicates that a user can see the `InformationBody` and `InformationTitle` fields in a private group. This field is set to `true` for members of a private group and users with "Modify All Data" or "View All Data" permissions.

Field	Details
`InformationBody`	**Type** textarea **Properties** Create, Nillable, Update **Description** The text of the Information section. For private groups, only visible to members and users with “Modify All Data” or “View All Data” permissions.
`InformationTitle`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The title of the Information section. For private groups, only visible to members and users with “Modify All Data” or “View All Data” permissions.
`IsArchived`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the group is archived (`true`) or not (`false`).
`IsAutoArchiveDisabled`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether automatic archiving is enabled for the group (`true`) or not (`false`).
`LastFeedModifiedDate`	**Type** dateTime **Properties** Filter, Sort **Description** The date of the last post or comment on the group.
`LastReferencedDate`	**Type** date

Field	Details
	Properties Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`MemberCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of members in the group.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Name of the group. The group name must be unique.
`NetworkId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** ID of the community that this group is part of. This field is available only if Salesforce Communities is enabled in your organization. You can only add a `NetworkId` when creating a group. You can't change or add a `NetworkId` for an existing group. This field is available in API version 26.0 and later.

Field	Details
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the owner of the group. Only the current group owner or people with the "Modify All Data" permission can update the `OwnerId`.
SmallPhotoUrl	**Type** url **Properties** Filter, Nillable, Sort **Description** The URL for a thumbnail of the group's profile photo. The URL is updated every time a photo is uploaded and reflects the most recent photo. The URL returned for an older photo is not guaranteed to return a photo if a newer photo has been uploaded. You should always query this field for the URL of the most recent photo. This field is available in API version 20.0 and later.

Usage

Use this object to create, edit, or delete Chatter groups. Anyone can create a group, however, you must own or manage a group in order to edit it. To delete a group, you must be the group owner or have the "Modify All Data" permission. Deleting a group permanently deletes all posts and comments to the group. It also deletes all files and links posted to the group and removes the files from other locations where they were shared.

If Communities is enabled, you can use this object to create a group within a community. Once a group is associated with a community, only members and users with "Create and Customize Communities," "Modify All Data," or "View All Data" will see the group and any related posts, comments, and files. Users with "Create and Customize Communities" or "Modify All Data" can also modify groups in all communities regardless of community membership.

As a Chatter group member, you can post to the group using the CollaborationGroupFeed object. As a Chatter group owner or manager, you can add or remove group members using the CollaborationGroupMember object, post announcements to the group using the Announcement object, and accept or decline requests to join private groups using the CollaborationGroupMemberRequest object. Additionally, the group owner, manager, or your Salesforce system administrator can invite people to join the group using the CollaborationInvitation object.

Note: The Salesforce system administrator doesn't need to be a member of the group in order to send invitations using the API.

See Also:

CollaborationGroupFeed
CollaborationGroupMember
CollaborationGroupMemberRequest

CollaborationGroupFeed

Represents a single feed item on a Chatter group feed.

A group feed shows posts and comments about the group. This object is available in API version 19.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of CollaborationGroupFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`.The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string

Field	Details
	Properties Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDate DESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization.

Field	Details
	NetworkScope can have the following values: • NetworkId—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • AllNetworks—The feed item is available in all communities. Note the following exceptions for NetworkScope: • Only feed items with a Group or User parent can set a NetworkId or a null value for NetworkScope. • For feed items with a record parent, users can set NetworkScope only to AllNetworks. • You can't filter a FeedItem on the NetworkScope field.
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the group that is tracked in the feed. The feed is displayed on the detail page for this record.
RelatedRecordId	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a ContentPost. This field is null for all posts except ContentPost.
Title	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the Type is LinkPost, the LinkUrl is the URL and this field is the link name.
Type	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field	Details
	Description The type of feed item. With the exception of `ContentPost`, `LinkPost`, and `TextPost`, feed item types should not be created directly from the API. • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received.

Field	Details
	• FacebookPost—generated when a Facebook post is created from a case. Deprecated. • MilestoneEvent—generated when a case milestone is completed or reaches violation status. • SocialPost—generated when a social post is created from a case. **Note:** If you set Type to ContentPost, you must specify ContentData and ContentFileName.
Visibility	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. Visibility can have the following values: • AllUsers—The feed item is available to all users who have permission to see the feed item. • InternalUsers—The feed item is available to internal users only. Note the following exceptions for Visibility: • For record posts, Visibility is set to InternalUsers for all internal users by default. • External users can set Visibility only to AllUsers. • On user and group posts, only internal users can set Visibility to InternalUsers.

Usage

Use this object to track changes for a group.

See Also:

CollaborationGroup
CollaborationGroupMember
NewsFeed

CollaborationGroupMember

Represents a member of a Chatter group. This object is available in API version 19.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `describeLayout()`, `query()`, `retrieve()`, `update()`

Fields

Field	Details
`CollaborationGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the associated CollaborationGroup.
`CollaborationRole`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The role of a group member. Group owners and managers can change roles for members of their groups. The valid values are: • `Standard`—Indicates that a user is a group member. Members can post and comment in the group. • `Admin`—Indicates that a user is a group manager. Managers can post and comment, change member roles, edit group settings, add and remove members, delete posts and comments, and edit the group information field. **Note:** To change the group owner, use the `OwnerId` field on the CollaborationGroup object.
`LastFeedAccessDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Date and time when a group member last accessed the group's feed. The value is only updated when a member explicitly consumes the group's feed, not when the member sees group posts in other feeds, like the profile feed.
`MemberId`	**Type** reference

Field	Details
	Properties Create, Filter, Group, Sort **Description** ID of the group member.
`NotificationFrequency`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort **Description** Required. The frequency at which Salesforce sends Chatter group email digests to this member. Can only be set by the member or users with the "Modify All Data" permission. The valid values are: • `D`—Daily • `W`—Weekly • `N`—Never • `P`—On each post The default value is specified by the member in their Chatter email settings. In Communities, the Email on Each Post option is disabled once a group exceeds 10,000 members. All members who had this option selected are automatically switched to Daily digests.

Usage

Use this object to view, create, and delete Chatter group members. You must be a group owner or manager to create members for private Chatter groups.

See Also:

CollaborationGroup
CollaborationGroupFeed
CollaborationGroupMemberRequest

CollaborationGroupMemberRequest

Represents a request to join a private Chatter group. This object is available in API version 21.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`CollaborationGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the private Chatter group.
`RequesterId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the user requesting to join the group; must be the ID of the context user.
`ResponseMessage`	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** Optional message to be included in the notification email when `Status` is `Declined`.
`Status`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update **Description** The status of the request. Available values are: • `Accepted` • `Declined` • `Pending`

Usage

This object represents a request to join a private Chatter group, and can be used to accept or decline requests to join private groups you own or manage. On create, an email is sent to the owner and managers of the private group to be accepted or declined. When the `Status` is `Accepted` or `Declined`, an email is sent to notify the requester. When the `Status` is `Declined`, a `ResponseMessage` is optionally included to provide additional details.

Note the following when working with requests:

- Users with the “Modify All Data” or “View All Data” permission can view records for all groups, regardless of membership.
- A user can be a member of 100 groups. Requests to join groups count against this limit.
- `Status` can't be specified on create.
- You can only update a request when the `Status` is `Pending`.
- You can't delete or update a request with a `Status` of `Accepted` or `Declined`.

See Also:

CollaborationGroup

CollaborationGroupMember

CollaborationInvitation

Represents an invitation to join Chatter, either directly or through a group. This object is available in API version 21.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Invitations are available if “Allow Invitations” is enabled for your organization.

Invitations are limited to your allowed domain(s) unless the invite is sent from a private group that allows customers. Allowed domains are set by the administrator.

Invitations to customers are available if “Allow Customer Invitations” is enabled for your organization. Users must have the “Invite Customers to Chatter” permission to send invitations to people outside their Chatter domain.

Fields

Field	Details
`InvitedUserEmail`	**Type** string **Properties** Create, Filter, Group, Sort **Description** The email address for the user invited to join Chatter. Label is `Invited Email`.
`InvitedUserEmailNormalized`	**Type** email **Properties** Filter, Group, Sort **Description** A normalized version of the `InvitedUserEmail` entered. Label is `Invited Email (Normalized)`.

Field	Details
`InviterId`	**Type** reference **Properties** Filter, Group, Sort **Description** The person that initiated the invitation.
`OptionalMessage`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort **Description** An optional message from the person sending the invitation to the person receiving it.
`ParentId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Used when the email address on the invitation is different than the one entered when the invitee accepts the invitation.
`SharedEntityId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the user or group associated with this invitation. • If the invitation is to join Chatter, the `SharedEntityId` is the ID of the User that created the invitation. The invitee will auto-follow the inviter. • If the invitation is to join a group within Chatter, the `SharedEntityId` is the ID of the Chatter CollaborationGroup. • To invite a customer, set `SharedEntityId` to the ID of the private CollaborationGroup with Allow Customers turned on.
`Status`	**Type** picklist

Field	Details
	Properties Filter, Group, Restricted picklist, Sort **Description** The status of the invitation. Possible values are: • `Sent` • `Accepted` • `Canceled`

Usage

Use this object to create or delete (cancel) invitations to join Chatter. You can either invite a user to join Chatter directly or as part of a CollaborationGroup.

Note: To invite someone to join a CollaborationGroup, you must be either the owner or a manager of the group or a Salesforce system administrator.

The Salesforce system administrator doesn't need to be a member of the group in order to send invitations using the API.

When the person accepts your CollaborationGroup invitation, they join the CollaborationGroup and Chatter as well.

Note: You can't send invitations to users of the organization the invite was sent from.

Invited users can view profiles, post on their feed, and join groups, but can't see your Salesforce data or records.

If your organization allows groups with customers, owners and managers of private groups with the "Allow Customers" setting, as well as system administrators, can use this object to invite customers.

Java Samples

The following example shows how to send an invitation to join Chatter:

```
public void invitePeople(String inviterUserId, String invitedEmail) throws Exception {
    CollaborationInvitation invitation = new CollaborationInvitation();
    invitation.setSharedEntityId(inviterUserId);//pass the userId of the inviter
    invitation.setInvitedUserEmail(invitedEmail);//email of the invited user
    insert(invitation);
}
```

The following example shows how to send an invitation to a customer user from a group that allows customers:

```
public void inviteToGroup(String GroupName, String invitedEmail) throws Exception {
    QueryResult qr = query("select id from collaborationgroup where name = '" +
        GroupName); //pass the group name
    String groupId = qr.getRecords()[0].getId();
    CollaborationInvitation invitation = new CollaborationInvitation();
    invitation.setSharedEntityId(groupId);//pass the groupId
    invitation.setInvitedUserEmail(invitedEmail);//email of the invited user
    insert(invitation);
}
```

Apex Samples

```
String emailAddress = 'bob@external.com';
CollaborationGroup chatterGroup = [SELECT Id
      FROM CollaborationGroup
      WHERE Name='All acme.com'
      LIMIT 1];
CollaborationInvitation inv = New CollaborationInvitation();
inv.SharedEntityId = chatterGroup.id;
inv.InvitedUserEmail = emailAddress;

try {
   Insert inv;
} catch(DMLException e){
   System.debug('There was an error with the invite: '+e);
}
```

CombinedAttachment

This read-only object contains all notes, attachments, Google Docs, documents uploaded to libraries in Salesforce CRM Content, and files added to Chatter that are associated with a record.

Supported Calls

`describeSObjects()`

Fields

Field Name	Details
`ContentSize`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Size of the document in bytes.
`ContentUrl`	**Type** url **Properties** Filter, Group, Nillable, Sort **Description** URL for links and Google Docs. This field is set only for links and Google Docs, and is one of the fields that determine the `FileType`. This field is available in API version 31.0 and later.

Field Name	Details
FileExtension	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** File extension of the document. This field is available in API version 31.0 and later.
FileType	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Type of document, determined by the file extension.
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the parent object.
RecordType	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The parent object type.
Title	**Type** string **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description Title of the attached file.

Usage

Use this object to list all notes, attachments, documents uploaded to libraries in Salesforce CRM Content, and files added to Chatter for a record, such as a related list on a detail page.

To determine if an object supports the CombinedAttachment object, call `describeSObject()` on the object. For example, `describeSObject('Account')` returns all the child relationships of the Account object, including `CombinedAttachment`. You can then query the CombinedAttachment child relationship.

```
SELECT Name, (SELECT Title FROM CombinedAttachments)
FROM Account
```

You can't directly query CombinedAttachment.

Community (Zone)

Represents a zone that contains Idea or Question objects.

Note: Starting with the Summer '13 release, Chatter Answers and Ideas "communities" have been renamed to "zones." In API version 28, the API object label has changed to `Zone`, but the API type is still `Community`.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`Description`	**Type** textarea **Properties** Filter, Nillable, Sort **Description** Text description of the zone.
`IsActive`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates whether the zone is active or inactive. An idea or question can only be posted to an active zone.
`Name`	**Type** string **Properties** Filter, Group, Sort **Description** The name of the zone.
`NetworkId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the Chatter community that this zone is associated with. This field is available only if Salesforce Communities is enabled in your organization. This field is available in API version 31.0 and later.

Usage

Use this object to create a zone in Ideas, Chatter Answers, or Answers. Zones help organize ideas and questions into logical groups and are shared by the Ideas, Answers, and Chatter Answers.

Contact

Represents a contact, which is an individual associated with an account.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `merge()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can access only portal-enabled contacts.

Fields

Field	Details
`AccountId`	**Type** reference

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** ID of the account that is the parent of this contact. We recommend that you only update up to 50 contacts simultaneously when changing the accounts on contacts enabled for a Customer Portal or partner portal. We also recommend that you make this update during times outside of your organization's business hours.
`AssistantName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The name of the assistant.
`AssistantPhone`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The telephone number of the assistant.

`Birthdate`	**Type**
	date
	Properties
	Create, Filter, Group, Nillable, Sort, Update
	Description
	The birthdate of the contact.
	The year portion of the `Birthdate` field is ignored in filter criteria, including report filters, list view filters, and SOQL queries. For example, the following SOQL query returns contacts with birthdays later in the year than today:
	`SELECT Name, Birthdate`
	`FROM Contact`
	`WHERE Birthdate > TODAY`

`CanAllowPortalSelfReg`	**Type** boolean

Field	Details
	Properties Create, Defaulted on create, Filter, Update **Description** Indicates whether this contact can self-register for your organization's Customer Portal (`true`) or not (`false`).
`CleanStatus`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates the record’s clean status as compared with Data.com. Values are: `In Sync`, `Different`, `Reviewed`, `Not Found`, `Inactive`, `Not Compared`, `Select Match`, or `Skipped`.
`ConnectionReceivedId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that shared this record with your organization. This field is only available if you have enabled Salesforce to Salesforce.
`ConnectionSentId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that you shared this record with. This field is only available if you have enabled Salesforce to Salesforce. Beginning with API version 15.0, the `ConnectionSentId` field is no longer supported. The `ConnectionSentId` field is still visible, but the value is null. You can use the new PartnerNetworkRecordConnection object to forward records to connections.
`Department`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The department of the contact.

Field	Details
`Description`	**Type** textarea **Properties** Create, Nillable, Update **Description** A description of the contact. Label is **Contact Description**. Limit: 32 KB.
`DoNotCall`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Update **Description** Indicates that the contact does not wish to be called.
`Email`	**Type** email **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Email address for the contact.
`EmailBouncedDate`	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** If bounce management is activated and an email sent to the contact bounces, the date and time the bounce occurred.
`EmailBouncedReason`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** If bounce management is activated and an email sent to the contact bounces, the reason the bounce occurred.
`Fax`	**Type** phone

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** Fax number for the contact. Label is **Business Phone**.
`FirstName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** First name of the contact. Maximum size is 40 characters.
`HasOptedOutOfEmail`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Update **Description** Indicates whether the contact would prefer not to receive email from salesforce.com (`true`) or not (`false`). Label is **Email Opt Out**.
`HasOptedOutOfFax`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Update **Description** Indicates that the contact does not wish to receive faxes.
`HomePhone`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Home telephone number for the contact.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter

Field	Details
	Description Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsEmailBounced`	**Type** boolean **Properties** Filter, Group, Sort **Description** If bounce management is activated and an email is sent to a contact, indicates whether the email bounced (`true`) or not (`false`).
`IsPersonAccount`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Read only. Label is **Is Person Account**. Indicates whether this account has a record type of Person Account (`true`) or not (`false`).
`Jigsaw`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** References the ID of a company in Data.com. If an account has a value in this field, it means that the account was imported from Data.com. If the field value is `null`, the account was not imported from Data.com. Maximum size is 20 characters. Available in API version 22.0 and later. Label is **Data.com Key**. **Important:** The `Jigsaw` field is exposed in the API to support troubleshooting for import errors and reimporting of corrected data. Do not modify this value.
`LastActivityDate`	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Value is one of the following, whichever is the most recent: • Due date of the most recent event logged against the record.

Field	Details
	• Due date of the most recently closed task associated with the record.
`LastCURequestDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The last date that a Stay-in-Touch request was sent to the contact.
`LastCUUpdateDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The last time a Stay-in-Touch update was processed for the contact.
`LastName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Last name of the contact. Maximum size is 80 characters.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.

Field	Details
LeadSource	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The source of the lead.
MailingAddress (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the mailing address. Read-only. See Address Compound Fields for details on compound address fields.
• MailingCity • MailingState • MailingCountry • MailingPostalCode	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details for the mailing address.
• MailingStateCode • MailingCountryCode	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO codes for the mailing address's state and country.
MailingStreet	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Street address for mailing address.
MailingLatitude (beta)	**Type** double

Field	Details
	Properties Create, Filter, Nillable, Sort, Update **Description** Used with `MailingLongitude` to specify the precise geolocation of a mailing address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`MailingLongitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `MailingLatitude` to specify the precise geolocation of a mailing address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`MasterRecordId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** If this object was deleted as the result of a merge, this field contains the ID of the record that was kept. If this object was deleted for any other reason, or has not been deleted, the value is `null`.
`MiddleName` (beta)	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Middle name of the contact. Maximum size is 40 characters. Contact salesforce.com Customer Support to enable this field.
`MobilePhone`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Contact's mobile phone number.

Field	Details
Name	**Type** string **Properties** Filter, Group, Sort **Description** Concatenation of `FirstName`, `MiddleName` (beta), `LastName`, and `Suffix` (beta). Maximum size is 121 characters.
`OtherAddress` (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the other address. Read-only. See Address Compound Fields for details on compound address fields.
• `OtherCity` • `OtherCountry` • `OtherPostalCode` • `OtherState`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details for alternate address.
• `OtherCountryCode` • `OtherStateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO codes for the alternate address's state and country.
`OtherLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `OtherLongitude` to specify the precise geolocation of an alternate address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.

Field	Details
OtherLongitude (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with OtherLatitude to specify the precise geolocation of an alternate address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
OtherPhone	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Telephone for alternate address.
OtherStreet	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Street for alternate address.
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The ID of the owner of the account associated with this contact.
Phone	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Telephone number for the contact. Label is **Business Phone**.
PhotoUrl	**Type** url

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** Path to be combined with the URL of a Salesforce instance (for example, https://na1.salesforce.com/) to generate a URL to request the social network profile image associated with the contact. Generated URL returns an HTTP redirect (code 302) to the social network profile image for the contact. Blank if Social Accounts and Contacts isn't enabled for the organization or if Social Accounts and Contacts is disabled for the requesting user.
`RecordTypeId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the record type assigned to this object.
`ReportsToId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** This field is not visible if `IsPersonAccount` is `true`.
`Salutation`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Honorific abbreviation, word, or phrase to be used in front of name in greetings, such as Dr. or Mrs.
`Suffix` (beta)	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Name suffix of the contact. Maximum size is 40 characters. Contact salesforce.com Customer Support to enable this field.

Field	Details
`Title`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Title of the contact such as CEO or Vice President.

Note: If you are importing Contact data and need to set the value for an audit field, such as `CreatedDate`, contact salesforce.com. Audit fields are automatically updated during API operations unless you request to set these fields yourself..

Usage

Use this object to manage individuals who are associated with an Account in your organization. You can create, query, delete, or update any Attachment associated with a contact.

Create or update contacts by converting a Lead with the `convertLead()` call.

See Also:

Standard and Custom Object Basics
ContactHistory

ContactCleanInfo

Stores the metadata Data.com Clean uses to determine a contact record's clean status. Helps you automate the cleaning or related processing of contact records. ContactCleanInfo includes a number of bit vector fields.

- `CleanedBy` indicates who (a user) or what (a Clean job) cleaned the contact record.
- `IsDifferent` indicates whether or not a field on the contact record has a value that differs from the corresponding field on the matched Data.com record.
- `IsFlaggedWrong` indicates whether or not a field on the contact record has a value that is flagged as wrong to Data.com.
- `IsReviewed` indicates whether or not a field on the contact record is in a `Reviewed` state, which means that the value was reviewed by a user but not accepted.

Their individual bits are defined here

Fields

Field Name	Details
`Address` (beta)	**Type** address **Properties** Filter, Nillable

Field Name	Details
	Description The compound form of the address. Read-only. See Address Compound Fields for details on compound address fields.
`City`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the contact.
`CleanedByJob`	**Type** boolean **Properties** Filter **Description** Indicates whether the contact record was cleaned by a Data.com Clean job (`true`) or not (`false`).
`CleanedByUser`	**Type** boolean **Properties** Filter **Description** Indicates whether the contact record was cleaned by a Salesforce user (`true`) or not (`false`).
`ContactId`	**Type** reference **Properties** Filter, Group, Sort **Description** The unique, system-generated ID assigned when the contact record was created.
`ContactStatusDataDotCom`	**Type** picklist **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description The status of the contact per Data.com. Values are: `Contact is Active per Data.com`, `Phone is Wrong per Data.com` , `Email is Wrong per Data.com`, `Phone and Email are Wrong per Data.com`, `Contact Not at Company per Data.com`, `Contact is Inactive per Data.com`, `Company this contact belongs to is out of business per Data.com`, `Company this contact belongs to never existed per Data.com` or `Email address is invalid per Data.com`.
`Country`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the contact.
`DataDotComID`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The ID Data.com maintains for the contact.
`Email`	**Type** email **Properties** Filter, Group, Nillable, Sort **Description** The email address for the contact.
`FirstName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The contact's first name.

Field Name	Details
`IsDifferentCity`	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `City` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentCountry`	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `Country` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentCountryCode`	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `Country Code` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentEmail`	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `Email` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentFirstName`	**Type** boolean **Properties** Filter

Field Name	Details
	Description Indicates whether the contact's `First Name` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentLastName	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `Last Name` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentPhone	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `Phone` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentPostalCode	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `Postal Code` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentState	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `State` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).

Field Name	Details
IsDifferentStateCode	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `State Code` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentStreet	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `Street` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentTitle	**Type** boolean **Properties** Filter **Description** Indicates whether the contact's `Title` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsFlaggedWrongAddress	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the contact's `Address` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongEmail	**Type** boolean **Properties** Filter, Update

Field Name	Details
	Description Indicates whether the contact's `Email` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongName`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the contact's `Name` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongPhone`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the contact's `Phone` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongTitle`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the contact's `Title` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsInactive`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the contact has been reported to Data.com as `Inactive` (`true`) or not (`false`).
`IsReviewedAddress`	**Type** boolean **Properties** Filter, Update

Field Name	Details
	Description Indicates whether the contact's `Address` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedEmail`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the contact's `Email` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedName`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the contact's `Name` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedPhone`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the contact's `Phone` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedTitle`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the contact's `Title` field value is in a `Reviewed` state (`true`) or not (`false`).
`LastMatchedDate`	**Type** dateTime **Properties** Filter, Sort

Field Name	Details
	Description The date the contact record was last matched and linked to a Data.com record.
`LastName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The contact’s last name.
`LastStatusChangedById`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of who or what last changed the record’s `Clean Status` field value: a Salesforce user or a Clean job.
`LastStatusChangedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date on which the record’s `Clean Status` field value was last changed.
`Latitude` (beta)	**Type** double **Properties** Filter, Nillable, Sort **Description** Used with `Longitude` to specify the precise geolocation of a billing address. Data not currently provided.
`Longitude` (beta)	**Type** double **Properties** Filter, Nillable, Sort

Field Name	Details
	Description Used with `Latitude` to specify the precise geolocation of a billing address. Data not currently provided.
`Name`	**Type** string **Properties** Filter, Group, Sort, Update **Description** Field label is **Contact Clean Info Name**. The name of the contact. Maximum size is 255 characters.
`Phone`	**Type** phone **Properties** Filter, Group, Nillable, Sort **Description** The phone number for the contact.
`PostalCode`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the contact.
`State`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the contact.
`Street`	**Type** textarea **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description Details for the billing address of the contact.
`Title`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The contact's title.

Usage

Create a customized set of `Title` field values. Use triggers to map values from fields on imported or cleaned records onto a standard set of values.

ContactFeed

Represents a single feed item in the feed on a contact record detail page.

A contact feed shows recent changes to a contact record for any fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to Contact records. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the Contact object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort

Field	Details
	Description The content of ContactFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string

Field	Details
	Properties Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only.

Field	Details
	ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.

Field	Details
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of a LinkPost.
NetworkScope	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. NetworkScope can have the following values: • NetworkId—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • AllNetworks—The feed item is available in all communities. Note the following exceptions for NetworkScope: • Only feed items with a Group or User parent can set a NetworkId or a null value for NetworkScope. • For feed items with a record parent, users can set NetworkScope only to AllNetworks. • You can't filter a FeedItem on the NetworkScope field.
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the contact record that is tracked in the feed. The feed is displayed on the detail page for this record.
RelatedRecordId	**Type** reference **Properties** Group, Nillable, Sort

Field	Details
	Description ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question.

Field	Details
	• ReplyPost—generated when Chatter Answers posts a reply. • RypplePost—generated when a user creates a Thanks badge in Work.com. • TextPost—a direct text entry on a feed. • TrackedChange—a change or group of changes to a tracked field. • UserStatus—automatically generated when a user adds a post. Deprecated. The following values appear in the Type picklist for all feed objects but apply only to CaseFeed: • CaseCommentPost—generated event when a user adds a case comment for a case object • EmailMessageEvent—generated event when an email related to a case object is sent or received • CallLogPost—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • ChangeStatusPost—generated event when a user changes the status of a case • AttachArticleEvent—generated event when a user attaches an article to a case **Note:** If you set Type to ContentPost, you must specify ContentData and ContentFileName.
Visibility	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. Visibility can have the following values: • AllUsers—The feed item is available to all users who have permission to see the feed item. • InternalUsers—The feed item is available to internal users only. Note the following exceptions for Visibility: • For record posts, Visibility is set to InternalUsers for all internal users by default. • External users can set Visibility only to AllUsers. • On user and group posts, only internal users can set Visibility to InternalUsers.

Usage

Use this object to track changes for a contact record.

See Also:

Contact
EntitySubscription
NewsFeed
UserProfileFeed

ContactHistory

Represents the history of changes to the values in the fields of a contact. This object is available in versions 11.0 and later.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`ContactId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Contact. Label is **Contact ID**.

Field	Details
NewValue	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
OldValue	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.

Usage

Use this object to identify changes to a contact.

This object respects field level security on the parent object.

See Also:

Contact

ContactOwnerSharingRule

Represents the rules for sharing a contact with a User other than the owner.

Note: Contact salesforce.com customer support to enable access to this object for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
ContactAccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update

Field	Details
	Description A value that represents the type of access granted to the target Group, UserRole, or User for Contacts. The possible values are: • `Read` • `Edit`
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
`DeveloperName`	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Rule Name** in the user interface. This field is available in API version 24.0 and later. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`GroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the source group. A Contact owned by a User in the source Group triggers the rule to give access.

Field	Details
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to **Label** on the user interface.
UserOrGroupId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the User or Group being granted access.

Usage

Use this object to manage the sharing rules for contacts.

See Also:

Contact
ContactShare

ContactShare

Represents a list of access levels to a Contact along with an explanation of the access level. For example, if you have access to a record because you own it, the `ContactAccessLevel` is `All` and `RowCause` is Owner.

Supported Calls

`describeSObjects()`, `create()`, `delete()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
ContactId	**Type** reference

Field	Details
	Properties Filter, Group, Sort **Description** ID of the Contact associated with this sharing entry. This field can't be updated.
`ContactAccessLevel`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Level of access that the User or Group has to cases associated with the account Contact. The possible values are: • `Read` • `Edit` • `All` This value is not valid for create or update. This field must be set to an access level that is higher than the organization's default access level for contacts.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. There are many possible values, including: • `Account Sharing`—The User or Group has access because a User with "`All`" manually shared the Account associated with the contact. • `Account Sharing Rule`—The User or Group has access via an Account sharing rule for the account associated with the contact. • `Account Team`—The User has access to the Contact via his or her status as an AccountTeamMember. • `Contact Sharing Rule`—The User or Group has access via a Contact sharing rule.

Field	Details
	• `ImplicitChild`—The User or Group has access to the Contact via sharing access on the associated Account. • `Manual Sharing`—The User or Group has access because a User with “All” access manually shared the Contact with them. • `Owner`—The User is the owner of the Contact or is in a Role above the Contact owner in the role hierarchy.
`UserOrGroupId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the User or Group that has been given access to the Contact. This field can't be updated.

Usage

This object allows you to determine which users and groups can view or edit Contact records owned by other users.

See Also:

AccountShare

ContactTag

Associates a word or short phrase with a Contact.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string

Field Name	Details
	Properties Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.
`Type`	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

ContactTag stores the relationship between its parent TagDefinition and the Contact being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

ContentDocument

Represents a document that has been uploaded to a library in Salesforce CRM Content or a file in Chatter. This object is available in API versions 17.0 and later for Salesforce CRM Content. The maximum number of documents that can be published is 2,000,000.

- Contact Manager, Group, Professional, Enterprise, Unlimited, and Performance Edition customers can publish a maximum of 36,000 new versions per 24–hour period.
- Developer Edition and trial users can publish a maximum of 2,500 new versions per 24–hour period.

This object is available in API version 21.0 and later for Chatter files.

Supported Calls

`delete()`, `describeLayout()``describeSObjects()`, `query()`, `retrieve()`, `undelete()`, `update()`

Special Access Rules

- Customer and Partner Portal users must have the "View Content in Portal" permission in order to query content in libraries where they have access.
- Users (including users with the "View All Data" permission) can only query files they have access to, including:
  - ◊ All Salesforce CRM Content files in libraries they're a member of and in their personal library, regardless of library permissions (API version 17.0 and later).
  - ◊ All Chatter files they own, posted on their profile, posted on groups they can see, and shared directly with them (API version 21.0 and later).
- A Salesforce CRM Content document can be deleted if any of the following are true:
  - ◊ The document is published into a personal library or is in the user's upload queue.
  - ◊ The document is published into a public library, the user is a member of that library with the "Add Content" library privilege enabled, and the user trying to delete the document is the owner.
  - ◊ The document is published into a public library that has the "Delete Content" or "Manage Library" permission enabled, and the user trying to delete the document is not the owner.

For API version 25.0 and later, you can change ownership of Chatter files and Salesforce CRM Content documents. Note the following:

- To change ownership of a Chatter file, Chatter must be enabled.
- The following must be true to change ownership of a Salesforce CRM Content document:
  - ◊ The Salesforce CRM Content app must be enabled.
  - ◊ The user who is becoming the owner of the document must have a Salesforce CRM Content feature license.
- A user can change ownership of a Salesforce CRM Content document or Chatter file if any of the following are true:
  - ◊ The user is the current owner, or has either the "Modify All Data" or "Manage Salesforce CRM Content" permission enabled.
  - ◊ The user has the "Manage Library" permission enabled for the library containing the document.

Note:

- ◊ The user who is becoming the owner of the document must be a visible user who is active, but the original owner can be inactive.
- ◊ A document's owner can be changed to a user who doesn't have access to the library that contains the document. Library administrators may need to give the new owner membership to the library.

Fields

Field	Details
`ArchivedById`	**Type** reference

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** The ID of the user who archived the document. This field is available in API version 24.0 and later.
`ArchivedDate`	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** The date when the document was archived. This field is available in API version 24.0 and later.
`ContentSize`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The size of the document in bytes. This field is available in API version 31.0 and later.
`Description`	**Type** textarea **Properties** Filter, Nillable, Sort **Description** A description of the document. This field is available in API version 31.0 and later.
`Division`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort

Field	Details
	Description A logical segment of your organization's data. Available only if the organization has the Division permission enabled.
`FileExtension`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** File extension of the document. This field is available in API version 31.0 and later.
`FileType`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Type of document, determined by the file extension. This field is available in API version 31.0 and later.
`IsArchived`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the document has been archived (`true`) or not (`false`).
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date

Field	Details
	Properties Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`LatestPublishedVersionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the latest document version.
`OwnerId`	**Type** reference **Properties** Filter, Group, Sort, Update **Description** ID of the owner of this document.
`ParentId`	**Type** reference **Properties** Filter, Group, Nillable, Sort, Update **Description** ID of the library that owns the document. Created automatically when inserting a ContentVersion via the API for the first time. This field is available in API version 24.0 and later when Salesforce CRM Content is enabled.
`PublishStatus`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Indicates if and how the document is published. Valid values are:

Field	Details
	• `P`—The document is published to a public library and is visible to other users. Label is **Public**. • `R`—The document is published to a personal library and is not visible to other users. Label is **Personal Library**. • `U`—The document is not published because publishing was interrupted. Label is **Upload Interrupted**.
`Title`	**Type** string **Properties** Filter, Group, Sort **Description** The title of a document.

Usage

- Use this object to retrieve, query, update, and delete the latest version of a document, but not a content pack, in a library or a file in Chatter. Use the ContentVersion object to create, query, retrieve, search, edit, and update a specific version of a Salesforce CRM Content document or Chatter file.
- The `query()` call doesn't return archived documents. The `queryAll()` call returns archived documents.
- If Chatter isn't enabled, Chatter files aren't included in query results.
- You can't query files that are only on records, whether they're public or not.
- To create a document, create a new version via the ContentVersion object without setting the `ContentDocumentId`. This automatically creates a parent document record. When adding a new version of the document, you must specify an existing `ContentDocumentId` which initiates the revision process for the document. When the latest version is published, the title, owner, and publish status fields are updated in the document.
- When you delete a document, all versions of that document are deleted, including ratings, comments, and tags.
- You can't create, edit, or delete content packs via the API.

Note: Content metadata, such as tags, custom fields, and content owners are tracked at the version level rather than at the document level.

- If you query versions in the API, versions with a `PublishStatus` of `Upload Interrupted` are not returned.
- A document record is a container for multiple version records. You create a new version to add a document to the system. The new version contains the actual file data which allows the document to have multiple versions. The version stores the body of the uploaded document.

See Also:

ContentDocumentFeed
ContentDocumentHistory
ContentVersion

ContentDocumentFeed

Represents a single feed item associated with ContentDocument. A content document feed shows these content document changes: creating a ContentDocument file and uploading a new ContentDocument. This object is available in versions 20.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the ContentDocument object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Create, Filter, Nillable, Sort **Description** The content of the FeedItem. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ContentData`	**Type** base64 **Properties** Create, Nillable

Field	Details
	Description The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Create, Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Create, Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime

Field	Details
	Properties Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDate DESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort

Field	Details
	Description Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Create, Filter, Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`.

Field	Details
	• For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the document that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Create, Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as a FeedItem object of `Type ContentPost`.
`Title`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort **Description** The title of the FeedItem. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Create,Filter, Group **Description** The type of FeedItem. With the exception of `ContentPost`, `LinkPost`, and `TextPost`, FeedItem types should not be created directly from the API. • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on

Field	Details
	cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case.

Field	Details
	Note: If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a ContentDocument object.

See Also:

ContentDocument

ContentDocumentHistory

Represents the history of a document. This object is available in versions 17.0 and later.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

- Customer and Partner Portal users must have the “View Content in Portal” permission to query content in libraries where they have access.
- A user can query all versions of a document from their personal library and any version that is part of or shared with a library where they are a member, regardless of library permissions.

Fields

Field	Details
ContentDocumentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the document.
Division	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A logical segment of your organization’s data. Available only if the organization has the Division permission enabled.
Field	**Type** picklist **Properties** Filter, Group, Sort, Restricted picklist **Description** The name of the field that was changed. Possible values include: • contentDocPublished—The document is published into a library. • contentDocUnpublished—The document is archived or removed from a library, either directly or when the owning library is changed. • contentDocRepublished—The document is removed from the archive. • contentDocFeatured—The document is featured. • contentDocSubscribed—The document is subscribed to. • contentDocUnsubscribed—The document is no longer subscribed to.
NewValue	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
OldValue	**Type** anyType

Field	Details
	Properties Nillable, Sort **Description** The latest value of the field before it was changed.

Usage

Use this read-only object to query the history of a document.

See Also:

ContentDocument

ContentDocumentLink

Represents the link between a Salesforce CRM Content document or Chatter file and where it's shared. A file can be shared with other users, Chatter groups, records, and Salesforce CRM Content libraries. This object is available in versions 21.0 and later for Salesforce CRM Content documents and Chatter files.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`

Special Access Rules

- Customer and Partner Portal users must have the “View Content in Portal” permission to query content in libraries where they have access.
- Users (including users with the “View All Data” permission) can only query files they have access to, including:
  - ◊ All Salesforce CRM Content files in libraries they're a member of and in their personal library, regardless of library permissions (API version 17.0 and later).
  - ◊ All Chatter files they own, posted on their profile, posted on groups they can see, and shared directly with them (API version 21.0 and later).
- For API version 25.0 and later, creation and deletion of ContentDocumentLink objects with a `LinkedEntityId` of types `User`, `CollaborationGroup`, or `Organization` are supported via the API.
- For organizations with the Communities pilot enabled, a document can only be shared with users and groups that are a part of the community the file was created in.

Fields

Field	Details
`ContentDocumentId`	**Type** reference **Properties** Create, Filter, Group, Sort

Field	Details
	Description ID of the document.
`LinkedEntityId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the linked object. Can include Chatter users, groups, records (any that support Chatter feed tracking including custom objects), and Salesforce CRM Content libraries.
`ShareType`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Required. The permission granted to the user of the shared file in a library. This is determined by the permission the user already has in the library.This field is available in API version 25.0 and later. **`Viewer`** The user can explicitly view but not edit the shared file. **`Collaborator`** The user can explicitly view and edit the shared file. **`Inferred`** The user's permission is determined by the related record. For shares with a library, this is defined by the permissions the user has in that library.
`Visibility`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item.

Field	Details
	• `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`. The visibility of a ContentDocumentLink on a record post is derived from the highest level of visibility of the link on that record.

Usage

Use this object to query the locations where a file is shared or query which files are linked to a particular location. For example, the following query returns a particular document shared with a Chatter group:

```
SELECT ContentDocument.title FROM ContentDocumentLink WHERE ContentDocumentId =
'069D00000000so2' AND LinkedEntityId = '0D5000000089123'
```

- You can't run a query without filters against ContentDocumentLink.
- You can't filter on ContentDocument fields if you're filtering by `ContentDocumentId`. You can only filter on ContentDocument fields if you're filtering by `LinkedEntityId`.
- You can't filter on the related object fields. For example, you can't filter on the properties of the account to which a file is linked. You can filter on the properties of the file, such as the title field.

A SOQL query must filter on one of `Id`, `ContentDocumentId`, or `LinkedEntityId`.

See Also:

ContentDocument

ContentVersion

Represents a specific version of a document in Salesforce CRM Content or file in Chatter. The maximum number of versions that can be published in a 24-hour period is 36,000. This object is available in versions 17.0 and later for Salesforce CRM Content documents. This object is available in versions 20.0 and later for Chatter files.

Supported Calls

`create()`, `describeLayout()`, `describeSObjects()`, `query()`, `retrieve()`, `search()`, `update()`, `upsert()`

Special Access Rules

- Customer and Partner Portal users must have the “View Content in Portal” permission to query content in libraries where they have access.
- Customer and Partner Portal users can only publish, version, or edit documents if they have a Salesforce CRM Content feature license.
- All users with a content feature license can create versions in their personal library.
- Users (including users with the “View All Data” permission) can only query files they have access to, including:

- ◊ All Salesforce CRM Content files in libraries they're a member of and in their personal library, regardless of library permissions (API version 17.0 and later).
- ◊ All Chatter files they own, posted on their profile, posted on groups they can see, and shared directly with them (API version 21.0 and later).

- All users can update versions in their personal library.
- The owner of a version or document can update the document if they are a member of the library, regardless of library permissions.
- To update a Salesforce CRM Content document, the user must be a member of the library with one of these library privileges enabled:
  - ◊ "Add Content"
  - ◊ "Add Content On Behalf of Others"
  - ◊ "Manage Library"
- `FileType` is defined by either `ContentUrl` for links or `PathOnClient` for documents, but not both.

Fields

Field	Details
`Checksum`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** MD5 checksum for the file.
`ContentDocumentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the document.
`ContentModifiedById`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the user who modified the document.
`ContentModifiedDate`	**Type** dateTime

Field	Details
	Properties Filter, Nillable, Sort **Description** Date the document was modified.
ContentSize	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Size of the document in bytes. Always zero for links.
ContentUrl	**Type** url **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** URL for links. This is only set for links. One of the fields that determines the FileType.
Description	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Description of the content version.
Division	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A logical segment of your organization’s data. Available only if the organization has the Division permission enabled.
FeaturedContentBoost	**Type** int **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Read only. Designates a document as featured.
`FeaturedContentDate`	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Date the document was featured.
`FileExtension`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** File extension of the document. This field is available in API version 31.0 and later.
`FileType`	**Type** string **Properties** Filter, Group, Sort **Description** Type of content determined by `ContentUrl` for links or `PathOnClient` for documents.
`FirstPublishLocationId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** ID of the location where the version was first published. If the version is first published into a user's personal library or My Files, the field will contain the ID of the user who owns the personal library or My Files. If the first version is published into a public library, the field will contain the ID of that library. This field is only set the first time a version is published via the API. It isn't set when a version is published in Salesforce CRM Content or uploaded in Chatter. After the version is published, it is a read-only field.

Field	Details
	If you don't set a `FirstPublishLocationId` via the API, this field defaults to the user's personal library, unless `Origin` is set to **H**, then this field defaults to My Files.
`IsLatest`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this is the latest version of the document (`true`) or not (`false`).
`Language`	**Type** picklist **Properties** Create, Filter, Nillable, Restricted picklist, Update **Description** The language for this document. This field defaults to the user's language unless the organization is multi-language enabled. Specifies the language of the labels returned. The value must be a valid user locale (language and country), such as `de_DE` or `en_GB`. For more information on locales, see the `Language` field on the CategoryNodeLocalization object.
`NegativeRatingCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Read only. The number of times different users have given the document a thumbs down. Rating counts for the latest version are not version-specific. If Version 1 receives 10 thumbs-down votes, and Version 2 receives 2 thumbs-down votes, the `NegativeRatingCount` on Version 2 is 12. However, rating counts are not retroactive for prior versions. The `NegativeRatingCount` on Version 1 is 10.
`NetworkId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort

Field	Details
	Description ID of the community that this file originated from. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. You can only add a `NetworkId` when creating a file. You can't change or add a `NetworkId` for an existing file.
`Origin`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** The source of the content version. Valid values are: • **C**—This is a Content document from the user's personal library. Label is **Content**. The `FirstPublishLocationId` must be the user's ID. If `FirstPublishLocationId` is left blank, it defaults to the user's ID. • **H**—This is a Chatter file from the user's My Files. Label is **Chatter**. The `FirstPublishLocationId` must be the user's ID. If `FirstPublishLocationId` is left blank, it defaults to the user's ID. Origin can only be set to **H** if Chatter is enabled for your organization. This field defaults to C. Label is **Content Origin**.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the owner of this document.
`PathOnClient`	**Type** string **Properties** Create, Filter, Nillable, Sort **Description** The complete path of the document. One of the fields that determines the `FileType`. **Note:** You must specify a complete path including the path extension in order for the document to be visible in the Preview tab.
`PositiveRatingCount`	**Type** int

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** Read only. The number of times different users have given the document a thumbs up. Rating counts for the latest version are not version-specific. If Version 1 receives 10 thumbs-up votes, and Version 2 receives 2 thumbs-up votes, the `PositiveRatingCount` on Version 2 is 12. However, rating counts are not retroactive for prior versions. The `PositiveRatingCount` on Version 1 is 10.
`PublishStatus`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Indicates if and how the document is published. Valid values are: • `P`—The document is published to a public library and is visible to other users. Label is **Public**. • `R`—The document is published to a personal library and is not visible to other users. Label is **Personal Library**. • `U`—The document is not published because publishing was interrupted. Label is **Upload Interrupted**.
`RatingCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Read only. Total number of positive and negative ratings.
`ReasonForChange`	**Type** string **Properties** Create, Filter, Nillable, Sort **Description** The reason why the document was changed. This field can only be set when inserting a new version (revising) a document.
`RecordTypeId`	**Type** reference

Field	Details
	Properties Create, Filter, Nillable, Update **Description** ID of the content type of the version. Content types are containers for custom fields in Salesforce CRM Content. Custom fields are restricted in `RecordTypeId`. When an administrator creates a custom field via the API it must be added to at least one content type: • If the custom field is added to the default (General) content type, the `RecordTypeId` that corresponds to that content type does not have to be set on the version record. • If the custom field is added to a custom content type, the `RecordTypeId` that corresponds to that content type must be set on the version record.
`TagCsv`	**Type** textarea **Properties** Create, Nillable, Sort, Update **Description** Text used to apply tags to a content version via the API.
`Title`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The title of a document.
`VersionData`	**Type** base64 **Properties** Create, Nillable **Description** Encoded file data. This field can't be set for links. The maximum file size you can upload via the SOAP API must be less than 50 MB. When a document is uploaded or downloaded via the API, it is converted to base64 and stored in `VersionData`. This conversion increases the document size by approximately 37%. You must account for the base64 conversion increase so that the file you plan to upload is less than 50 MB.

Field	Details
`VersionNumber`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The version number. The number increments with each version of the document, for example, 1, 2, 3.

Usage

- Use this object to create, query, retrieve, search, edit, and update a specific version of a Salesforce CRM Content document or Chatter file. Use the ContentDocument object to retrieve, query, update, and delete the latest version of a document, but not a content pack, in a library or a file in Chatter.
- Use this object to create, query, retrieve, search, edit, and update a specific version of a Chatter file. Use the ContentDocument object to retrieve, query, update, and delete the latest version of a file in Chatter.
- If Chatter isn't enabled, Chatter files aren't included in query results.
- You can't query files that are only on records, whether they're public or not.
- Not all fields can be set for Chatter files.
- You can only update a version if it is the latest version and if it is published.
- You can't archive versions.
- You can't change content types on versions.
- You can't delete a version via the API.
- The maximum file size you can upload via the SOAP API must be less than 50 MB. When a document is uploaded or downloaded via the API, it is converted to base64 and stored in `VersionData`. This conversion increases the document size by approximately 37%. You must account for the base64 conversion increase so that the file you plan to upload is less than 50 MB.
- To download a document via the API, you must export the `VersionData` of the document. This does not increase the download count.
- When you upload a document from your local drive using the Data Loader, you must specify the actual path in both `VersionData` and `PathOnClient`. `VersionData` identifies the location and extracts the format and `PathOnClient` identifies the type of document being uploaded.
- SOQL queries on the ContentVersion object return all versions of the document. SOSL searches on the ContentVersion object return only the most recent version of the document.
- If you query versions in the API, versions with a `PublishStatus` of `Upload Interrupted` are not returned.
- Documents published into a personal library assume the default (General) content type.

  Note: When Salesforce CRM Content is provisioned for an organization the administrator can create a General content type. Additional content types can be created to categorize documents. The administrator can rename the General content type.

- Contact Manager, Group, Professional, Enterprise, Unlimited, and Performance Edition customers can publish a maximum of 36,000 new versions per 24–hour period. Developer Edition and trial users can publish a maximum of 2,500 new versions per 24–hour period.
- Custom validation rules can prevent an update of documents published into a personal library via the API.

ContentVersion API Tagging

Tags can be applied to ContentVersion records using either Enterprise or Partner API.

To apply tags to a ContentVersion record, set a value in the `TagCsv` field. For example, setting this field to `one,two,three` creates and associates three tags to that version.

- The maximum length of the `TagCsv` field is 2,000 characters.
- The maximum length of an individual tag is 100 characters.
- When tags are applied to a version, the content is indexed automatically and the tags are searchable.
- You can't apply tags to a `TagCsv` that is published into a personal library.
- You can't apply tags using the ContentDocument object.
- You can't change or delete tag names. You can remove tags from a document, but that doesn't delete the tag.
- Tags are case insensitive. You can't have two tags with the same name even if they use different uppercase and lowercase letters. The case of the original tag is always used.

To delete tags from a ContentVersion record, perform a standard API update, and remove any values from the `TagCsv` field that you want to delete. For example, if the original `TagCsv` is `one,two,three`, perform an API update specifying `one,three` in the `TagCsv` field to delete `two`. To delete all tags from a ContentVersion you perform a standard API update by setting the field to `null`.

If you create a ContentVersion record and want to revise it via the API, you insert another ContentVersion record but associate it to the same ContentDocument record as the original. This has an impact on tagging:

- If you insert the revision and do not set any value in the `TagCsv` field, any tags applied to the previous version are automatically applied to the new version.
- If you insert the revision and specify a new `TagCsv` field, no tags transfer over and the tags you specify are applied instead.

When you perform a SOQL query for a ContentVersion record and select the `TagCsv` field, all the tags associated with that record are returned. The tags in the string are always ordered alphabetically even if they were inserted in a different order. You can't use the `TagCsv` field as part of a filter in a SOQL query. You can't query all tags in your organization.

Library tagging rules:

- API tagging respects the tagging restrictions that exist on any library that the document is published into. For example, if the library is in restricted tagging mode and only allows tags `one,three`, you can't save a version with a `TagCsv` of `one,two,three`.
- If the library is in guided tagging mode, you can apply tags to the ContentVersion. You can't query the value of guided tags on a library, but you can query the tagging model of a library.

See Also:

ContentDocument
ContentVersionHistory

ContentVersionHistory

Represents the history of a specific version of a document. This object is available in versions 17.0 and later.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

- Customer and Partner Portal users must have the "View Content in Portal" permission to query content in libraries where they have access.
- A user can query all versions of a document from their personal library and any version that is part of or shared with a library where they are a member, regardless of library permissions.

Fields

Field	Details
ContentVersionId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the version.
Division	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A logical segment of your organization's data. Available only if the organization has the Division permission enabled.
Field	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed. Possible values include: • `contentVersionCreated`—A new version is created. • `contentVersionUpdated`—The title, description, or any custom field on the version is changed. • `contentVersionDownloaded`—A version is downloaded. • `contentVersionViewed`—The version details are viewed. • `contentVersionRated`—The version is rated. • `contentVersionCommented`—The version receives a comment. • `contentVersionDataReplaced`—The new version replaces the previous version, which can happen only when the new version is uploaded immediately after the previous version.
NewValue	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.

Field	Details
OldValue	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.

Usage

Use this read-only object to query the history of a document version.

See Also:

ContentVersion

ContentWorkspace

Represents a public library in Salesforce CRM Content. This object is available in versions 17.0 and later.

Note: This object does not apply to personal libraries.

Supported Calls

`describeSObjects(), query(), retrieve()`

Special Access Rules

- Customer and Partner Portal users can only edit the library document object if they have a Salesforce CRM Content feature license.
- Customer and Partner Portal users can query this object if they have the "View Content in Portal" permission. A user can query all public libraries where they are members, regardless of library permissions.

Fields

Field	Details
DefaultRecordTypeId	**Type** reference **Properties** Filter, Nillable **Description** ID of the default content type for the library. Content types are the containers for custom fields in Salesforce CRM Content.

Field	Details
Description	**Type** textarea **Properties** Filter, Nillable **Description** Text description of the content library.
IsRestrictContentTypes	**Type** boolean **Properties** Defaulted on create, Filter **Description** Read only. Indicates whether content types have been restricted (`true`) or not (`false`).
IsRestrictLinkedContentTypes	**Type** boolean **Properties** Defaulted on create, Filter **Description** Read only. Indicates whether linked content types have been restricted (`true`) or not (`false`).
Name	**Type** string **Properties** Filter, idLookup **Description** Name of the library.
TagModel	**Type** picklist **Properties** Filter, Nillable, Restricted picklist **Description** The type of tagging assigned to a library. Valid values are: • `Open Tagging`—No restrictions on tagging. Users can enter any tag when publishing or editing content.

Field	Details
	• Guided Tagging—Users can enter any tag when publishing or editing content, but they are also offered a list of suggested tags. • Restricted Tagging—Users must choose from a list of suggested tabs.

Usage

Use this object to query libraries to find out where documents can be published.

If the content type is not specified when publishing a new version into a library, it will be determined by the `DefaultRecordTypeId` of the primary library.

You can't create, update, or delete a library via the API.

See Also:

ContentWorkspaceDoc

ContentWorkspaceDoc

Represents a link between a document and a public library in Salesforce CRM Content. This object is available in versions 17.0 and later.

Note: This object does not apply to documents and versions in a personal library.

Supported Calls

`create()`, `delete()`, `describeSObjects()` `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- Customer and Partner Portal users must have the "View Content in Portal" permission in order to query and obtain content in libraries where they have access.
- Customer and Partner Portal users can only edit documents if they have a Salesforce CRM Content feature license.
- To create a ContentWorkspaceDoc, you must be a member of the library with one of these library privileges enabled:
  - ◊ "Add Content"
  - ◊ "Add Content On Behalf of Others"
  - ◊ "Manage Library"
- To query all library documents in a library, a user must be a member of that library, regardless of library permissions.

Fields

Field	Details
ContentDocumentId	**Type** reference

Field	Details
	Properties Create, Filter **Description** Read only. ID of the library document.
`ContentWorkspaceId`	**Type** reference **Properties** Create, Filter **Description** Read only. ID of the library.
`IsOwner`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Read only. Indicates whether the library owns the document and determines permissions for that document (`true`) or not (`false`). Documents can belong to more than one library, but only one library owns the document and determines its permissions.

Usage

- Use this object to link a document to one or more libraries.
- To share a document with additional libraries, create additional ContentWorkspaceDoc records which join the document to the additional libraries.
- Inserting a ContentWorkspaceDoc triggers the publish process for public libraries.
- A document can be published into many public libraries, but it will always be owned by one library which controls the security of the document.
- A document can only be published into the document owner's personal library. You can't publish into another user's personal library. Personal libraries are not visible via the API.
- To publish a document into a personal library, you must specify your user ID as the first publish location ID. If you leave the first publish location ID blank, it defaults to the current user's ID.
- A document can be published from a personal library into a public library, but once it has been published into the public library, it can't be published into the personal library again.
- You can't publish a document from a personal library into a public library that has restricted content types.
- You can't update or delete a library document via the API.

See Also:

ContentWorkspace

Contract

Represents a contract (a business agreement) associated with an Account.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`AccountId`	**Type** reference **Properties** Create, Filter, Group, Sort,Update **Description** Required. ID of the Account associated with this contract.
`ActivatedById`	**Type** reference **Properties** Filter, Group, Nillable, Sort,Update **Description** ID of the User who activated this contract.
`ActivatedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort, Update **Description** Date and time when this contract was activated.
`BillingAddress` (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the billing address. Read-only. See Address Compound Fields for details on compound address fields.

Field	Details
BillingCity	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details for the billing address. Maximum size is 40 characters.
BillingCountry	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details for the billing address of this account. Maximum size is 80 characters.
BillingCountryCode	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO country code for the contract's billing address.
BillingLatitude (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with BillingLongitude to specify the precise geolocation of a billing address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
BillingLongitude (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with BillingLatitude to specify the precise geolocation of a billing address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
BillingPostalCode	**Type** string

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** Details for the billing address of this account. Maximum size is 20 characters.
`BillingState`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Details for the billing address. Maximum size is 80 characters.
`BillingStateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO state code for the contract's billing address.
`BillingStreet`	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Street address for the billing address.
`CompanySignedDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Date on which the contract was signed by your organization.
`CompanySignedId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description ID of the User who signed the contract.
`ContractNumber`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, Sort **Description** Number of the contract.
`ContractTerm`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Number of months that the contract is valid.
`CustomerSignedDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Date on which the customer signed the contract.
`CustomerSignedId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the Contact who signed this contract.
`CustomerSignedTitle`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Title of the customer who signed the contract.

Field	Details
Description	**Type** textarea **Properties** Create, Nillable, Update **Description** Description of the contract.
EndDate	**Type** date **Properties** Filter, Group, Nillable, Sort, **Description** Read-only. Calculated end date of the contract. This value is calculated by adding the `ContractTerm` to the `StartDate`.
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
LastActivityDate	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Value is one of the following, whichever is the most recent: • Due date of the most recent event logged against the record. • Due date of the most recently closed task associated with the record.
LastApprovedDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Last date the contract was approved.

Field	Details
LastReferencedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
LastViewedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (LastReferencedDate) and not viewed.
OwnerExpirationNotice	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Number of days ahead of the contract end date (15, 30, 45, 60, 90, and 120). Used to notify the owner in advance that the contract is ending.
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the user who owns the contract.
Pricebook2Id	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the pricebook, if any, associated with this contract.

Field	Details
RecordTypeId	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the record type assigned to this object.
ShippingAddress (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the shipping address. Read-only. See Address Compound Fields for details on compound address fields.
ShippingCity	**Type** string **Properties** Create, Filter, Nillable, Update **Description** Details of the shipping address. City maximum size is 40 characters.
ShippingCountry	**Type** string **Properties** Create, Filter, Nillable, Update **Description** Details of the shipping address. Country maximum size is 80 characters.
ShippingCountryCode	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO country code for the contract's shipping address.
ShippingLatitude (beta)	**Type** double

Field	Details
	Properties Create, Filter, Nillable, Sort, Update **Description** Used with `ShippingLongitude` to specify the precise geolocation of a shipping address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`ShippingLongitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `ShippingLatitude` to specify the precise geolocation of an address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`ShippingPostalCode`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** Details of the shipping address. Postal code maximum size is 20 characters.
`ShippingState`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** Details of the shipping address. State maximum size is 80 characters.
`ShippingStateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO state code for the contract's shipping address.
`ShippingStreet`	**Type** string

Field	Details
	Properties Create, Filter, Nillable, Update **Description** The street address of the shipping address. Maximum of 255 characters.
`SpecialTerms`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Special terms that apply to the contract.
`StartDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort,Update **Description** Start date for this contract. Label is **Contract Start Date**.
`Status`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Restricted picklist, Sort, Update **Description** The picklist of values that indicate order status. Each value is within one of two status categories defined in `StatusCode`. For example, the status picklist may contain: Ready to Ship, Shipped, Received as values within the Activated `StatusCode`.
`StatusCode`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The status category for the order. An order can be either Draft, InApproval, or Activated. Label is **Status Category**.

Usage

The Contract object represents a business agreement.

The `Status` field specifies the current state of a contract. Status strings (defined in the ContractStatus object) represent its current state (Draft, InApproval, or Activated).

Client applications must initially create a Contract in a non-Activated state. Client applications can subsequently activate a Contract by updating it and setting the value in its `Status` field to Activated; however, the `Status` field is the only field you can update when activating the Contract.

Once a Contract has been activated, your client application can't change its status; however, prior to activation, your client application can change the status value from Draft to InApproval via the API. Also, your client application can delete contracts whose status is Draft or InApproval but not when a contract status is Activated.

Client applications can use the API to create, update, delete, and query any Attachment associated with a contract.

See Also:

ContractContactRole
ContractHistory
ContractStatus

ContractContactRole

Represents the role that a given Contact plays on a Contract.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`ContactId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the Contact associated with this Contract.
`ContractId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Contract.
`IsDeleted`	**Type** boolean

Field	Details
	Properties Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsPrimary`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort,Update **Description** Specifies whether this Contact plays the primary role on this Contract (`true`) or not (`false`). Note that each contract has only one primary contact role. Default is `false`. Labels is **Primary**.
`Role`	**Type** picklist **Properties** Create, Filter, Nillable, Group, Sort, Update **Description** Name of the role played by the Contact on this Contract, such as Decision Maker, Approver, Buyer, and so on. Must be unique—there can't be multiple records in which the `ContractId`, `ContactId`, and `Role` values are identical. Different contacts can play the same role on the same contract. A contact can play different roles on the same contract.

Usage

Use the ContractContactRole object to define the role that a given Contact plays on a given Contract within the context of a specific Opportunity.

See Also:

ContractStatus

ContractFeed

Represents a single feed item in the feed on the contract record detail page.

A contract feed shows recent changes to a contract record for any fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to contracts in Salesforce. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the Contract object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of ContractFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64

Field	Details
	Properties Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.

Field	Details
CreatedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
FeedPostId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
InsertedById	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
LastModifiedDate	**Type** dateTime

Field	Details
	Properties Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`:

Field	Details
	• Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the contract record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging).

Field	Details
	For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field	Details
	Description Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a contract record.

See Also:

Contract
EntitySubscription
NewsFeed
UserProfileFeed

ContractHistory

Represents the history of changes to the values in the fields of a contract.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`ContractId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Contract. Label is Contract ID.

Field	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.

Usage

Use this object to identify changes to a contract.

This object respects field level security on the parent object.

See Also:

ContractStatus

ContractLineItem

Represents a Product2 in a ServiceContract (customer support agreement).

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`AssetId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** Required. ID of the Asset associated with the contract line item. Must be a valid asset ID.
`Description`	**Type** textarea **Properties** Create, Nillable, Update **Description** Description of the contract line item.
`Discount`	**Type** percent **Properties** Create, Filter, Nillable, Update **Description** The discount for the product as a percentage. When updating, if you specify `Discount` without specifying `TotalPrice`, the `TotalPrice` will be adjusted to accommodate the new `Discount` value, and the `UnitPrice` will be held constant. If you specify both `Discount` and `Quantity`, you must also specify either `TotalPrice` or `UnitPrice` so the system can determine which one to automatically adjust.
`EndDate`	**Type** date

Field	Details
	Properties Create, Filter, Nillable, Update **Description** The last day the contract line item is in effect.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`LineItemNumber`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Update **Description** Automatically-generated number that identifies the contract line item.
`ListPrice`	**Type** currency **Properties** Filter, Nillable **Description** Corresponds to the `UnitPrice` on the PricebookEntry that is associated with this line item, which can be in the standard pricebook or a custom pricebook. A client application can use this information to show whether the unit price (or sales price) of the line item differs from the pricebook entry list price.
`PricebookEntryId`	**Type** reference

Field	Details
	Properties Create, Filter **Description** Required. ID of the associated PricebookEntry. Only exists if Product2 is enabled.
`Quantity`	**Type** double **Properties** Create, Filter, Update **Description** Number of units of the contract line item (product) included in the associated service contract.
`ServiceContractId`	**Type** reference **Properties** Create, Filter **Description** Required. ID of the ServiceContract associated with the contract line item. Must be a valid asset ID.
`StartDate`	**Type** date **Properties** Create, Filter, Nillable, Update **Description** The first day the contract line item is in effect.
`Status`	**Type** picklist **Properties** Filter, Nillable **Description** Status of the contract line item.
`Subtotal`	**Type** currency

Field	Details
	Properties Filter, Nillable **Description** Contract line item's sales price multiplied by the `Quantity`.
`TotalPrice`	**Type** currency **Properties** Filter, Nillable **Description** This field is available only for backward compatibility. It represents the total price of the ContractLineItem If you do not specify `UnitPrice`, this field is required. If you specify `Discount` and `Quantity`, this field or `UnitPrice` is required. When updating, you can change either this value or the `UnitPrice`, but not both at the same time. This field is nillable, but you can't set both `TotalPrice` and `UnitPrice` to null in the same update request. To insert the `TotalPrice` for a contract line item via the API (given only a unit price and the quantity), calculate this field as the unit price multiplied by the quantity.
`UnitPrice`	**Type** currency **Properties** Create, Filter, Update **Description** The unit price for the contract line item. In the user interface, this field's value is calculated by dividing the total price of the contract line item by the quantity listed for that line item. Label is **Sales Price**. This field or `TotalPrice` is required. You can't specify both. If you specify `Discount` and `Quantity`, this field or `TotalPrice` is required.

Usage

Use this object to query and manage contract line items.

See Also:

ContractLineItemHistory

ContractLineItemHistory

Represents the history of changes to the values in the fields on a ContractLineItem (items in a customer support agreement).

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`ContractLineItemId`	**Type** reference **Properties** Filter, Group, Sort **Description** Required. ID of the ContractLineItem.
`Division`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A logical segment of your organization’s data. Available only if the organization has the Division permission enabled.
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`NewValue`	**Type** anyType **Properties** Nillable, Sort

Field	Details
	Description The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.

Usage

Use this object to identify changes to a contract line item.

This object respects field level security on the parent object.

See Also:

ContractLineItem

ContractStatus

Represents the status of a Contract, such as Draft, InApproval, Activated, Terminated, or Expired.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`IsDefault`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this is the default contract status value (`true`) or not (`false`) in the picklist.
`MasterLabel`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Master label for this contract status value. This display value is the internal label that does not get translated.
`SortOrder`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Number used to sort this value in the contract status picklist. These numbers are not guaranteed to be sequential, as some previous contract status values might have been deleted.
`StatusCode`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Code indicating the status of a contract. One of the following values: • `Draft` • `InApproval` • `Activated` Two other values (`Terminated` and `Expired`) are defined but are not available for use via the API.

Usage

This object represents a value in the contract status picklist. The contract status picklist provides additional information about the status of a Contract, such as its current state (`Draft`, `InApproval`, or `Activated`). You can query these records to retrieve the set of values in the contract status picklist, and then use that information while processing Contract objects to determine more information about a given contract. For example, the application could test whether a given contract is activated based on its `Status` value and the value of the `StatusCode` property in the associated ContractStatus object.

See Also:

ContractContactRole

ContractTag

Associates a word or short phrase with a Contract.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.
`Type`	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

ContractTag stores the relationship between its parent TagDefinition and the Contract being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

CronJobDetail

Contains details about the associated scheduled job, such as the job's name and type. This object is available in API version 29.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`JobType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of the associated scheduled job. The following are the available job types. Each job type label is listed with its value in parenthesis. Use the job type value when querying for a specific job type. • Data Export (0) • Dashboard Refresh (3) • Analytic Snapshot (4) • Scheduled Apex (7) • Report Run (8) • Batch Job (9)
`Name`	**Type** string **Properties** Filter, Group, idLookup, Sort, **Description** The name of the associated scheduled job.

Usage

Use this object to query additional information about a scheduled job, such as the job's name and type.

CronTrigger

Represents a scheduled job, similar to a cron job on UNIX systems. This object is available in API version 17.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`CronExpression`	**Type** string **Properties** Filter, Group, Nillable, Sort, **Description** The cron expression used to initiate the schedule. Syntax: ***`Minutes, Hours, Day of month, Month, Day of week, optional Year`*** See Using the `System.Schedule` Method in the *Force.com Apex Code Developer's Guide*.
`CronJobDetailId`	**Type** reference **Properties** Filter, Group, Nillable, Sort, **Description** The ID of the CronJobDetail record containing more details about this scheduled job.
`EndTime`	**Type** dateTime **Properties** Filter, Nillable, Sort, **Description** The date and time when the job either finished or will finish.
`NextFireTime`	**Type** dateTime **Properties** Filter, Nillable, Sort

Field	Details
	Description The next date and time the job is scheduled to run. `null` if the job is not scheduled to run again.
`OwnerId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Owner of the job.
`PreviousFireTime`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The most recent date and time the job ran. `null` if the job has not run before current local time.
`StartTime`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date and time when the most recent iteration of the scheduled job started.
`State`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The current state of the job. Possible values are: • `WAITING` • `ACQUIRED` • `EXECUTING` • `COMPLETE` • `BLOCKED` • `ERROR` • `PAUSED` • `PAUSED_BLOCKED`

Field	Details
	• DELETED
TimesTriggered	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of times this job has been triggered.
TimeZoneSidKey	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Returns the timezone ID. For example, `America/Los_Angeles`.

Usage

Use this object to query scheduled jobs in your organization.

CurrencyType

Represents the currencies used by an organization for which the multicurrency feature is enabled.

Supported Calls

`create()`, `describeSObjects()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `update()`

Special Access Rules

- This object is not available in single-currency organizations.
- You need the “Customize Application” permission to edit this object.
- Your client application can't delete this object.
- Customer Portal users can't access this object.

Fields

Field	Details
ConversionRate	**Type** double **Properties** Filter

Field	Details
	Description Required. Conversion rate of this currency type against the corporate currency.
`DecimalPlaces`	**Type** int **Properties** Filter **Description** Required. For this currency, specifies the number of digits to the right of the decimal point, such as zero (`0`) for JPY or `2` for USD.
`IsActive`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether this currency type is active (`true`) or not (`false`). Inactive currency types do not appear in picklists in the user interface. Label is **Active**. This field defaults to `false` if no value is provided when updating or inserting a record.
`IsCorporate`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether this currency type is the corporate currency (`true`) or not (`false`). Label is **Corporate Currency**. All other currency conversion rates are applied against this corporate currency. If a currency is already defined as the corporate currency in the user interface, it can't be unset. When a non-corporate currency is set to a corporate currency, the system reconfigures all conversion rates based on the new corporate currency.
`IsoCode`	**Type** picklist **Properties** Filter, Restricted picklist **Description** Required. ISO code of the currency. Must be one of the valid alphabetic, three-letter currency ISO codes defined by the ISO 4217 standard, such as

Field	Details
	`USD`, `GBP`, or `JPY`. Must be unique within your organization. Label is **Currency ISO Code**.

Usage

This object is for multicurrency organizations only. Use this object to define the currencies your organization uses.

When updating an existing record, make sure to provide values for all fields to avoid undesired changes to the CurrencyType. For example, if a value for `IsActive` is not provided, the default (`false`) is used, which could result in a currently active CurrencyType becoming inactive.

See Also:

DatedConversionRate

Standard and Custom Object Basics

CustomBrand

Represents a custom branding and color scheme. This object is available in API version 28.0 and later.

Supported Calls

`create()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

This object is available only if Salesforce Communities is enabled in your organization.

Fields

Field Name	Details
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** The ID of the parent entity that this branding applies to. The parent entity can be either a network or an organization. If the `ParentId` references a network ID, the branding applies to that network (community) only. If the `ParentId` references an organization ID, the branding applies to the organization when it is accessed through Salesforce1. Label is `Branded Entity ID`.

Usage

Use this object along with CustomBrandAsset to apply a custom branding scheme to your community. The branding scheme for the community shows in both the user interface and in Salesforce1. You must have “Create and Manage Communities” to customize community branding.

You can also use this object to apply a custom branding scheme to your organization when it is accessed through Salesforce1.

See Also:

Network

CustomBrandAsset

Represents a branding element in a custom branding scheme. For example, a color, logo image, header image, or footer text. A CustomBrandAsset can apply to a community or to an organization using Salesforce1. This object is available in API version 28.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

This object is available only if Salesforce Communities is enabled in your organization.

Fields

Field Name	Details
`AssetCategory`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Values include: • `MotifZeronaryColor`—The background color for the header. Label is `Zeronary motif color`. If this CustomBrandAsset is for a network, this is the header color for the network. If it is for an organization, this is the header color when users access Salesforce1. • `MotifPrimaryColor`—The color used for the active tab. Label is `Primary motif color`. Not used for Salesforce1 branding. • `MotifSecondaryColor`—The color used for the top borders of lists and tables. Label is `Secondary motif color`. Not used for Salesforce1 branding.

Field Name	Details
	• `MotifTertiaryColor`—The background color for section headers on edit and detail pages. Label is `Tertiary motif color`. Not used for Salesforce1 branding. • `MotifQuaternaryColor`—If this CustomBrandAsset is for a network, this is the background color for network pages. If it is for an organization, this is the background color on a splash page. Label is `Quaternary motif color`. • `MotifZeronaryComplementColor`—Font color used with `zeronaryColor`. Label is `Zeronary motif colors complement color`. • `MotifPrimaryComplementColor`—Font color used with `primaryColor`. Label is `Primary motif colors complement color`. Not used for Salesforce1 branding. • `MotifTertiaryComplementColor`—Font color used with `tertiaryColor`. Label is `Tertiary motif colors complement color`. Not used for Salesforce1 branding. • `MotifQuaternaryComplementColor`—Font color used with `quaternaryColor`. Label is `Quaternary motif colors complement color`. Not used for Salesforce1 branding. • `PageHeader`—An image that appears on the header of the community pages. Can be an .html, .gif, .jpg, or .png file. Label is `Page Header`. Not used for Salesforce1 branding. • `PageFooter`—An image that appears on the footer of the community pages. Must be an .html file. Label is `Page Footer`. Not used for Salesforce1 branding. • `LoginFooterText`—The text that appears in the footer of the community login page. Label is `Footer text displayed on the login page`. Not used for Salesforce1 branding. • `LoginLogoImageId`—The logo that appears on the community login page for external users. In Salesforce1, this logo also appears on the community splash page. Label is `Logo image displayed on the login page`. • `LargeLogoImageId`—Only used for Salesforce1. The logo that appears on the splash page when you start Salesforce1. Label is `Large logo image`. • `SmallLogoImageId`—Only used for Salesforce1. The logo that appears on the publisher in Salesforce1. Label is `Small logo image`.

Field Name	Details
CustomBrandId	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the associated CustomBrand.
ForeignKeyAssetId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the document used if the `AssetCategory` is `PageHeader`, `PageFooter`, or `LoginLogoImageId`.
TextAsset	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** Text used if the `AssetCategory` is `LoginFooterText`.

Usage

Use this object to add basic branding elements—color scheme, header or footer images, login page logo, or footer text—to the branding scheme (CustomBrand) for your Network. You must have “Create and Manage Communities” to customize community branding.

If you’re using communities in Salesforce1, the loading page shows the logo.

See Also:

Network

*`Custom Object`*__Feed

Represents a single feed item on a custom object detail page.

A custom object feed shows recent changes to a custom object record for any fields that are tracked in feeds, as well as posts and comments about the object. It is a useful way to stay up-to-date with changes made to a custom object. This object is available in API version 18.0 and later. The object name is variable and has a syntax of *`Custom Object`*__Feed, where *`Custom Object`* is the `Object Name` for the custom object. For example, `Offer__Feed` represents a feed object for the custom object named Offer.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of *`Custom Object`*__Feed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64

Field	Details
	Properties Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.

Field	Details
CreatedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDate DESC` sorts the feed by the most recent feed item.
FeedPostId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
InsertedById	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
LastModifiedDate	**Type** dateTime

Field	Details
	Properties Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`:

Field	Details
	• Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the custom object that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `TrackedChange`—a change or group of changes to a tracked field • `UserStatus`—automatically generated when a user adds a post • `TextPost`—a direct text entry on a feed • `LinkPost`—a URL posting on a feed

Field	Details
	• `ContentPost`—an uploaded file on a feed • `PollPost`—a poll posted on a feed • `DashboardComponentSnapshot`—a posting of a dashboard snapshot on a feed. • `ApprovalPost`—automatically generated by a feed query on an approval item • `CollaborationGroupCreated`—automatically generated post on a user's feed when the user creates a public group • `ActivityEvent`—generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases) or when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.

Usage

Use this object to track changes for a custom object. A record of this object type is automatically created when a user enables feed tracking for a custom object.

CustomPermission

Note: Custom permissions are currently available as a Developer Preview.

Represents a permission created to control access to a custom process or app, such as sending email. This object is available in API version 31.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Description`	**Type** textarea **Properties** Filter, Group, Nillable, Sort **Description** A description of the custom permission. Limit: 255 characters.
`DeveloperName`	**Type** string **Properties** Filter, Group, Sort **Description** The unique name of the custom permission in the API. This name can contain only underscores and alphanumeric characters and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. The label corresponds to **Name** in the user interface. Limit: 80 characters. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`Language`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The language of the custom permission. Valid values are: • Chinese (Simplified) • Chinese (Traditional) • Danish • Dutch • English • Finnish • French • German • Italian • Japanese • Korean

Field Name	Details
	• Portuguese (Brazilian) • Russian • Spanish • Swedish • Thai
`MasterLabel`	**Type** string **Properties** Filter, Group, Sort **Description** The custom permission label, which corresponds to **Label** in the user interface. Limit: 80 characters.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.

Usage

Use the CustomPermission object to determine users' access to custom permissions.

For example, to query all permission sets where the Button1 permission is enabled:

```
SELECT Id, DeveloperName,
(select Id, Parent.Name, Parent.Profile.Name from SetupEntityAccessItems)
FROM CustomPermission
WHERE DeveloperName = 'Button1'
```

To query all permission sets and profiles with custom permissions:

```
SELECT Assignee.Name, PermissionSet.Id,
PermissionSet.Profile.Name,
PermissionSet.isOwnedByProfile,
PermissionSet.Label
FROM PermissionSetAssignment
WHERE PermissionSetId
IN (SELECT ParentId
   FROM SetupEntityAccess
   WHERE SetupEntityType =
'CustomPermission')
```

To query for all SetupEntityAccess rows with custom permissions:

```
SELECT Id,ParentId,Parent.Name, SetupEntityId
FROM SetupEntityAccess
WHERE SetupEntityType='CustomPermission'
AND ParentId
IN (SELECT Id
   FROM PermissionSet
   WHERE isOwnedByProfile = false)
```

See Also:

PermissionSet
Profile
SetupEntityAccess

DandBCompany

Represents a D&B Company record, which is associated with an account. This object is available in API version 25.0 and later.

Supported Calls

`delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`

Special Access Rules

Only organizations with Data.com Premium Prospector or Data.com Premium Clean can access this object.

Fields

Field Name	Details
`Address`	**Type** address **Properties** Filter, Nillable

Field Name	Details
	Description The compound form of the address. Read-only. See Address Compound Fields for details on compound address fields.
`City`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The city where a company is physically located. Maximum size is 40 characters.
`CompanyCurrencyIsoCode`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The code used to represent a company's local currency. This data is provided by the International Organization for Standardization (ISO) and is based on their three-letter currency codes. For example, USD is the ISO code for United States Dollar. Maximum size is 3 characters.
`Country`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The country where a company is physically located. Maximum size is 40 characters.
`CountryAccessCode`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The required code for international calls. Maximum size is 4 characters.
`CurrencyCode`	**Type** picklist

Field Name	Details
	Properties Filter, Group, Nillable, Restricted picklist, Sort **Description** The currency in which the company's sales volume is expressed. The full list of values can be found at the Optimizer Resources page maintained by D&B. Maximum size is 4 characters.
`Description`	**Type** textarea **Properties** Nillable **Description** A brief description of the company, which may include information about its history, its products and services, and its influence on a particular industry. Maximum size is 32000 characters.
`DomesticUltimateBusinessName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The primary name of the Domestic Ultimate, which is the highest ranking subsidiary, specified by country, within an organization's corporate structure. Maximum size is 255 characters.
`DomesticUltimateDunsNumber`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The D-U-N-S Number for the Domestic Ultimate, which is the highest ranking subsidiary, specified by country, within an organization's corporate structure. Maximum size is 9 characters.
`DunsNumber`	**Type** string **Properties** Filter, Group, Sort

Field Name	Details
	Description The Data Universal Numbering System (D-U-N-S) number is a unique, nine-digit number assigned to every business location in the D&B database that has a unique, separate, and distinct operation. D-U-N-S numbers are used by industries and organizations around the world as a global standard for business identification and tracking. Maximum size is 9 characters.
`EmployeesHere`	**Type** double **Properties** Filter, Nillable, Sort **Description** The number of employees at a specified location, such as a branch location. Maximum size is 15 characters.
`EmployeesHereReliability`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The reliability of the `EmployeesHere` figure. Available values are `Actual number`, `Low`, `Estimated (for all records)`, `Modeled (for non-US records)`. A blank value indicates this data is unavailable.
`EmployeesTotal`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total number of employees in the company, including all subsidiary and branch locations. This data is only available on records that have a value of `Headquarters/Parent` in the `LocationStatus` field. Maximum size is 15 characters.
`EmployeesTotalReliability`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field Name	Details
	Description The reliability of the `EmployeesTotal` figure. Available values are `Actual number`, `Low`, `Estimated (for all records)`, `Modeled (for non-US records)`. A blank value indicates this data is unavailable.
`FamilyMembers`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The total number of family members, worldwide, within an organization, including the Global Ultimate, its subsidiaries (if any), and its branches (if any). Maximum size is 5 characters.
`Fax`	**Type** phone **Properties** Filter, Group, Nillable, Sort **Description** The company's facsimile number.
`FifthNaics`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An additional NAICS code used to further classify an organization by industry. Maximum size is 6 characters.
`FifthNaicsDesc`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on the corresponding NAICS code. Maximum size is 120 characters.
`FifthSic`	**Type** string

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** An additional SIC code used to further classify an organization by industry. Maximum size is 4 characters.
`FifthSicDesc`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on the corresponding SIC code. Maximum size is 80 characters.
`FipsMsaCode`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The Federal Information Processing Standards (FIPS) and the Metropolitan Statistical Area (MSA) codes identify the organization's location. The MSA codes are defined by the US Office of Management and Budget. Maximum size is 5 characters.
`FipsMsaDesc`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's FIPS MSA code. Maximum size is 255 characters.
`FourthNaics`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An additional NAICS code used to further classify an organization by industry. Maximum size is 6 characters.

Field Name	Details
FourthNaicsDesc	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on the corresponding NAICS code. Maximum size is 120 characters.
FourthSic	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An additional SIC code used to further classify an organization by industry. Maximum size is 4 characters.
FourthSicDesc	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on the corresponding SIC code. Maximum size is 80 characters.
GeoCodeAccuracy	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The level of accuracy of a location's geographical coordinates compared with its physical address. Available values include `Rooftop level`, `Street level`, `Block level`, `Census tract level`, and `Mailing address level`.
GlobalUltimateBusinessName	**Type** string **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description The primary name of the Global Ultimate, which is the highest entity within an organization's corporate structure and may oversee branches and subsidiaries. Maximum size is 255 characters.
`GlobalUltimateDunsNumber`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The D-U-N-S Number of the Global Ultimate, which is the highest entity within an organization's corporate structure and may oversee branches and subsidiaries. Maximum size is 9 characters.
`GlobalUltimateTotalEmployees`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total number of employees at the Global Ultimate, which is the highest entity within an organization's corporate structure and may oversee branches and subsidiaries. Maximum size is 15 characters.
`ImportExportAgent`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Identifies whether a business imports goods or services, exports goods or services, and/or is an agent for goods.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`Latitude`	**Type** string

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** Used with longitude to specify a precise location, which is then used to assess the Geocode Accuracy. Maximum size is 11 characters.
`LegalStatus`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Identifies the legal structure of an organization. Available values include `Cooperative`, `Nonprofit organization`, `Local government body`, `Partnership of unknown type`, and `Foreign company`.
`LocationStatus`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Identifies the organizational status of a company. Available values are `Single location`, `Headquarters/Parent`, and `Branch`.
`Longitude`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Used with latitude to specify a precise location, which is then used to assess the Geocode Accuracy. Maximum size is 11 characters.
`MailingAddress`	**Type** address **Properties** Filter, Nillable **Description** The compound form of the mailing address. Read-only. See Address Compound Fields for details on compound address fields.

Field Name	Details
MailingCity	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The city where a company has its mail delivered. Maximum size is 40 characters.
MailingCountry	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The country where a company has its mail delivered. Maximum size is 40 characters.
MailingPostalCode	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The postal code that a company uses on its mailing address. Maximum size is 20 characters.
MailingState	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The state where a company has its mail delivered. Maximum size is 20 characters.
MailingStreet	**Type** textarea **Properties** Filter, Group, Nillable, Sort **Description** The street address where a company has its mail delivered. Maximum size is 255 characters.

Field Name	Details
MarketingPreScreen	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The probability that a company will pay with a significant delay compared to the agreed terms. The risk level is based on the standard Commercial Credit Score, and ranges from low risk to high risk. Available values are `High risk of delinquency`, `Low risk of delinquency`, and `Moderate risk of delinquency`.
MarketingSegmentationCluster	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Twenty-two distinct, mutually exclusive profiles, created as a result of cluster analysis of D&B data for US organizations. Available values include `High-Tension Branches of Insurance/Utility Industries`, `Rapid-Growth Large Businesses`, `Labor-Intensive Giants`, `Spartans`, `Main Street USA`.
MinorityOwned	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Indicates whether an organization is owned or controlled by a member of a minority group.
Name	**Type** string **Properties** Filter, Group, Sort **Description** The primary or registered name of a company. Maximum size is 255 characters.
NationalId	**Type** string

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** The identification number used in some countries for business registration and tax collection. Maximum size is 255 characters.
`NationalIdType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** A code value that identifies the type of national identification number used. The full list of values can be found at the Optimizer Resources page maintained by D&B. Maximum size is 5 characters.
`OutOfBusiness`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Indicates whether the company at the specified address has discontinued operations.
`OwnOrRent`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Indicates whether a company owns or rents the building it occupies.
`ParentOrHqBusinessName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The primary name of the parent or headquarters company. Maximum size is 255 characters.
`ParentOrHqDunsNumber`	**Type** string

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** The D-U-N-S Number for the parent or headquarters. Maximum size is 9 characters.
`Phone`	**Type** phone **Properties** Filter, Group, Nillable, Sort **Description** A company's primary telephone number.
`PostalCode`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The postal code that corresponds to a company's physical location. Maximum size is 20 characters.
`PrimaryNaics`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The six-digit North American Industry Classification System (NAICS) code is the standard used by business and government to classify business establishments according to their economic activity for the purpose of collecting, analyzing, and publishing statistical data related to the US business economy. The full list of values can be found at the Optimizer Resources page maintained by D&B. Maximum size is 6 characters.
`PrimaryNaicsDesc`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on its NAICS code. Maximum size is 120 characters.

Field Name	Details
PrimarySic	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The four-digit Standard Industrial Classification (SIC) code is used to categorize business establishments by industry. The full list of values can be found at the Optimizer Resources page maintained by D&B. Maximum size is 4 characters.
PrimarySicDesc	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on its SIC code. Maximum size is 80 characters.
PublicIndicator	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Indicates whether ownership of the company is public or private.
SalesVolume	**Type** double **Properties** Filter, Nillable, Sort **Description** The total annual sales revenue in local currency. D&B tracks revenue data for publicly traded companies, Global Ultimates, Domestic Ultimates, and some headquarters.
SalesVolumeReliability	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field Name	Details
	Description The reliability of the `SalesVolume` figure.
`SecondNaics`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An additional NAICS code used to further classify an organization by industry. Maximum size is 6 characters.
`SecondNaicsDesc`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on the corresponding NAICS code. Maximum size is 120 characters.
`SecondSic`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An additional SIC code used to further classify an organization by industry. Maximum size is 4 characters.
`SecondSicDesc`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on the corresponding SIC code. Maximum size is 80 characters.
`SixthNaics`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description An additional NAICS code used to further classify an organization by industry. Maximum size is 6 characters.
`SixthNaicsDesc`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on the corresponding NAICS code. Maximum size is 120 characters.
`SixthSic`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An additional SIC code used to further classify an organization by industry. Maximum size is 4 characters.
`SixthSicDesc`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization's line of business, based on the corresponding SIC code. Maximum size is 80 characters.
`SmallBusiness`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Indicates whether the company is designated a small business as defined by the Small Business Administration of the US government.
`State`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description The state where a company is physically located. Maximum size is 20 characters.
`StockExchange`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The corresponding exchange for a company's stock symbol. For example: NASDAQ or NYSE. Maximum size is 16 characters.
`StockSymbol`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The abbreviation used to identify publicly traded shares of a particular stock. Maximum size is 6 characters.
`Street`	**Type** textarea **Properties** Filter, Group, Nillable, Sort **Description** The street address where a company is physically located. Maximum size is 255 characters.
`Subsidiary`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Indicates whether a company is more than 50 percent owned by another organization.
`ThirdNaics`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description An additional NAICS code used to further classify an organization by industry. Maximum size is 6 characters.
`ThirdNaicsDesc`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization’s line of business, based on the corresponding NAICS code. Maximum size is 120 characters.
`ThirdSic`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An additional SIC code used to further classify an organization by industry. Maximum size is 4 characters.
`ThirdSicDesc`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of an organization’s line of business, based on the corresponding SIC code. Maximum size is 80 characters.
`TradeStyle1`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A name, different from its legal name, that an organization may use for conducting business. Similar to “Doing business as” or “DBA”. Maximum size is 255 characters.
`TradeStyle2`	**Type** string

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** An additional tradestyle used by the organization. Maximum size is 255 characters.
`TradeStyle3`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An additional tradestyle used by the organization. Maximum size is 255 characters.
`TradeStyle4`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An additional tradestyle used by the organization. Maximum size is 255 characters.
`TradeStyle5`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An additional tradestyle used by the organization. Maximum size is 255 characters.
`URL`	**Type** url **Properties** Filter, Group, Nillable, Sort **Description** An organization's primary website address. Maximum size is 104 characters.
`UsTaxId`	**Type** string

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** The identification number for the company used by the Internal Revenue Service (IRS) in the administration of tax laws. Also referred to as Federal Taxpayer Identification Number. Maximum size is 9 characters.
`WomenOwned`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Indicates whether a company is more than 50 percent owned or controlled by a woman.
`YearStarted`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The year the company was established or the year when current ownership or management assumed control of the company. Maximum size is 4 characters.

Usage

Use this object to manage D&B Company records in your organization. These fields are read-only (except for Data.com Clean updates).

Dashboard

Represents a dashboard, which shows data from custom reports as visual components. Access is read-only. This object is available in API version 20.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`, `search()`

Fields

Field	Details
`BackgroundDirection`	**Type** picklist

Field	Details
	Properties Filter, Group, Restricted picklist, Sort **Description** Returns the direction of the background fade. Available values are: • `Top to Bottom` • `Left to Right` • `Diagonal` (default value) Label is `Background Fade Direction`.
`BackgroundEnd`	**Type** int **Properties** Filter, Group, Sort **Description** Returns the ending fade color in hexadecimal. Label is `Ending Color`.
`BackgroundStart`	**Type** int **Properties** Filter, Group, Sort **Description** Returns the starting fade color in hexadecimal. Label is `Starting Color`.
`Description`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Returns the description of the dashboard. Limit: 255 characters.
`DeveloperName`	**Type** string **Properties** Filter, Group, Sort **Description** Required. The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not

Field	Details
	end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Label is `Dashboard Unique Name`. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`FolderId`	**Type** reference **Properties** Filter, Group, Sort **Description** Required. Returns the ID of the Folder that contains the dashboard. See Folder.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is `Deleted`.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update

Field	Details
	Description The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`LeftSize`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Returns the size of the left column of the dashboard. Available values are: • `Narrow` • `Medium` • `Wide`
`MiddleSize`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Returns the size of the middle column of the dashboard. Available values are: • `Narrow` • `Medium` • `Wide`
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values:

Field	Details
	• In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.
`RightSize`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Returns the size of the right column in the dashboard. Available values are: • `Narrow` • `Medium` • `Wide`
`RunningUserId`	**Type** reference **Properties** Filter, Group, Sort **Description** Returns the ID of the running user specified for the dashboard. If the dashboard is dynamic, it returns the user ID of the viewing user.
`TextColor`	**Type** int **Properties** Filter, Group, Sort **Description** Returns the body text color in hexadecimal. Label is `Text Color`.
`Title`	**Type** string **Properties** Filter, Group, Sort

Field	Details
	Description Returns the title of the dashboard. Limit: 80 characters.
`TitleColor`	**Type** int **Properties** Filter, Group, Sort **Description** Returns the title text color in hexadecimal. Label is `Title Color`.
`TitleSize`	**Type** int **Properties** Filter, Group, Sort **Description** Returns the title font size in points. Label is `Title Size`.
`Type`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Returns the dashboard type. Available values are: • `SpecificUser`—The dashboard displays data according to the access level of one specific running user. • `LoggedInUser`—The dashboard displays data according to the access level of the logged-in user. • `MyTeamUser`—The dashboard displays data according to the access level of the logged-in user, and managers can view dashboards from the point of view of users beneath them in the role hierarchy.

Usage

Provides read only access to the current values in the dashboard fields.

See Also:

DashboardFeed
DashboardTag
Report

DashboardComponent

Represents a dashboard component, which can be a chart, metric, table, or gauge on a dashboard. Access is read-only. This object is available in API version 21.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`DashboardId`	**Type** reference **Properties** Filter, Group, Sort **Description** Returns the ID of the dashboard that contains the component. See Dashboard.
`Name`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Returns the name of the dashboard component.

Usage

Provides read only access to the current values in dashboard component fields.

DashboardComponentFeed

Represents a single feed item in the feed displayed on a dashboard component. This object is available in API version 21.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Body`	**Type** textarea

Field	Details
	Properties Create, Filter, Nillable, Sort **Description** The content of the FeedItem. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ContentData`	**Type** base64 **Properties** Create, Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Create, Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Create, Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int

Field	Details
	Properties Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDate DESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Create, Filter, Nillable, Sort **Description** The URL of a `LinkPost`.

Field	Details
NetworkScope	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. NetworkScope can have the following values: • NetworkId—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • AllNetworks—The feed item is available in all communities. Note the following exceptions for NetworkScope: • Only feed items with a Group or User parent can set a NetworkId or a null value for NetworkScope. • For feed items with a record parent, users can set NetworkScope only to AllNetworks. • You can't filter a FeedItem on the NetworkScope field.
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the dashboard that is tracked in the feed. The feed is displayed on the detail page for this record.
RelatedRecordId	**Type** reference **Properties** Create, Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a ContentPost. This field is null for all posts except ContentPost. For example, set this field to an existing ContentVersion and post it to a feed as a FeedItem object of Type ContentPost.
Title	**Type** string

Field	Details
	Properties Create, Filter, Group, Nillable, Sort **Description** The title of the FeedItem. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Create,Filter, Group **Description** The type of FeedItem. With the exception of `ContentPost`, `LinkPost`, and `TextPost`, FeedItem types should not be created directly from the API. • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated.

Field	Details
	The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case. **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this to retrieve the current contents of the feed fields, such as type of feed or feed ID.

DashboardFeed

Represents a single feed item in the feed displayed on a dashboard.

Shows changes to dashboard fields tracked in feeds, as well as posts and comments about the dashboard. This object is available in API version 20.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of DashboardFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string

Field	Details
	Properties Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization.

Field	Details
	NetworkScope can have the following values: • NetworkId—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • AllNetworks—The feed item is available in all communities. Note the following exceptions for NetworkScope: • Only feed items with a Group or User parent can set a NetworkId or a null value for NetworkScope. • For feed items with a record parent, users can set NetworkScope only to AllNetworks. • You can't filter a FeedItem on the NetworkScope field.
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the dashboard that is tracked in the feed. The feed is displayed on the detail page for this record.
RelatedRecordId	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a ContentPost. This field is null for all posts except ContentPost.
Title	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the Type is LinkPost, the LinkUrl is the URL and this field is the link name.
Type	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field	Details
	Description The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.

Field	Details
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this to retrieve the current contents of the feed fields, such as type of feed or feed ID.

See Also:

Dashboard

DashboardTag

Associates a word or short phrase with a Dashboard. This object is available in API version 20.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter

Field Name	Details
	Description ID of the tagged item.
Name	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
TagDefinitionId	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.
Type	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

DashboardTag stores the relationship between its parent TagDefinition and the Dashboard being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

See Also:

Dashboard

DatacloudCompany

Represents the fields for Data.com company records. This object is available in API version 30.0 or later.

Supported Calls

`describeSObjects()`, `query()`

Fields

Field Name	Details
`ActiveContacts`	**Type** int **Properties** Nillable **Description** The number of active contacts that are associated with a company.
`AnnualRevenue`	**Type** currency **Properties** Filter, Nillable **Description** The amount of money that the company makes in one year. Annual revenue is measured in US dollars.
`City`	**Type** string **Properties** Filter, Nillable, Sort **Description** The name of the city where the company is located.
`CompanyId`	**Type** string **Properties** Filter, Nillable

Field Name	Details
	Description A unique numerical identifier for the company and theData.com identifier for a company.
`Country`	**Type** string **Properties** Filter, Nillable, Sort **Description** A string that represents the standard abbreviation for the country where the company is located.
`Description`	**Type** string **Properties** Nillable **Description** A brief synopsis of the company that provides a general overview of the company and what it does.
`DunsNumber`	**Type** string **Properties** Filter, Nillable **Description** A randomly generated nine-digit number that's assigned by Dun & Bradstreet (D&B) to identify unique business establishments.
`ExternalId`	**Type** string **Properties** Filter, Nillable, Sort **Description** A unique numerical identifier for the company. This is a system-generated number.

Field Name	Details
Fax	**Type** phone **Properties** Nillable **Description** The telephone number that's used to send and receive faxes.
Industry	**Type** string **Properties** Nillable **Description** A description of the type of industry such as "Telecommunications," "Agriculture," or "Electronics."
IsInactive	**Type** boolean **Properties** Defaulted on create, Filter **Description** A true or false response. • `false` = active • `true` = not active
IsOwned	**Type** boolean **Properties** Defaulted on create **Description** Whether your organization owns this DatacloudCompnay record. • `true` = own record • `false` = not yet purchased
NaicsCode	**Type** string

Field Name	Details
	Properties Filter, Nillable **Description** A value that represents the North American Industry Classification System (NAICS) code. NAICS was created to provide details about a business's service orientation. The code descriptions are focused on what a business does.
NaicsDesc	**Type** string **Properties** Nillable **Description** A description of the NAICS classification.
Name	**Type** string **Properties** Filter, Nillable, Sort **Description** The company's name.
NumberOfEmployees	**Type** int **Properties** Filter, Nillable **Description** The number of employees working for the company.
Ownership	**Type** string **Properties** Filter, Nillable **Description** The type of ownership of the company:

Field Name	Details
	• `Public` • `Private` • `Government` • `Other`
`Phone`	**Type** phone **Properties** Nillable **Description** A numeric string containing the primary telephone number for the company.
`Sic`	**Type** string **Properties** Filter, Nillable **Description** A value that represent the Standard Industrial Codes (SIC). SIC is a numbering convention that indicates what type of service a business provides. It is a four-digit value.
`SicDesc`	**Type** string **Properties** Nillable **Description** A description of the SIC classification.
`Site`	**Type** string **Properties** Filter, Nillable **Description** An organizational status of the company. • Branch: a secondary location to a headquarter location • Headquarter: a parent company with branches or subsidiaries

Field Name	Details
	• Single Location: a single business with no subsidiaries or branches
`State`	**Type** string **Properties** Filter, Nillable, Sort **Description** The two-letter standard abbreviation for a state.
`Street`	**Type** string **Properties** Nillable **Description** A postal address for the company.
`TickerSymbol`	**Type** string **Properties** Nillable **Description** The symbol that uniquely identifies companies that are traded on public stock exchanges.
`TradeStyle`	**Type** string **Properties** Nillable **Description** A legal name under which a company conducts business.
`UpdatedDate`	**Type** dateTime **Properties** Nillable

Field Name	Details
	Description The last date and time when the information for this company was updated.
`Website`	**Type** url **Properties** Nillable **Description** The standard URL for the company's home page.
`YearStarted`	**Type** string **Properties** Nillable **Description** The year when the company was founded.
`Zip`	**Type** string **Properties** Filter, Nillable **Description** A numeric postal code that's designated for the address.

Usage

Use the DatacloudCompany object to search the Data.com database for companies with the specific criteria that you enter. Use this object to find company records that you are interested in purchasing for your organization. Data.com APIs use the term "company," which is similar to Salesforce term "accounts."

DatacloudContact

The fields and properties for Data.com contact records. This object is available in API version 30.0 or later.

Supported Calls

`describeSObjects()`, `query()`

Fields

Field Name	Details
`City`	**Type** string **Properties** Filter, Nillable, Sort **Description** The city where the company is located.
`CompanyId`	**Type** string **Properties** Filter, Nillable **Description** The unique numerical identifier for the company and the Data.com company identification number or Data.com Key.
`CompanyName`	**Type** string **Properties** Filter, Sort **Description** The name of the company.
`ContactId`	**Type** string **Properties** Filter, Nillable **Description** The unique numeric identifier for this contact.
`Country`	**Type** string **Properties** Filter, Nillable, Sort

Field Name	Details
	Description The standard abbreviation or name for the country where the company is located. **Note:** You can enter a comma-separated list of countries; however, for a country that uses a comma in its name, leave out the comma. For example, `Taiwan, ROC` should be entered as `Taiwan, ROC`.
`Department`	**Type** string **Properties** Filter, Nillable **Description** The department in the company that the contact is affiliated with.
`Email`	**Type** email **Properties** Filter, Nillable **Description** A business email address for the contact.
`ExternalId`	**Type** string **Properties** Filter, Nillable, Sort **Description** A unique system-generated numerical identifier for the contact.
`FirstName`	**Type** string **Properties** Filter, Nillable **Description** The first name of the contact.

Field Name	Details
`IsInactive`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Whether the record is active (false) or not (true).
`IsOwned`	**Type** boolean **Properties** Defaulted on create **Description** • `True`: You own this record. • `False`: You do not own this record.
`LastName`	**Type** string **Properties** Filter, Nillable, Sort **Description** The last name of the contact.
`Level`	**Type** string **Properties** Filter, Nillable **Description** A human resource label that designates a person’s level in the company.
`Phone`	**Type** phone **Properties** Nillable **Description** The direct-dial telephone number for the contact.

Field Name	Details
SocialHandles	**Type** string **Properties** The social handles for this contact. Social handles are a normalized URL and user name for social media accounts such as, LinkedIn, Facebook, and Twitter. This is a response-only field. The DatacloudSocialHandles object is a child of the DatacloudContact object.
State	**Type** string **Properties** Filter, Nillable, Sort **Description** The state where the company is located, which can also be a province or other equivalent to a state, depending on the country where the company is located.
Street	**Type** string **Properties** Nillable **Description** The street address for the company where the contact works.
Title	**Type** string **Properties** Filter, Nillable, Sort **Description** Title of the contact such as CEO or Vice President.
UpdatedDate	**Type** dateTime **Properties** Nillable

Field Name	Details
	Description The last date and time when the information for a contact was updated.
`Zip`	**Type** string **Properties** Filter, Nillable **Description** The postal or zip code for the address.

Usage

This object searches the Data.com database for contacts with the specific criteria that you enter. Use this object to find contact records that you are interested in purchasing for your organization.

DatacloudDandBCompany

Represents a set of read-only fields that are used to return D&B company data from Data.com API calls. This object is available in API version 30.0 or later.

Supported Calls

`describeSObjects()`, `query()`

Fields

Field Name	Details
`City`	**Type** string **Properties** Nillable **Description** The name of the city where the company is physically located.
`CompanyCurrencyIsoCode`	**Type** picklist **Properties** Nillable, Restricted picklist

Field Name	Details
	Description The code used to represent a company's local currency. This data is provided by the International Organization for Standardization (ISO) and is based on their three-letter currency codes. For example, USD is the ISO code for United States Dollar.
`CompanyId`	**Type** string **Properties** Filter, Nillable, Sort **Description** A unique numeric identifier for a company.
`Country`	**Type** string **Properties** Nillable **Description** The country where a company is physically located.
`CountryAccessCode`	**Type** string **Properties** Nillable **Description** The required code for international calls.
`CurrencyCode`	**Type** picklist **Properties** Nillable, Restricted picklist **Description** The currency in which the company's sales volume is expressed.
`Description`	**Type** string

Field Name	Details
	Properties Nillable **Description** A brief description of the company, which may include information about its history, its products and services, and its influence on a particular industry.
`DomesticUltimateBusinessName`	**Type** string **Properties** Nillable **Description** The primary name of the Domestic Ultimate, which is the highest ranking subsidiary, specified by country, within an organization's corporate structure.
`DomesticUltimateDunsNumber`	**Type** string **Properties** Nillable **Description** The D-U-N-S number for the Domestic Ultimate, which is the highest-ranking subsidiary, specified by country, within an organization's corporate structure.
`DunsNumber`	**Type** string **Properties** Filter, Nillable **Description** The Data Universal Numbering System (D-U-N-S) number is a unique, nine-digit number assigned to every business location in the D&B database that has a unique, separate, and distinct operation. D-U-N-S numbers are used by industries and organizations around the world as a global standard for business identification and tracking.
`EmployeesHere`	**Type** double

Field Name	Details
	Properties Nillable **Description** The number of employees at a specified location, such as a branch location.
EmployeesHereReliability	**Type** picklist **Properties** Nillable, Restricted picklist **Description** The reliability of the EmployeesHere figure. Available values are Actual number, Low, Estimated (for all records), Modeled (for non-US records). A blank value indicates this data is unavailable.
EmployeesTotal	**Type** double **Properties** Nillable **Description** The total number of employees in the company, including all subsidiary and branch locations. This data is available only on records that have a value of Headquarters/Parent in the LocationStatus field.
EmployeesTotalReliability	**Type** picklist **Properties** Nillable, Restricted picklist **Description** The reliability of the EmployeesTotal figure. Available values are Actual number, Low, Estimated (for all records), Modeled (for non-US records). A blank value indicates this data is unavailable.
ExternalId	**Type** string

Field Name	Details
	Properties Filter, Nillable, Sort **Description** A system generated numeric identification.
`FamilyMembers`	**Type** int **Properties** Nillable **Description** The total number of family members, worldwide, within an organization, including the Global Ultimate, its subsidiaries (if any), and its branches (if any).
`Fax`	**Type** phone **Properties** Nillable **Description** The company's facsimile number.
`FifthNaics`	**Type** string **Properties** Nillable **Description** An additional NAICS code that's used to further classify an organization by industry.
`FifthNaicsDesc`	**Type** string **Properties** Nillable **Description** A brief description of an organization's line of business, based on the corresponding NAICS code.

Field Name	Details
FifthSic	**Type** string **Properties** Nillable **Description** An additional Standard Industrial Classificatoin (SIC) code that's used to further classify an organization by industry.
FifthSicDesc	**Type** string **Properties** Nillable **Description** A brief description of an organization's line of business, based on the corresponding SIC code.
FipsMsaCode	**Type** string **Properties** Nillable **Description** The Federal Information Processing Standards (FIPS) and the Metropolitan Statistical Area (MSA) codes identify the organization's location. The MSA codes are defined by the US Office of Management and Budget.
FipsMsaDesc	**Type** string **Properties** Nillable **Description** A brief description of an organization's FIPS MSA code.
FourthNaics	**Type** string

Field Name	Details
	Properties Nillable **Description** An additional NAICS code used to further classify an organization by industry.
`FourthNaicsDesc`	**Type** string **Properties** Nillable **Description** A brief description of an organization's line of business, based on the corresponding NAICS code.
`FourthSic`	**Type** string **Properties** Group, Nillable **Description** An additional SIC code used to further classify an organization by industry.
`FourthSicDesc`	**Type** string **Properties** Nillable **Description** A brief description of an organization's line of business, based on the corresponding SIC code.
`GeoCodeAccuracy`	**Type** picklist **Properties** Nillable, Restricted picklist **Description** The level of accuracy of a location's geographical coordinates compared with its physical address. Available values include `Rooftop level`,

Field Name	Details
	`Street level`, `Block level`, `Census tract level`, and `Mailing address level`.
`GlobalUltimateBusinessName`	**Type** string **Properties** Nillable **Description** The primary name of the Global Ultimate, which is the highest entity within an organization’s corporate structure and may oversee branches and subsidiaries.
`GlobalUltimateDunsNumber`	**Type** string **Properties** Filter, Nillable **Description** The D-U-N-S number of the Global Ultimate, which is the highest-ranking entity within an organization’s corporate structure and might oversee branches and subsidiaries.
`GlobalUltimateTotalEmployees`	**Type** double **Properties** Nillable **Description** The total number of employees at the Global Ultimate, which is the highest entity within an organization’s corporate structure and may oversee branches and subsidiaries.
`ImportExportAgent`	**Type** picklist **Properties** Nillable, Restricted picklist **Description** Identifies whether a business imports goods or services, exports goods or services, and/or is an agent for goods.

Field Name	Details
`IsOwned`	**Type** boolean **Properties** Defaulted on create **Description** Whether your organization owns this DatacloudCompnay record. • `true` = own record • `false` = not yet purchased
`Latitude`	**Type** string **Properties** Nillable **Description** Used with longitude to specify a precise location, which is used to assess the Geocode Accuracy.
`LegalStatus`	**Type** picklist **Properties** Nillable, Restricted picklist **Description** Identifies the legal structure of an organization. Available values include `Cooperative`, `Nonprofit organization`, `Local government body`, `Partnership of unknown type`, and `Foreign company`.
`LocationStatus`	**Type** picklist **Properties** Nillable, Restricted picklist **Description** Identifies the organizational status of a company. Available values are `Single location`, `Headquarters/Parent`, and `Branch`.
`Longitude`	**Type** string

Field Name	Details
	Properties Nillable **Description** Used with latitude to specify a precise location, which is used to assess the Geocode Accuracy.
`MailingCity`	**Type** string **Properties** Nillable **Description** The city where a company has its mail delivered.
`MailingCountry`	**Type** string **Properties** Nillable **Description** The country where a company has its mail delivered.
`MailingState`	**Type** string **Properties** Nillable **Description** The state where a company has its mail delivered.
`MailingStreet`	**Type** string **Properties** Nillable **Description** The street address where a company has its mail delivered.

Field Name	Details
MailingZip	**Type** string **Properties** Nillable **Description** The postal zip code for the company.
MarketingPreScreen	**Type** picklist **Properties** Nillable, Restricted picklist **Description** The probability that a company will pay with a significant delay compared to the agreed terms. The risk level is based on the standard Commercial Credit Score, and ranges from low risk to high risk. Available values are `High risk of delinquency`, `Low risk of delinquency`, and `Moderate risk of delinquency`.
MarketingSegmentationCluster	**Type** picklist **Properties** Nillable, Restricted picklist **Description** Twenty-two distinct, mutually exclusive profiles, created as a result of cluster analysis of D&B data for US organizations. Available values include `High-Tension Branches of Insurance/Utility Industries`, `Rapid-Growth Large Businesses`, `Labor-Intensive Giants`, `Spartans`, `Main Street USA`.
MinorityOwned	**Type** picklist **Properties** Nillable, Restricted picklist **Description** Indicates whether an organization is owned or controlled by a member of a minority group.

Field Name	Details
Name	**Type** string **Properties** Nillable **Description** The primary or registered name of a company.
NationalId	**Type** string **Properties** Nillable **Description** The identification number used in some countries for business registration and tax collection.
NationalIdType	**Type** picklist **Properties** Nillable, Restricted picklist **Description** A code value that identifies the type of national identification number that's used.
OutOfBusiness	**Type** picklist **Properties** Nillable, Restricted picklist **Description** Indicates whether the company at the specified address has discontinued operations.
OwnOrRent	**Type** picklist **Properties** Nillable, Restricted picklist

Field Name	Details
	Description Indicates whether a company owns or rents the building it occupies.
`ParentOrHqBusinessName`	**Type** string **Properties** Nillable **Description** The primary name of the parent or headquarters company.
`ParentOrHqDunsNumber`	**Type** string **Properties** Nillable **Description** The D-U-N-S number for the parent or headquarters.
`Phone`	**Type** phone **Properties** Nillable **Description** A company's primary telephone number.
`PrimaryNaics`	**Type** string **Properties** Nillable **Description** The six-digit North American Industry Classification System (NAICS) code is the standard used by business and government to classify business establishments according to their economic activity for the purpose of collecting, analyzing, and publishing statistical data related to the US business economy.
`PrimaryNaicsDesc`	**Type** string

Field Name	Details
	Properties Nillable **Description** A brief description of an organization's line of business, based on its NAICS code.
`PrimarySic`	**Type** string **Properties** Nillable **Description** The four-digit SIC code that's used to categorize business establishments by industry.
`PrimarySicDesc`	**Type** string **Properties** Nillable **Description** A brief description of an organization's line of business, based on its SIC code.
`PublicIndicator`	**Type** picklist **Properties** Nillable, Restricted picklist **Description** Indicates whether ownership of the company is public or private.
`SalesVolume`	**Type** double **Properties** Nillable

Field Name	Details
	Description The total annual sales revenue in local currency. D&B tracks revenue data for publicly traded companies, Global Ultimates, Domestic Ultimates, and some headquarters.
`SalesVolumeReliability`	**Type** picklist **Properties** Nillable, Restricted picklist **Description** The reliability of the `SalesVolume` figure.
`SecondNaics`	**Type** string **Properties** Nillable **Description** An additional NAICS code that's used to further classify an organization by industry.
`SecondNaicsDesc`	**Type** string **Properties** Nillable **Description** A brief description of an organization's line of business, based on the corresponding NAICS code.
`SecondSic`	**Type** string **Properties** Nillable **Description** An additional SIC code used to further classify an organization by industry.

Field Name	Details
`SecondSicDesc`	**Type** string **Properties** Nillable **Description** A brief description of an organization's line of business, based on the corresponding SIC code.
`SixthNaics`	**Type** string **Properties** Nillable **Description** An additional NAICS code that's used to further classify an organization by industry.
`SixthNaicsDesc`	**Type** string **Properties** Nillable **Description** A brief description of an organization's line of business, based on the corresponding SIC code.
`SixthSic`	**Type** string **Properties** Nillable **Description** An additional SIC code used to further classify an organization by industry.
`SixthSicDesc`	**Type** string **Properties** Nillable

Field Name	Details
	Description A brief description of an organization's line of business, based on the corresponding SIC code.
`SmallBusiness`	**Type** picklist **Properties** Nillable, Restricted picklist **Description** Indicates whether the company is designated a small business as defined by the Small Business Administration of the US government.
`State`	**Type** string **Properties** Nillable **Description** Where a company is physically located.
`StockExchange`	**Type** string **Properties** Nillable **Description** The corresponding exchange for a company's stock symbol, for example, NASDAQ or NYSE.
`StockSymbol`	**Type** string **Properties** Nillable **Description** The abbreviation that's used to identify publicly traded shares of a particular stock.

Field Name	Details
Street	**Type** string **Properties** Nillable **Description** The street address where a company is physically located.
Subsidiary	**Type** picklist **Properties** Nillable, Restricted picklist **Description** Indicates whether a company is more than 50 percent owned by another organization.
ThirdNaics	**Type** string **Properties** Nillable **Description** An additional NAICS code that's used to further classify an organization by industry.
ThirdNaicsDesc	**Type** string **Properties** Nillable **Description** A brief description of an organization's line of business, based on the corresponding NAICS code.
ThirdSic	**Type** string **Properties** Nillable

Field Name	Details
	Description An additional SIC code that's used to further classify an organization by industry.
ThirdSicDesc	**Type** string **Properties** Nillable **Description** A brief description of an organization's line of business, based on the corresponding SIC code.
TradeStyle1	**Type** string **Properties** Nillable **Description** A name, different from its legal name, that an organization may use for conducting business. Similar to "Doing business as" or "DBA".
TradeStyle2	**Type** string **Properties** Nillable **Description** An additional tradestyle used by the organization.
TradeStyle3	**Type** string **Properties** Nillable **Description** An additional tradestyle that's used by the organization.
TradeStyle4	**Type** string

Field Name	Details
	Properties Nillable **Description** An additional tradestyle used by the organization.
`TradeStyle5`	**Type** string **Properties** Nillable **Description** An additional tradestyle used by the organization.
`URL`	**Type** url **Properties** Nillable **Description** An organization's primary website address.
`UsTaxId`	**Type** string **Properties** Nillable **Description** The identification number for the company used by the Internal Revenue Service (IRS) in the administration of tax laws. Also referred to as Federal Taxpayer Identification Number.
`WomenOwned`	**Type** picklist **Properties** Nillable, Restricted picklist **Description** Indicates whether a company is more than 50 percent owned or controlled by a woman.

Field Name	Details
YearStarted	**Type** string **Properties** Nillable **Description** The year when the company was established or the year when current ownership or management assumed control of the company.
Zip	**Type** string **Properties** Nillable **Description** A five or nine-digit code that's used to help sort mail.

Usage

Use this object to return D&B Company information. These fields are read-only.

DatacloudOwnedEntity

Represents fields in the DatacloudOwnedEntity object. The DatacloudOwnedEntity object tracks user-purchased records. This object is available in API version 30.0 or later.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
DataDotComKey	**Type** string **Properties** Create, Filter, Sort **Description** The Data.com contact or company record identification number used by the DatacloudPurchaseUsage object to keep track of purchased records. This is equivalent to the Data.com record ID for a contact or company.

Field Name	Details
`DatacloudEntityType`	**Type** picklist **Properties** Create, Filter, Restricted picklist, Sort **Description** The type of Data.com record you want to purchase. • 0—contact • 1—company
`Name`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, Sort **Description** An optional field used to name your record.
`PurchaseType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** A read only field set by the API to identify the purchase type. • Added • Export • API
`PurchaseUsageId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** The unique identification number for the DatacloudPurchaseUsage object created by making a REST POST request. • 0—contact • 1—company

Field Name	Details
UserId	**Type** reference **Properties** Filter, Group, Sort **Description** A unique identifier for the user making the purchase.

Usage

The Datacloud object that tracks records that are purchased and owned by a specific user.

DatacloudPurchaseUsage

Represents an object used to identify and track Data.com record purchases. This object is available in API version 30.0 or later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
DatacloudEntityType	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The type of Data.com record you want to purchase. • 0—indicates contact entity type. • 1—indicates company entity type.
Description	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** An optional field. You can add a description for your purchase.

Field Name	Details
`Name`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, Sort **Description** An optional field used to name your record.
`PurchaseType`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A read only field set by the API to identify the purchase type. • Added • Export • API
`Usage`	**Type** double **Properties** Filter, Sort **Description** A read only field set by the API. It is used to track the points used to purchase records.
`UserId`	**Type** reference **Properties** Filter, Group, Sort **Description** A read only field set by the API that identifies the user purchasing the records.
`UserType`	**Type** picklist

Field Name	Details
	Properties Create, Filter, Group, Restricted picklist, Sort, Update **Description** A read only field set by the API with 2 user types. • Monthly Usage • List Pool User

Usage

The DatacloudPurchaseUsage object allows you to track Data.com record purchases for CRM users.

DatacloudSocialHandle

Returns normalized URLs with userids for different social media used by Data.com contacts. The DatacloudSocialHandle object is a child object of the DatacloudContact object.This object is available in API version 30.0 or later.

Supported Calls

`describeSObjects()`, `query()`

Fields

Field Name	Details
`DatacloudContactId`	**Type** string **Properties** Filter, Nillable, Sort **Description** The unique numeric identifier for a Data.com contact record.
`ProviderName`	**Type** string **Properties** Filter, Nillable, Sort **Description** The name of the social media provider.
`SocialId`	**Type** string

Field Name	Details
	Properties Filter, Nillable, Sort **Description** The normalized userid for the user on this social media.
`Url`	**Type** url **Properties** Filter, Nillable, Sort **Description** A normalized URL and userid for the website of the social media provider.

Usage

Returns social handles with Data.com contacts from the DatacloudContact object. Social handle fields are read only fields and can't be used to filter results.

DatedConversionRate

Represents the dated exchange rates used by an organization for which the multicurrency and the effective dated currency features are enabled.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`,

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`ConversionRate`	**Type** double **Properties** Filter, Update **Description** Required. Conversion rate of this currency type against the corporate currency.

Field	Details
`IsoCode`	**Type** picklist **Properties** Filter, Restricted picklist **Description** Required. ISO code of the currency. Must be one of the valid alphabetic, three-letter currency ISO codes defined by the ISO 4217 standard, such as `USD`, `GBP`, or `JPY`. Must be unique within your organization. Label is **Currency ISO Code**.
`NextStartDate`	**Type** date **Properties** Filter, Nillable **Description** Read only. The date on which the next effective dated exchange rate will start. Effectively the day after the end date for this exchange rate.
`StartDate`	**Type** date **Properties** Filter **Description** The date on which the effective dated exchange rate starts.

Usage

This object is for multicurrency organizations with advanced currency management enabled. Use this object to define the exchange rates your organization uses for a date range. This object is not available in single-currency organizations, nor is it available if the organization does not have advanced currency management enabled.

DeclinedEventRelation

Represents invitees with the status `Declined` for a given event.

This object is available in API versions 29.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`EventId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the event.
`RelationId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the invitee.
`RespondedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Indicates the most recent date and time when the invitee declined an invitation to the event.
`Response`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Indicates the content of the response field. Label is `Comment`.
`Type`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Indicates whether the invitee is a user, lead or contact, or resource.

Usage

Query invitees who have declined an invitation to an event

```
SELECT eventId, type, response FROM DeclinedEventRelation WHERE eventid='00UTD000000ZH5LA'
```

See Also:

AcceptedEventRelation
UndecidedEventRelation

Division

A logical segment of your organization's data. For example, if your company is organized into different business units, you could create a division for each business unit, such as “North America,” “Healthcare,” or “Consulting.”

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- Divisions must be enabled for your organization to access this object. To discover whether divisions have been enabled for an organization, inspect the User or Group object for the `DefaultDivision` field—if it is present, then divisions have been enabled, and this field (the field is named Division in objects other than User and Group) will be available in all relevant objects.
- Customer Portal users can’t access this object.

Fields

Field	Details
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Update **Description** Indicates whether the division is active (`true`) or not (`false`). Label is **Active**.
`IsGlobalDivision`	**Type** boolean **Properties** Defaulted on createFilter **Description** Indicates whether the division is your organization’s global default division (`true`) or not (`false`). Label is **Global Division**.

Field	Details
`Name`	**Type** string **Properties** Create, Filter, Update **Description** A descriptive name for the division. Limit: 80 characters.
`SortOrder`	**Type** int **Properties** Create, Filter, Nillable, Update **Description** The order in which this division name appears in the Division picklist field when creating or editing users in the Salesforce user interface.

Usage

The values available for that field are the global division ID for the organization, created when divisions are first enabled, and any other division IDs that have been created. The division ID associated with a user is populated in the objects owned or created by the user.

You can use the division ID to make searches, reports, and list views run more quickly and return more relevant results if an organization has very large data sets. For more information, see the Salesforce online help, in the Fields description for the object.

You can use WITH in SOSL to pre-filter results based on division. This is faster than specifying the division in a WHERE clause.

Note: The User object has a `Division` field that is unrelated to this object. The `Division` field is a standard text field similar to Company or Department that has no special properties. Do not confuse it with the `DefaultDivision` field, which does relate to this object.

See Also:

Standard and Custom Object Basics

DivisionLocalization

When the Translation Workbench is enabled for your organization, the DivisionLocalization object provides the translation of the label for a division.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- Your organization must be using Professional, Enterprise, Developer, Unlimited, or Performance Edition and be enabled for the Translation Workbench.
- To view this object, you must have the “View Setup and Configuration” permission.

Fields

Field	Details
`Language`	**Type** string **Properties** Create, Filter, Nillable, Restricted picklist **Description** The language for this translated label.
`NamespacePrefix`	**Type** string **Properties** Filter, Nillable **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.
`ParentId`	**Type** reference **Properties** Create, Filter, Nillable **Description** The ID of the Division associated with the label that is being translated.

Field	Details
`Value`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** The actual translated label for the division. Label is **Translation**.

Usage

Use this object to translate the labels of your divisions into the different languages supported by Salesforce.

Document

Represents a file that a user has uploaded. Unlike Attachment records, documents are not attached to a parent object.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

You must have the “Edit” permission on documents and the appropriate access to the Folder that contains a document in order to create or update a document in that Folder.

Fields

Field	Details
`AuthorId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the User who is responsible for the Document.
`Body`	**Type** base64 **Properties** Create, Nillable, Update **Description** Required. Encoded file data. If specified, then do not specify a URL.

Field	Details
`BodyLength`	**Type** int **Properties** Filter, Group, Sort **Description** Size of the file (in bytes).
`ContentType`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Type of content. Label is **Mime Type**. Limit: 120 characters. If the `Don't allow HTML uploads as attachments or document records` security setting is enabled for your organization, you cannot upload files with the following file extensions: `.htm`, `.html`, `.htt`, `.htx`, `.mhtm`, `.mhtml`, `.shtm`, `.shtml`, `.acgi`, `.svg`.
`Description`	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Text description of the Document. Limit: 255 characters.
`DeveloperName`	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Label is **Document Unique Name**. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.

Field	Details
`FolderId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** Required. ID of the Folder that contains the document.
`IsBodySearchable`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the contents of the object can be searched using a SOSL `FIND` call. The `ALL FIELDS` search group includes the content as a searchable field.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsInternalUseOnly`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the object is only available for internal use (`true`) or not (`false`). Label is **Internal Use Only**.
`IsPublic`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the object is available for external use (`true`) or not (`false`). Label is **Externally Available**.

Field	Details
Keywords	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Keywords. Limit: 255 characters.
LastReferencedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
LastViewedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (LastReferencedDate) and not viewed.
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of the document. Label is **Document Name**.
NamespacePrefix	**Type** string **Properties** Filter, Group, Sort, Nillable **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***namespacePrefix__componentName*** notation.

Field	Details
	The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.
`Type`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** File type of the Document. In general, the values match the file extension for the type of Document (such as pdf or jpg). Label is **File Extension**.
`Url`	**Type** string **Properties** Create, Filter, Nillable, Group, Sort, Update **Description** URL reference to the file (instead of storing it in the database). If specified, do not specify the `Body` or `BodyLength`.

Usage

When creating or updating a document, you can specify a value in either the `Body` or `Url` fields, but not both.

Encoded Data

The API sends and receives the binary file data encoded as a base64 data type. Prior to creating a record, clients must encode the binary file data as base64. Upon receiving an API response, clients must decode the base64 data to binary (this conversion is usually handled for you by the SOAP client).

Maximum Document Size

You can only create or update documents to a maximum size of 5 MB.

See Also:

Standard and Custom Object Basics

DocumentAttachmentMap

Maps the relationship between an EmailTemplate and its attachment, which is stored as a Document.

Supported Calls

`create()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can’t access this object.

Fields

Field	Details
`DocumentId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the document that this object tracks.
`DocumentSequence`	**Type** int **Properties** Create, Filter, Group, Sort, Update **Description** Represents the order that the attachments will be included in the email defined by the EmailTemplate specified by the `DocumentId`. Label is **Attachment Sequence**. The first attachment is given a value of 0, and each subsequent attachment is given a value incremented by 1.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the EmailTemplate parent. The attachment identified by `DocumentId` is attached to the EmailTemplate specified in this field.

Usage

Use this object to map the relationship of an EmailTemplate to its attachments, and to specify the order of the attachments.

See Also:

EmailTemplate

DocumentTag

Associates a word or short phrase with a Document.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.

Field Name	Details
Type	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • Public—The tag can be viewed and manipulated by all users in an organization. • Personal—The tag can be viewed or manipulated only by a user with a matching OwnerId.

Usage

DocumentTag stores the relationship between its parent TagDefinition and the Document being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

Domain

Read-only object that represents a custom Web address assigned to a site in your organization. This object is available in API version 26.0 and later.

To access this object, Force.com Sites or Site.com must be enabled for your organization.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

- Customer Portal users can't access this object.
- To view this object, you must have the "View Setup and Configuration" permission.
- Site.com Publisher users have read-only API access to the Domain and DomainSite objects.

Fields

Field	Description
Domain	**Type** string **Properties** Filter, Sort

Field	Description
	Description The branded custom Web address within the global namespace identified by this domain's type. In the Domain Name System (DNS) global namespace, this field is the custom Web address that you registered with a third-party domain name registrar. The custom Web address can be used to access the site of this domain.
`DomainType`	**Type** picklist **Properties** Filter, Group, Sort **Description** The global namespace that this custom Web address belongs to. This value is set to DNS for custom Web addresses in the global DNS. DomainType can have the following values: • Domain Name System (DNS) • Facebook Page

Usage

Use this read-only object to query the domains that are associated with each website in your organization.

DomainSite

Read-only junction object that joins together the Site and Domain objects.

This object is available in API version 26.0 and later. To access this object, Force.com Sites or Site.com must be enabled for your organization.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

- Customer Portal users can’t access this object.
- To view this object, you must have the “View Setup and Configuration” permission.
- Site.com Publisher users have read-only API access to the Domain and DomainSite objects.

Fields

Field	Description
`DomainId`	**Type** reference

Field	Description
	Properties Filter, Group, Sort **Description** The ID of the associated Domain.
`PathPrefix`	**Type** string **Properties** Filter, Group, Sort **Description** Shows where a site's root exists on a domain. Can only be set for custom Web addresses. Always begins with a `/`.
`SiteId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the associated Site.

Usage

Use this read-only object to query or retrieve information about your Force.com site.

EmailMessage

An email message.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`

Special Access Rules

Customer Portal users have read access to this object.

`update()` is supported only on records whose `Status` is `Draft`.

Fields

Field	Details
ActivityId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** ID of the activity that is associated with the email. Usually represents an open task that is created for the case owner when a new unread email message is received.
BccAddress	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** The addresses that were sent a blind carbon copy of the email.
CcAddress	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** The addresses that were sent a carbon copy of the email.
FromAddress	**Type** email **Properties** Create, Filter, Nillable, Update **Description** The address that originated the email.
FromName	**Type** string **Properties** Create, Filter, Nillable, Update **Description** The sender's name.

Field	Details
HasAttachment	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the email was sent with an attachment (`true`) or not (`false`).
Headers	**Type** textarea **Properties** Create, Nillable, Update **Description** The Internet message headers of the incoming email. Used for debugging and tracing purposes. Does not apply to outgoing emails.
HtmlBody	**Type** textarea **Properties** Create, Nillable, Update **Description** The body of the email in HTML format.
Incoming	**Type** boolean **Properties** Create, Defaulted on create, Filter **Description** Indicates whether the email was received (`true`) or sent (`false`).
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.

Field	Details
MessageDate	**Type** dateTime **Properties** Create, Filter, Nillable, Update **Description** The date the email was created.
ParentId	**Type** ID **Properties** Create, Filter **Description** ID of the Case to which the email is associated.
ReplyToEmailMessageId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** ID of the inbound or outbound EmailMessage the current EmailMessage is a reply to. It's not possible to reply to a message whose `Status` is `Draft`.
Status	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Read only. The status of the email. For example, `New`, `Draft`, `Unread`, `Replied`, "`Sent`."
Subject	**Type** string **Properties** Create, Filter, Nillable, Update **Description** The subject line of the email.
TextBody	**Type** textarea

Field	Details
	Properties Create, Nillable, Update **Description** The body of the email, in plain text format.
`ToAddress`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** The address of the email's recipient.

Usage

This object supports the Email-to-Case feature, which allows email sent to one of your company's email addresses to automatically create a new case.

See Also:

Case
Standard and Custom Object Basics

EmailServicesAddress

An email service address.

Each email service has one or more email addresses to which users can send messages for processing. An email service only processes messages it receives at one of its addresses.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`AuthorizedSenders`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update

Field	Details
	Description Configures the email service address to only accept messages from the email addresses or domains listed in this field. If the email service address receives a message from an unlisted email address or domain, the email service performs the action specified in the `AuthorizationFailureAction` field of its associated email service. Leave this field blank if you want the email service address to receive email from any email address.
`EmailDomainName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A read only field you can query that contains the system-generated domain part of this email service address. The system generates a unique domain-part for each email service address to ensure that no two email service addresses are identical.
`FunctionId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** The ID of the email service for which the email service address receives messages.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this object is active (true) or not (false).
`LocalPart`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The local-part of the email service address. For the local-part of a Salesforce email address, all alphanumeric characters are valid, plus the following special characters: `! # $ % & amp; ' * / = ? ^ _ + - ` { | } ~ ,`

Field	Details
	The dot character (.) is also valid as long as it's not the first or last character. Email addresses aren't case sensitive.
`RunAsUserId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** The user whose permissions the email service assumes when processing messages sent to this address.

Usage

This object supports the email services feature, which allows you to create automated processes that use Apex classes to process the contents, headers, and attachments of inbound email. For example, you can create an email service that automatically creates contact records based on contact information in messages.

See Also:

EmailServicesFunction

EmailServicesFunction

An email service.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`AddressInactiveAction`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates what the email service does with messages received at an email address that is inactive. One of the following values: • `0`—The system default is used.

Field	Details
	• `1`—The email service returns the message to the sender with a notification that explains why the message was rejected. • `2`—The email service deletes the message without notifying the sender. • `3`—The email service queues the message for processing in the next 24 hours. If the message is not processed within 24 hours, the email service returns the message to the sender with a notification that explains why the message was rejected.
`ApexClassId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Required. The ID of the Apex class that the email service uses to process inbound messages. This field is required for API version 12.0 and later.
`AttachmentOption`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Indicates the types of attachments the email service accepts. One of the following values: • `0`—The email service accepts the message but discards any attachment. • `1`—The email service only accepts the following types of attachments: ◊ Attachments with a Multipurpose Internet Mail Extension (MIME) type of text. ◊ Attachments with a MIME type of application/octet-stream and a file name that ends with either a .vcf or .vcs extension. These are saved as text/x-vcard and text/calendar MIME types, respectively. • `2`—The email service only accepts binary attachments, such as image, audio, application, and video files. • `3`—The email service accepts any type of attachment.
`AuthenticationFailureAction`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update

Field	Details
	Description Indicates what the email service does with messages that fail or do not support any of the authentication protocols if the `IsAuthenticationRequired` field is true. One of the following values: • `0`—The system default is used. • `1`—The email service returns the message to the sender with a notification that explains why the message was rejected. • `2`—The email service deletes the message without notifying the sender. • `3`—The email service queues the message for processing in the next 24 hours. If the message is not processed within 24 hours, the email service returns the message to the sender with a notification that explains why the message was rejected.
`AuthorizationFailureAction`	**Type** picklist **Properties** Group, Sort, Create, Filter, Nillable, Restricted picklist, Update **Description** Indicates what the email service does with messages received from senders who are not listed in the `AuthorizedSenders` field on either the email service or email service address. One of the following values: • `0`—The system default is used. • `1`—The email service returns the message to the sender with a notification that explains why the message was rejected. • `2`—The email service deletes the message without notifying the sender. • `3`—The email service queues the message for processing in the next 24 hours. If the message is not processed within 24 hours, the email service returns the message to the sender with a notification that explains why the message was rejected.
`AuthorizedSenders`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Configures the email service to only accept messages from the email addresses or domains listed in this field. If the email service receives a message from an unlisted email address or domain, the email service performs the action specified in the `AuthorizationFailureAction` field. Leave this field blank if you want the email service to receive email from any email address.
`ErrorRoutingAddress`	**Type** email

Field	Details
	Properties Create, Filter, Nillable, Sort, Update **Description** The destination email address for error notification email messages when `IsErrorRoutingEnabled` is `true`.
`FunctionInactiveAction`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates what the email service does with messages it receives when the email service itself is inactive. One of the following values: • `0`—The system default is used. • `1`—The email service returns the message to the sender with a notification that explains why the message was rejected. • `2`—The email service deletes the message without notifying the sender. • `3`—The email service queues the message for processing in the next 24 hours. If the message is not processed within 24 hours, the email service returns the message to the sender with a notification that explains why the message was rejected.
`FunctionName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The name of the email service.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this object is active (`true`) or not (`false`).
`IsAuthenticationRequired`	**Type** boolean

Field	Details
	Properties Create, Defaulted on create, Filter, Group, Sort, Update **Description** Configures the email service to verify the legitimacy of the sending server before processing a message. The email service uses the SPF, SenderId, and DomainKeys protocols to verify the sender's legitimacy: If the sending server passes at least one of these protocols and does not fail any, the email service accepts the email. If the server fails a protocol or does not support any of the protocols, the email service performs the action specified in the `AuthenticationFailureAction` field.
`IsErrorRoutingEnabled`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** When incoming email messages can't be processed, indicates whether error notification email messages are routed to a chosen address or to the senders.
`IsTextAttachmentsAsBinary`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the email service converts text attachments to binary files.
`IsTextTruncated`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** This field is deprecated. It is not available as of API version 23.0 and is deprecated and hidden in versions 17.0 through 22.0. In all API versions, the email service now accepts inbound email messages up to the 10 MB size limit, without truncating the text. Previously, it indicated whether the email service truncated and accepted email messages with HTML body text, plain body text, and text attachments over approximately 100,000 characters (`true`) or rejected these email messages and notified the sender (`false`).
`IsTlsRequired`	**Type** boolean

Field	Details
	Properties Create, Defaulted on create, Filter, Group, Sort, Update **Description** Configures the email service to use Transport Layer Security (TLS), a protocol for secure email communication, to ensure the security and authenticity of inbound email.
`OverLimitAction`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates what the email service does with messages if the total number of messages processed by all email services combined has reached the daily limit for your organization. One of the following values: • `0`—The system default is used. • `1`—The email service returns the message to the sender with a notification that explains why the message was rejected. • `2`—The email service deletes the message without notifying the sender. • `3`—The email service queues the message for processing in the next 24 hours. If the message is not processed within 24 hours, the email service returns the message to the sender with a notification that explains why the message was rejected. The system calculates the limit by multiplying the number of user licenses by 1,000.

Usage

This object supports the email services feature, which allows you to create automated processes that use Apex classes to process the contents, headers, and attachments of inbound email. For example, you can create an email service that automatically creates contact records based on contact information in messages.

See Also:

EmailServicesAddress

EmailStatus

Represents the status of email sent.

Supported Calls

`describeSObjects()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`EmailTemplateName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The name of the EmailTemplate.
`FirstOpenDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Date when the email was first opened by recipient. Label is **Date Opened**.
`LastOpenDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Date when the email was last opened by recipient.
`TaskId`	**Type** reference **Properties** Filter, Group, Sort **Description** The activity (task or event) associated with the email. Label is **Activity ID**.
`TimesOpened`	**Type** int **Properties** Filter, Group, Sort **Description** Number of times the recipient opened the email.

Field	Details
`WhoId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The contact or lead ID associated with the recipient. Label is **Contact/Lead ID**.

See Also:

EmailTemplate

EmailTemplate

Represents a template for mass email, or email sent when the activity history related list of a record is modified.

Note: You can't send a mass email using a Visualforce email template.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`ApiVersion`	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** The API version for this class. Every class has an API version specified at creation.
`Body`	**Type** textarea **Properties** Create, Nillable, Update

Field	Details
	Description Content of the email. Limit: 384 KB.
BrandTemplateId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** Required. ID of the BrandTemplate associated with this email template. The brand template supplies letterhead information for the email template.
Description	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** Description of the template, for example, Promotion Mass Mailing.
DeveloperName	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Label is **Template Unique Name**. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
Encoding	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Character set encoding for the template.

Field	Details
FolderId	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the folder that contains the template.
HtmlValue	**Type** textarea **Properties** Create, Nillable, Update **Description** This field contains the content of the email message, including HTML coding to render the email message. Limit: 384 KB.
IsActive	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates that this template is active if `true`, or inactive if `false`.
LastUsedDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Date and time when this EmailTemplate was last used.
Markup	**Type** textarea **Properties** Create, Nillable, Update **Description** The Visualforce markup, HTML, Javascript, or any other Web-enabled code that defines the content of the template.
Name	**Type** string

Field	Details
	Properties Create, Filter, Group, Sort, Update **Description** Name of the template. Label is **Email Template Name**.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects. This field cannot be accessed unless the logged-in user has the "Customize Application" permission.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the owner of the template.
`Subject`	**Type** string **Properties** Create, Group, Nillable, Sort, Update **Description** Content of the subject line.

Field	Details
`TemplateStyle`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** Style of the template, such as formalLetter or freeform.
`TemplateType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** Type of template, either HTML, text, or custom templates, or those generated by Visualforce.
`TimesUsed`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Number of times this template has been used.

Usage

To retrieve this object, issue a describe call on an object, which will return a query result for each activity since the object was created. You can't query these records.

See Also:

Attachment
EmailStatus
DocumentAttachmentMap

Entitlement

Represents the customer support an Account or Contact is eligible to receive. Entitlements may be based on an Asset, Product2, or ServiceContract.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
AccountId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the Account associated with the entitlement.
AssetId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Required. ID of the Asset associated with the entitlement. Must be a valid asset ID.
BusinessHoursId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Required. ID of the BusinessHours associated with the entitlement. Must be a valid business hours ID.
CasesPerEntitlement	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The total number of cases the entitlement supports. This field is only available if `IsPerIncident` is `true`.
ContractLineItemId	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Required. ID of the ContractLineItem associated with the entitlement. Must be a valid ID.
`EndDate`	**Type** date **Properties** Create, Filter, Nillable, Update **Description** The last day the entitlement is in effect.
`IsPerIncident`	**Type** boolean **Properties** Defaulted on create, Filter, Update **Description** Indicates whether the entitlement is limited to supporting a specific number of cases (`true`) or not (`false`).
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`Name`	**Type** string **Properties** Create, Filter, Update

Field	Details
	Description Required. Name of the entitlement.
`RemainingCases`	**Type** int **Properties** Create, Filter, Nillable, Update **Description** The number of cases the entitlement can support. This field decreases in value by one each time a case is created with the entitlement. This field is only available if `IsPerIncident` is selected.
`ServiceContractId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** Required. ID of the ServiceContract associated with the entitlement. Must be a valid ID.
`SlaProcessId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the SlaProcess associated with the entitlement. This field is available in version 19.0 and later.
`StartDate`	**Type** date **Properties** Create, Filter, Nillable, Update **Description** The first date the entitlement is in effect.
`Status`	**Type** picklist **Properties** Filter, Nillable

Field	Details
	Description Status of the entitlement, such as `Expired`.
`Type`	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** The type of entitlement, such as Web or phone support.

Usage

Use this object to query and manage entitlements.

See Also:

EntitlementContact
EntitlementHistory
SlaProcess

EntitlementContact

Represents a Contact eligible to receive customer support via an Entitlement.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`

Fields

Field	Details
`ContactId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Contact associated with the entitlement. Must be a valid ID.
`EntitlementId`	**Type** reference

Field	Details
	Properties Create, Filter **Description** Required. ID of the Entitlement associated with the entitlement contact. Must be a valid ID.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`Name`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Nillable **Description** Required. Name of the entitlement contact.

Usage

Use to query and manage entitlement contacts.

See Also:

Entitlement
EntitlementHistory

EntitlementFeed

Represents a single feed item in the feed displayed on the detail page for an entitlement.

An entitlement feed shows recent changes to an entitlement record for any fields that are tracked in feeds, as well as comments and posts about the record. It's a useful way to stay up-to-date with changes to entitlements. This object is available in API version 23.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the Entitlement object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field Name	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of the EntitlementFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable

Field Name	Details
	Description The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`InsertedById`	**Type** reference

Field Name	Details
	Properties Group, Nillable, Sort **Description** The ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, `InsertedById` is set to the ID of the logged-in user.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on the feed item, then `LastModifiedDate` becomes the `CreatedDate` for the FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of the `LinkPost`.
`NetworkScope`	**Type** picklist

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the entitlement record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as an EntitlementFeed object of `Type ContentPost`.
`Title`	**Type** string

Field Name	Details
	Properties Group, Nillable, Sort **Description** The title of the EntitlementFeed. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of EntitlementFeed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com.

Field Name	Details
	• `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case.
`Visibility`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for an entitlement record.

See Also:

Entitlement
EntitlementHistory
EntitlementContact
FeedItem

EntitlementHistory

Represents the changes to field values on an Entitlement.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`EntitlementId`	**Type** reference **Properties** Filter **Description** Required. ID of the Entitlement.
`Field`	**Type** picklist **Properties** Filter, Restricted picklist **Description** The name of the field that was changed.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.

Field	Details
`NewValue`	**Type** anyType **Properties** Nillable **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable **Description** The latest value of the field before it was changed.

Usage

This object respects field-level security on the parent object.

See Also:

Entitlement
EntitlementContact

EntitlementTemplate

Represents predefined terms of customer support for a product (Product2).

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`BusinessHoursId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the BusinessHours associated with the entitlement template. Must be a valid business hours ID.

Field	Details
`CasesPerEntitlement`	**Type** int **Properties** Create, Filter, Nillable, Update **Description** The total number of cases the entitlement template supports. This field is only available if `IsPerIncident` is `true`.
`IsPerIncident`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Update **Description** Indicates whether the entitlement template is limited to supporting a specific number of cases (`true`) or not (`false`).
`Name`	**Type** string **Properties** Create, Filter, idLookup, Update **Description** Required. Name of the entitlement template.
`SlaProcessId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the SlaProcess associated with the entitlement template. This field is available in API version 19.0 and later.
`Term`	**Type** int **Properties** Create, Filter, Nillable, Update **Description** Number of days that the entitlement template is valid.

Field	Details
Type	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** The type of entitlement template, such as Web or phone support.

Usage

Use this object to manage entitlement templates.

EntityHistory

Represents historical information about an object's changed field values. This object is only available to users with the "View All Data" permission.

This object is unavailable beginning with API version 8.0. Use the object-specific History objects instead: CaseHistory, ContractHistory, LeadHistory, OpportunityFieldHistory, OpportunityHistory ProcessInstanceHistory, QuantityForecastHistory, RevenueForecastHistory, or SolutionHistory.

Supported Calls

`describeSObjects()`, `getUpdated()`, `getDeleted()`, `query()`, `retrieve()`

Fields

Field	Details
FieldName	**Type** picklist **Properties** Filter, Restricted picklist **Description** ID of the standard or custom field.
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.

Field	Details
NewValue	**Type** anyType **Properties** Nillable **Description** New value of the modified field.
OldValue	**Type** anyType **Properties** Nillable **Description** Previous value of the modified field.
ParentId	**Type** reference **Properties** Filter **Description** ID of the object that contains the field.
ParentSobjectType	**Type** picklist **Properties** Filter, Restricted picklist **Description** The kind of object that contains the field.

Usage

In API version 7.0 and later, this object works with Case, Contract, and Solution objects:

- This object is always read-only in the online application.
- When a field is modified, this object records both the old and new field values. There are exceptions to this behavior for certain fields such as long text areas and multi-select picklists. These fields appear in this object to indicate that the field was changed, but the old and new values are not recorded.
- Two rows are added to this object when foreign key fields change. One row contains the foreign key object names that display in the online application. For example, "Jane Doe" is recorded as the name of a contact. The other row contains the actual foreign key ID that is only returned to and visible from the API.
- Up to a total of twenty fields (standard or custom) can be tracked for a given object.
- In the online application, you can specify which fields are tracked or not tracked at any time.

- As soon as tracking is turned on for a field, all changes to its value are recorded in the database.
- Turning off tracking for a field stops further changes from being recorded, but the history data is not deleted.
- Be advised that deleting a custom field also permanently deletes the history data for that custom field.

EntitySubscription

Represents a subscription for a user following a record or another user. A user can subscribe to a record or to another user. Changes to the record and updates from the users are displayed in the Chatter feed on the user's home page, which is a useful way to stay up-to-date with other users and with changes made to records in Salesforce. Feeds are available in API version 18.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`NetworkId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** ID of the community that this file originated from. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the record or user which the user is following.
`SubscriberId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the User who is following the record or user.

Usage

Consider this when following records and users:

- Users can only follow records that they can see.

- Users can see which records other users are following, unless they don't have access to the records.
- Administrators and users with the "Modify All Users" permission can configure a user to follow records that the user has read access to.
- Administrators and users with the "Modify All Users" permission can configure users to stop following records.
- Following topics is available in API version 29.0 and later. For this reason, a topic ID is now a supported value for the `ParentId` field.
- If you deactivate a user, any EntitySubscription where the user is associated with the ParentId or SubscriberId field, meaning all subscriptions both to and from the user, are soft deleted. If the user is reactivated, the subscriptions are restored. However, if you deactivate multiple users at once and these users follow each other, their subscriptions are hard deleted. In this case, the user-to-user EntitySubscription is deleted twice (double deleted). Such subscriptions can't be restored upon user reactivation.

When using `query()` with EntitySubscription,

- A query must include a LIMIT clause and the limit can't exceed 1000.
- A query using a `WHERE` clause can only filter by fields on the EntitySubscription object.
- If user sharing is enabled and the querying user is not an administrator, a SOQL query must be constrained either by the `ParentId` or `SubscriberId`. Otherwise, the query behavior at run time is undefined, meaning the result set can be incomplete or inconsistent from invocation to invocation. For an unconstrained query, the sharing check limits imposed on a non-adminstrative user are likely to be exceeded before the query completes, because access checks are run against both parent and subject, for each row of the result set. We recommend using the Chatter REST API to query EntitySubscription data instead of running a SOQL query.
- Users without the "View All Data" permission
  - ◊ Need read access on the object associated with the `ParentId` field to see which users are following records for the object.
  - ◊ Can use an `ORDER BY` clause in a query only to order by fields on the EntitySubscription object. For example, if the subscription relates to an Account record, the query can `ORDER BY ParentId`, but it can't `ORDER BY Account.Name`.
  - ◊ Don't always get all matching subscriptions when running a query. For these users, a query evaluates visibility criteria on a maximum of 500 records to reduce the prospect of long-running queries. If a user runs a query to see the CEO's subscriptions, it might scan a large number of records. The query only returns matches within the first 500 records scanned. It is possible that there are more subscriptions that are visible to the user, but they are not returned. To mitigate this, we recommend using a `WHERE` clause, if possible, to reduce the scope of the query.

Sample—SOQL

The following SOQL query returns subscriptions for all the accounts that a subscriber is following that have more than 10 employees:

```
SELECT Id
FROM EntitySubscription
WHERE SubscriberId = '005U0000000Rg2CIAS'
AND ParentId IN (
  SELECT Id FROM Account
  WHERE NumberOfEmployees > 10
)
LIMIT 200
```

See Also:

AccountFeed
CaseFeed
ContactFeed
OpportunityFeed
SolutionFeed

EnvironmentHubMember

Represents a member organization in the Environment Hub. This object is available in API version 29.0 and later.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`

Fields

Field Name	Details
CreatedById	**Type** reference **Properties** Defaulted on create, Filter **Description** System field that specifies the ID of the user who created this record.
CreatedDate	**Type** dateTime **Properties** Defaulted on create, Filter **Description** System field that specifies the date and time when this record was created.
Description	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** A brief description of this organization.
DisplayName	**Type** string **Properties** Create, Filter, Group, Sort **Description** The name that the user has specified for this member organization.

Field Name	Details
EnvironmentHubId	**Type** reference **Properties** Filter, Group, Sort **Description** The Organization ID of this member's Environment Hub organization.
Id	**Type** ID **Properties** Defaulted on create **Description** System field that uniquely identifies this record.
IsFedIdSsoMatchAllowed	**Type** boolean **Properties** Defaulted on create **Description** Indicates if single sign-on has been enabled based on matching Federation ID. The default value is `false`.
LastModifiedById	**Type** reference **Properties** Defaulted on create, Filter **Description** System field that specifies the ID of the user who last updated this record.
LastModifiedDate	**Type** reference **Properties** Defaulted on create, Filter **Description** System field that specifies the date and time when this record was last modified.
MemberEntity	**Type** string

Field Name	Details
	Properties Create, Filter, Group, Sort **Description** The Organization ID of the member organization this record is representing.
`OrgEdition`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The organization's edition, for example, Enterprise Edition or Unlimited Edition.
`OrgStatus`	**Type** picklist **Properties** Filter, Group, Sort, Update **Description** The licensing or creation status of this organization. Possible values include `Active`, `Demo`, `Deleted`, `Free`, `Inactive`, and `Trial`.
`Origin`	**Type** picklist **Properties** Filter, Group, Sort **Description** The method by which this organization was added to the Environment Hub. Possible values are `Auto Discovered` and `User Added`.
`ShouldAddRelatedOrgs`	**Type** boolean **Properties** Defaulted on create, Update **Description** If related organizations should be automatically connected to the hub, when this member organization is added. The default value is `true`.
`SsoStatus`	**Type** picklist

Field Name	Details
	Properties Filter, Group, Sort, Update **Description** If single sign-on has been enabled for this organization. Possible values are: • `Enabled`—Single sign-on is enabled. • `Disabled`—Single sign-on is disabled. • `Pending`—Single sign-on is in the process of being enabled. • `Failed`—Single sign-on enablement failed. Contact salesforce.com support for assistance. .
`SsoUsernameFormula`	**Type** string **Properties** Group, Nillable, Sort, Update **Description** The custom formula for matching users in the hub and member organizations.

Usage

Use this object to access and modify settings of member organizations in the Environment Hub.

Event

Represents an event in the calendar. In the user interface, event and task records are collectively referred to as activities.

Note:

- An EventRelation object can't be related to a child event, and child events don't include the invitee related list.
- `query()`, `delete()`, and `update()` aren't allowed with events related to more than one contact in API versions 25.0 and earlier.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`AccountId`	**Type** reference

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** Represents the ID of the related Account. The `AccountId` is determined as follows. If the value of `WhatId` is any of the following objects, then Salesforce uses that object's `AccountId`. • Account • Opportunity • Contract • Custom object that is a child of Account If the value of the `WhatId` field is any other object, and the value of the `WhoId` field is a Contact object, then Salesforce uses that contact's `AccountId`. (If your organization uses Shared Activities, Salesforce uses the `AccountId` of the primary contact.) Otherwise, Salesforce sets the value of the `AccountId` field to `null`. For information on IDs, see ID Field Type.
`ActivityDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Contains the event's due date if the `IsAllDayEvent` flag is set to `true`. This field is a date field with a timestamp that is always set to midnight in the Coordinated Universal Time (UTC) time zone. Don't attempt to alter the timestamp to account for time zone differences. Label is **Due Date Only**. This field is required in versions 12.0 and earlier if the `IsAllDayEvent` flag is set to `true`. The value for this field and `StartDateTime` must match, or one of them must be `null`.
`ActivityDateTime`	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** Contains the event's due date if the `IsAllDayEvent` flag is set to `false`. This field is a regular Date/Time field with a relevant time portion. The time portion is always transferred in the Coordinated Universal Time (UTC) time zone. Translate the time portion to or from a local time zone for the user or the application, as appropriate. Label is **Due Date Time**.

Field	Details
	This field is required in versions 12.0 and earlier if the `IsAllDayEvent` flag is set to `false`. The value for this field and `StartDateTime` must match, or one of them must be `null`.
`ClientGuid`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The client globally unique identifier identifies the external API client used to create the event. Label is **Client GUID**.
`CurrencyIsoCode`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.
`Description`	**Type** textarea **Properties** Create, Nillable, Update **Description** Contains a text description of the event. Limit: 32,000 characters.
`Division`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A logical segment of your organization's data. Available only if the organization has the Division permission enabled.
`DurationInMinutes`	**Type** int

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** Contains the event length, in minutes. Note that even though this field represents a temporal value, it is an integer type—not a Date/Time type. This field is required in versions 12.0 and earlier if `IsAllDayEvent` is false. In versions 13.0 and later, this field is optional, depending on the following: • If `IsAllDayEvent` is true, you can supply a value for either `DurationInMinutes` or `EndDateTime`. Supplying values in both fields is allowed if the values add up to equal time spans. If both fields are `null`, the duration defaults to one day. • If `IsAllDayEvent` is false, a value must be supplied for either `DurationInMinutes` or `EndDateTime`. Supplying values in both fields is allowed if the values add up to the same amount of time. If the multiday event feature is enabled, then API versions 13.0 and later support values greater than 1440 for the `DurationInMinutes` field. API versions 12.0 and earlier can't access event objects whose `DurationInMinutes` is greater than 1440. For more information, see **Multiday Events**.
`EndDateTime`	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** Available in versions 13.0 and later. This field is optional, depending on the following: • If `IsAllDayEvent` is true, you can supply a value for either `DurationInMinutes` or `EndDateTime`. Supplying values in both fields is allowed if the values add up to equal time spans. If both fields are `null`, the duration defaults to one day. • If `IsAllDayEvent` is false, a value must be supplied for either `DurationInMinutes` or `EndDateTime`. Supplying values in both fields is allowed if the values add up to the same amount of time.
`GroupEventType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Read-only. Available in API versions 19.0 and later. The possible values are: • `0` (Non–group event)—An event with no invitees.

Field	Details
	• `1` (Group event)—An event with invitees. • `2` (Proposed event)—An event created when a user requests a meeting with a contact, lead, or person account using the Salesforce user interface. When the user confirms the meeting, the proposed event becomes a group event. You can't create, edit, or delete proposed events in the API. For more information, see About Cloud Scheduler in the Salesforce help.
`IsAllDayEvent`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the `ActivityDate` field (`true`) or the `ActivityDateTime` field (`false`) is used to define the date or time of the event. Label is **All-Day Event**. See also `DurationInMinutes` and `EndDateTime`.
`IsArchived`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the event has been archived.
`IsChild`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the event is a child of another event (`true`) or not (`false`). For a child event, you can update `IsReminderSet` and `ReminderDateTime` only. You can query and delete a child event. If the objects related to the child event are different from those related to the parent event (this difference is possible if you use API version 25.0 or earlier) and one of the objects related to the child event is deleted, the objects related to the parent event are updated to ensure data integrity.
`IsClientManaged`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates whether the event is managed by an external client. If the value of this field is false, the event isn't owned or managed by an external client, and Salesforce can be used to update it. If the value is true, Salesforce can be used to change only noncritical fields on the event. Label is **Is Client Managed**.
`IsGroupEvent`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the event is a group event—that is, whether it has invitees (`true`) or not (`false`).
`IsPrivate`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether users other than the creator of the event can (`false`) or can't (`true`) see the event details when viewing the event user's calendar. However, users with the View All Data or Modify All Data permission can see private events in reports and searches, or when viewing other users' calendars. Private events can't be associated with opportunities, accounts, cases, campaigns, contracts, leads, or contacts. Label is **Private**.
`IsRecurrence`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** Indicates whether the event is scheduled to repeat itself (`true`) or only occurs once (`false`). This is a read-only field when updating records, but not when creating them. If this field value is `true`, then `RecurrenceEndDateOnly`, `RecurrenceStartDateTime`, `RecurrenceType`, and any recurrence fields associated with the given recurrence type must be populated. Label is **Create recurring series of events**.
`IsReminderSet`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field	Details
	Description Indicates whether the activity is a reminder (`true`) or not (`false`).
`IsVisibleInSelfService`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether an event associated with an object can be viewed in the Customer Portal (`true`) or not (`false`). If your organization has enabled Communities, events marked `IsVisibleInSelfService` are visible to any external user in the community, as long as the user has access to the record the event was created on. This field is available when Customer Portal or partner portal are enabled OR Communities is enabled and you have Customer Portal or partner portal licenses.
`Location`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Contains the location of the event.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Contains the ID of the user who owns the event. Label is **Assigned to ID**.
`RecurrenceActivityId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Read-only. Not required on create. Contains the ID of the main record of the recurring event. Subsequent occurrences have the same value in this field.

Field	Details
`RecurrenceDayOfMonth`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Indicates the day of the month on which the event repeats.
`RecurrenceDayOfWeekMask`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Indicates the day or days of the week on which the event repeats. This field contains a bitmask. The values are as follows: • Sunday = `1` • Monday = `2` • Tuesday = `4` • Wednesday = `8` • Thursday = `16` • Friday = `32` • Saturday = `64` Multiple days are represented as the sum of their numerical values. For example, Tuesday and Thursday = 4 + 16 = 20.
`RecurrenceEndDateOnly`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Indicates the last date on which the event repeats. For multiday recurring events, this is the day on which the last occurrence starts.
`RecurrenceInstance`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates the frequency of the event's recurrence. For example, `2nd` or `3rd`.

Field	Details
RecurrenceInterval	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Indicates the interval between recurring events.
RecurrenceMonthOfYear	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates the month in which the event repeats.
RecurrenceStartDateTime	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** Indicates the date and time when the recurring event begins. The value must precede the `RecurrenceEndDateOnly`.
RecurrenceTimeZoneSidKey	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates the time zone associated with a recurring event. For example, “UTC-8:00” for Pacific Standard Time.
RecurrenceType	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates how often the event repeats. For example, daily, weekly, or every nth month (where “nth” is defined in `RecurrenceInstance`).

Field	Details
ReminderDateTime	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** Represents the time when the reminder is scheduled to fire, if `IsReminderSet` is set to `true`. If `IsReminderSet` is set to `false`, then the user may have deselected the reminder checkbox in the Salesforce user interface, or the reminder has already fired at the time indicated by the value.
ShowAs	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates how this event appears when another user views the calendar: Busy, Out of Office, or Free Time. Label is **Show Time As**.
StartDateTime	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** Indicates the start date and time of the event. Available in versions 13.0 and later. If the Event `IsAllDayEvent` flag is set to true (indicating that it is an all-day Event), then the event start date information is contained in the `StartDateTime` field. This field is a date field with a timestamp that is always set to midnight in the Coordinated Universal Time (UTC) time zone. The timestamp is not relevant, and you should not attempt to alter it to account for any time zone differences. If the Event `IsAllDayEvent` flag is set to false (indicating that it is not an all-day event), then the event start date information is contained in the `StartDateTime` field. The time portion is always transferred in the Coordinated Universal Time (UTC) time zone. You need to translate the time portion to or from a local time zone for the user or the application, as appropriate. If this field has a value, then `ActivityDate` and `ActivityDateTime` must either be `null` or match the value of this field.
Subject	**Type** combobox **Properties** Create, Filter, Nillable, Sort, Update

Field	Details
	Description The subject line of the event, such as Call, Email, or Meeting. Limit: 255 characters.
`Type`	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** Indicates the event type, such as Call, Email, or Meeting.
`WhatCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Available if your organization has enabled Shared Activities. Represents the count of related EventRelations pertaining to the `WhatId`. The count of the `WhatId` must be `1` or less.
`WhatId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Contains the ID of a related Account, Opportunity, Campaign, Case, or custom object. Label is **Opportunity/Account ID**.
`WhoCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Available to organizations that have Shared Activities enabled. Represents the count of related EventRelations pertaining to the `WhoId`.
`WhoId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description Contains the ID of a related lead or contact. If the `WhoId` refers to a lead, then the `WhatId` field must be empty. Label is **Contact/Lead ID**. If Shared Activities is enabled, this field contains the ID of a related lead or primary contact.

Usage

Use Event to manage calendar appointments.

Querying and Filtering Events

Queries on events will be denied before they time out if they involve amounts of data that are deemed too large. In such cases, the exception code `OPERATION_TOO_LARGE` is returned. If you receive `OPERATION_TOO_LARGE`, refactor your query to return or scan a smaller amount of data.

When querying for events with a specific due date, you must filter on both the `ActivityDateTime` and `ActivityDate` fields. For example to find all events with a due date of February 14, 2003, you need two filters:

- One filter with the `ActivityDate` field equal to the Coordinated Universal Time (UTC) time zone on February 14, 2003.
- One filter with the `ActivityDateTime` field greater than or equal to midnight on February 14, 2003 in the user's local time zone AND less than or equal to midnight on February 15, 2003 in the user's local time zone.

Alternatively, in version 13.0 and later, you can find events with a specific due date by filtering on `StartDateTime`. For example, to find all events with a due date of February 14, 2003, filter with the `StartDateTime` greater than or equal to midnight on February 14, 2003 in the user's local time zone AND less than or equal to midnight on February 15, 2003 in the user's local time zone.

The `EventId` field of an EventRelation object always points to the master record. An invitee on a group event can query the EventRelation object to view the master record.

Multiday Events

- Multiday events are available in version 13.0 and later. Also, in earlier versions SOQL queries do not return multiday events.
- Multiday events are enabled through the user interface from Setup by clicking **Customize** > **Activities** > **Activity Settings**.
- If the multiday event feature is enabled, then API versions 13.0 and later support values greater than 1440 for the `DurationInMinutes` field. API versions 12.0 and earlier can't access event objects whose `DurationInMinutes` is greater than 1440.
- Multiday events can't exceed 14 days.

Recurring Events

- Recurring events are available in version 7.0 and later.
- After an event is created, it can't be changed from recurring to nonrecurring or vice versa.
- When you delete a recurring event series through the API, all past and future events in the series are removed. However, when you delete a recurring event series through the user interface, only future occurrences are removed.
- When creating a recurring event series, the duration of the event must be 24 hours or less (either the `DurationInMinutes` or the difference between `RecurrenceStartDateTime` and `EndDateTime` must be greater than 24 hours). Once the recurring event series is created, you can extend the length of individual occurrences beyond 24 hours if Multiday events are enabled; see **Multiday Events**.
- If `IsRecurrence` is `true`, then `RecurrenceStartDateTime`, `RecurrenceEndDateOnly`, `RecurrenceType`, and any properties associated with the given recurrence type (see the following table) must be populated.
- When updating a recurring event series, it's not possible to update the EventRelation for the event series object and the EventRelation for the series object occurrences at the same time.

The following table describes the usage of recurrence fields. Each recurrence type must have all of its properties set. All unused properties must be set to null.

RecurrenceType Value	Properties	Example Pattern
RecursDaily	RecurrenceInterval	Every second day
RecursEveryWeekday	RecurrenceDayOfWeekMask	Every weekday - can't be Saturday or Sunday
RecursMonthly	RecurrenceDayOfMonth RecurrenceInterval	Every second month, on the third day of the month
RecursMonthlyNth	RecurrenceInterval RecurrenceInstance RecurrenceDayOfWeekMask	Every second month, on the last Friday of the month
RecursWeekly	RecurrenceInterval RecurrenceDayOfWeekMask	Every three weeks on Wednesday and Friday
RecursYearly	RecurrenceDayOfMonth RecurrenceMonthOfYear	Every March on the twenty-sixth day of the month
RecursYearlyNth	RecurrenceDayOfWeekMask RecurrenceInstanceRecurrenceMonthOfYear	The first Saturday in every October

See Also:

Archived Activities
Standard and Custom Object Basics

EventFeed

Represents a single feed item in the feed on an Event.

This object is available in API version 20.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of EventFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort

Field	Details
	Description The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.

Field	Details
LikeCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
NetworkScope	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
ParentId	**Type** reference **Properties** Filter, Group, Sort

Field	Details
	Description ID of the event record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file.

Field	Details
	• `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`:

Field	Details
	• For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for an event record. You can only delete a feed if you created it, or if you have the “Modify All Data” permission.

EventRelation

Represents people (a user, a lead, or contacts) or a resource (such as a conference room) invited to an event. This object lets you add or remove invitees from an event and use the API to manage invitees’ responses to invitations.

EventRelation allows a variable number of relationships, as follows:

- If you’ve enabled Shared Activities for your organization, an event can be related to up to 50 contacts or one lead. If you haven’t enabled Shared Activities, an event can be related to only one contact or lead.
- An event can also be related to one account, asset, campaign, case, contract, opportunity, product, solution, or custom object.

Note:

- With API versions 26.0 and later, the EventRelation object replaces the EventAttendee object, and the EventAttendee object is no longer visible. You can still query the EventAttendee object using packages that support API versions 25.0 and earlier, or by using Apex.
- An EventRelation object can’t be created for a child event.
- EventRelation includes deactivated users.
- In API versions 25.0 and earlier, you can’t use `query()`, `delete()`, or `update()` with events related to more than one contact.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `queryAll()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`AccountId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Contains the Account ID of the relation. For information on IDs, see ID Field Type. `AccountId` is visible when Shared Activities is enabled.
`EventId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Contains the ID of the event. This value can't be changed after it's been specified.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is `Deleted`.
`IsInvitee`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the relation is an invitee. • `IsInvitee` is visible while Shared Activities is being enabled, after it has been enabled, and while it is being disabled. • `IsInvitee` defaults to `true` while Shared Activities is being enabled, after it has been enabled, and while it is being disabled if `IsInvitee`, `IsParent`, and `IsWhat` are not set. This configuration ensures compatibility when Shared Activities isn't enabled and EventRelation represents event invitees only. • `IsInvitee` defaults to `false` when Shared Activities is enabled if `IsParent` is set to `true`.
`IsParent`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field	Details
	Description Indicates whether the relation is a contact or lead, or an account, opportunity, or other object. `IsParent` is visible when Shared Activities is enabled.
`IsWhat`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** Indicates whether the relation is an account, an opportunity, or another object. Value is `false` if `RelationId` is a contact or a lead, and `true` otherwise. `IsWhat` is visible when Shared Activities is enabled.
`RelationId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Contains the ID of the person (User, Contact, or Lead) or the resource invited to an event. When Shared Activities is enabled, `RelationId` can also contain the ID of the contacts or lead related to an event. This value can't be changed after it's been specified.
`RespondedDate`	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** Indicates the most recent date and time when the invitee responded to an invitation to an event.
`Response`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Contains optional text that the invitee can enter when responding to an invitation to an event.

Field	Details
Status	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates the invitee status with one of the following values: • `New`: Invitee has received the invitation but hasn't yet responded. This value is the default. • `Declined`: Invitee has declined the invitation. • `Accepted`: Invitee has accepted the invitation. **Note:** `Uninvited` and `Maybe` aren't currently supported.

Usage

Send email notifications

To send email notifications for a given event, query EventRelation for the event, iterate through the list, examine the status, and send email notifications to every person who accepted the invitation.

Determine what events a given invitee is attending

To determine all the events that a particular person is attending during a given time period (for example, next week), you can have a client application query the Event object for a given date range, iterate through the results, and, for each event, query the EventRelation object to determine whether the particular person (`RelationId`) has accepted an invitation to that event.

Create an invitee if Shared Activities is enabled (or during the process of enabling it or rolling back)

If the invitee is already a contact or lead, update `IsInvitee` to `true`.

If the invitee is not already a contact or lead, create an EventRelation object for the invitee with `IsInvitee` set to `true`.

Create an invitee if Shared Activities is not enabled

Create an EventRelation object for the invitee.

Query relations to a contact or a lead

```
List<EventRelation> whoRelations = [SELECT Id, Relation.Name FROM
    EventRelation WHERE EventId = '00UD0000005zijD' AND isParent = true AND isWhat =
false];
```

Query invitee relations

```
List<EventRelation> inviteeRelations = [SELECT Id, Relation.Name FROM
    EventRelation WHERE EventId = '00UD0000005zijD' AND isInvitee = true];
```

Update an invitee relation to a contact or lead invitee relation

```
EventRelation er = [SELECT Id FROM EventRelation WHERE EventId =
    '00UD0000005zijD' AND isInvitee = true and isParent = false LIMIT 1];
er.isParent = true;
update er;
```

Update a contact or lead relation to a contact or lead invitee relation

```
EventRelation er = [SELECT Id FROM EventRelation WHERE EventId =
    '00UD0000005zijD' AND isParent = true and isInvitee = false LIMIT 1];
er.isInvitee = true;
update er;
```

Insert a contact or lead relation

```
EventRelation er = new EventRelation(EventId = '00UD0000005zijH',
    RelationId = '003D000000Q8aeV', isParent = true, isInvitee = false);
insert er;
```

Insert an invitee relation

If `isParent`, `isWhat` and `IsInvitee` are not set, and `RelationId` is a contact, lead, user, or calendar, `IsInvitee` defaults to `true`. This means if an EventRelation isn't specifically inserted as a relation to a contact or lead, it's treated as an Invitee relation by default.

```
EventRelation er = new EventRelation(EventId = '00UD0000005zijH',
    RelationId = '003D000000Q8adV');
insert er;
```

See Also:

Event
EventWhoRelation
Standard and Custom Object Basics

EventTag

Associates a word or short phrase with an Event.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.
`Type`	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

EventTag stores the relationship between its parent TagDefinition and the Event being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

EventWhoRelation

Represents the relationship between an event and a lead or contacts. This derived object is a filtered version of the EventRelation on page 639 object; that is, IsParent is `true` and IsWhat is `false`. It doesn't represent relationships to invitees or to accounts, opportunities, or other objects.

EventWhoRelation allows a variable number of relationships: one lead or up to 50 contacts. Available only if you've enabled Shared Activities for your organization. This object is available in API versions 29.0 and later.

Note: EventWhoRelation objects aren't created for child events.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`EventId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the event.
`RelationId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the contacts or lead related to the event.
`Type`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Indicates whether the person related to the event is a contact or lead.

Usage

Apex example that queries relations to a contact or lead

```
List<EventWhoRelation> whoRelations = [SELECT Id, Relation.Name FROM
EventWhoRelation WHERE EventId = '00UD0000005zijD'];
```

See Also:

Event

EventRelation

ExternalDataSource

Represents an external data source, such as SharePoint®.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ApiKey`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The organization-specific client key required by some external data systems.
`DeveloperName`	**Type** string **Properties** Filter, Group, Sort **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.

Field Name	Details
Endpoint	**Type** textarea **Properties** Nillable **Description** The root endpoint URL of the external data source.
Language	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The language of the `MasterLabel`.
MasterLabel	**Type** string **Properties** Filter, Group, Sort **Description** Master label for the external data source. This display value is the internal label that does not get translated.
NamespacePrefix	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation.
PrincipalType	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort

Field Name	Details
	Description Used to track users accessing the external server. Anonymous implies a user identity is not specified for external server access. Named Principal uses one user identity for all users to access the external server.
`Repository`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** An optional name of the repository in the data source. Not applicable to all data source types.
`Type`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** File type of the source document.

Usage

External data sources connect with data sources outside Salesforce, such as SharePoint®. Selected tables in an external data source map to external objects, which function much like custom objects in Salesforce.

ExternalDataUserAuth

Stores authentication settings required for a specific user and external data source, such as SharePoint®.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`ExternalDataSourceId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The name specified in the external data source definition.

Field Name	Details
UserId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Nillable, Sort, Update **Description** The user's first and last name in Salesforce.
Protocol	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The authentication protocol required by the external data source. For SharePoint, specify Basic Authentication.
Username	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The username required by the external data source.
Password	**Type** encrypted string **Properties** Create, Nillable, Update **Description** The password required by the external data source.

Usage

External authentication settings let you store user names and passwords for external data sources such as SharePoint, along with the required protocol. Unlike external data source definitions, which store only administrator credentials, these authentication settings store credentials for any user who will access the external data from Salesforce.

FeedComment

Represents a comment added to a feed by a user. Represents a comment added to a feed by a user.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `search()`

Special Access Rules

Note the following when working with feed comments:

- You must have read access to the feed's parent type to see a FeedComment record.
- You can add a comment if you have access to the feed.
- If the comment is related to a user record, the user can delete the comment. For example, if John Smith makes a comment about Sasha Jones, Sasha can delete the comment.
- If the logged-in user has the “Insert System Field Values for Chatter Feeds” user permission, the `create` field property is available on `CreatedBy` and `CreatedDate` system fields for this object. This allows the logged-in user to set these fields to the original post author and creation date upon migration instead of accepting the system field value when migrated, which would be the logged-in user and the date the migration was performed, respectively. The fields can't be updated after migration.

You can delete all feed items you created. To delete feed items you didn’t create, you must have one of these permissions:

- “Modify All Data”
- “Modify All” on the object associated with the feed as well as delete permission on the parent feed
- “Moderate Chatter”

Note: Users with the “Moderate Chatter” permission can delete only the feed items and comments they see.

Fields

Field	Details
`CommentBody`	**Type** textarea **Properties** Create, Filter, Sort **Description** The text in the comment.
`CommentType`	**Type** reference **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of comment: • `ContentComment`—an uploaded file on a comment • `TextComment`—a direct text entry on a comment Prior to API version 24.0, a text entry was required on a comment. As of version 24.0, a text entry is optional if the `CommentType` is `ContentComment`.

Field	Details
`FeedItemId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the feed containing the comment.
`InsertedById`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`ParentId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of a record associated with the feed comment. For example, if you are commenting on a change to a field on Account, `ParentId` is set to the account ID.
`RelatedRecordId`	**Type** reference **Properties** Create, Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentComment`. This field is null for all comments except `ContentComment`. For example, set this field to an existing ContentVersion and post it to a comment as a FeedComment object of `CommentTypeContentComment`.

Usage

- As of API version 23.0 and beyond, if you have "View All Data" permission you can query FeedComment records directly, without an ID filter. However, if you don't have "View All Data" permission, you can't query FeedComment records directly, with or without an ID filter.

For example, the following query returns general information about a feed:

```
SELECT ID, CreatedDate, CreatedById, CreatedBy.FirstName,
           CreatedBy.LastName, ParentId, Parent.Name, Body
FROM FeedItem
WHERE CreatedDate > LAST_MONTH
ORDER BY CreatedDate DESC, Id DESC
```

- You can search for text in comments using SOSL. For example, the following Java class uses `search()` to find the string “foo” in any field of a record:

```
public void searchSample() {
  try {
    SearchResult sr = connection.search("find {foo} in all fields " +
        "returning feedcomment(Id, FeedItemId, CommentBody)");
    // Put the results into an array of SearchRecords
    SearchRecord[] records = sr.getSearchRecords();
    // Check the length of the returned array of records to see
    // if the search found anything
    if (records != null && records.length > 0) {
      System.out.println("Found " + records.length + " comments: ");
      // Display each comment
      for (SearchRecord record : records) {
        FeedComment comment = (FeedComment) record.getRecord();
        System.out.println(comment.getId() + ": " +
            comment.getCommentBody());
      }
    } else {
      System.out.println("No records were found for the search.");
    }
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

- If you use an Apex trigger to modify the `Body` of a FeedComment object, all mentions hyperlinks are converted to plain text and the mentioned users don't get email notifications.

Note: This object is hard deleted. It isn’t sent to the Recycle Bin.

See Also:

NewsFeed
UserProfileFeed

FeedItem

FeedItem represents an entry in the feed, such as changes in a record feed, including text posts, link posts, and content posts.

This object is available in API version 21.0 and later. This object replaces FeedPost.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Only users with the "Modify All Data" permission can delete a FeedItem of `Type` `TrackedChange`.

If the logged-in user has the "Insert System Field Values for Chatter Feeds" user permission, the `create` field property is available on `CreatedBy` and `CreatedDate` system fields for this object. This allows the logged-in user to set these fields to the original post author and creation date upon migration instead of accepting the system field value when migrated, which would be the logged-in user and the date the migration was performed, respectively. The fields can't be updated after migration.

Fields

Field Name	Details
`Body`	**Type** textarea **Properties** Create, Filter, Nillable, Sort **Description** The content of the FeedItem. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.

Field Name	Details
ContentData	**Type** base64 **Properties** Create, Nillable **Description** The file uploaded to the feed. Required if Type is ContentPost. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets Type to ContentPost.
ContentDescription	**Type** textarea **Properties** Create, Nillable, Sort **Description** The description of the file specified in ContentData.
ContentFileName	**Type** string **Properties** Create, Group, Nillable, Sort **Description** The file uploaded to the feed. Required if Type is ContentPost. The name of the file uploaded to the feed. Setting ContentFileName automatically sets Type to ContentPost.
ContentSize	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
ContentType	**Type** string **Properties** Group, Nillable, Sort

Field Name	Details
	Description Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.

Field Name	Details
LinkUrl	**Type** url **Properties** Create, Filter, Nillable, Sort **Description** The URL of a LinkPost.
NetworkScope	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. NetworkScope can have the following values: • NetworkId—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • AllNetworks—The feed item is available in all communities. Note the following exceptions for NetworkScope: • Only feed items with a Group or User parent can set a NetworkId or a null value for NetworkScope. • For feed items with a record parent, users can set NetworkScope only to AllNetworks. • You can't filter a FeedItem on the NetworkScope field.
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the object type to which the FeedItem object is related. For example, set this field to a UserId to post to someone's profile feed, or an AccountId to post to a specific account.
RelatedRecordId	**Type** reference

Field Name	Details
	Properties Create, Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as a FeedItem object of `Type` `ContentPost`.
`Title`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort **Description** The title of the FeedItem. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Create,Filter, Group **Description** The type of FeedItem. With the exception of `ContentPost`, `LinkPost`, and `TextPost`, FeedItem types should not be created directly from the API. • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file.

Field Name	Details
	• CreatedRecordEvent—generated when a user creates a record from the publisher. • DashboardComponentAlert—generated when a dashboard metric or gauge exceeds a user-defined threshold. • DashboardComponentSnapshot—created when a user posts a dashboard snapshot on a feed. • LinkPost—a post with an attached URL. • PollPost—a poll posted on a feed. • ProfileSkillPost—generated when a skill is added to a user's Chatter profile. • QuestionPost—generated when a user posts a question. • ReplyPost—generated when Chatter Answers posts a reply. • RypplePost—generated when a user creates a Thanks badge in Work.com. • TextPost—a direct text entry on a feed. • TrackedChange—a change or group of changes to a tracked field. • UserStatus—automatically generated when a user adds a post. Deprecated. The following values appear in the Type picklist for all feed objects but apply only to CaseFeed: • AttachArticleEvent—generated event when a user attaches an article to a case. • CallLogPost—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • CaseCommentPost—generated event when a user adds a case comment for a case object. • ChangeStatusPost—generated event when a user changes the status of a case. • ChatTranscriptPost—generated event when Live Agent transcript is saved to a case. • EmailMessageEvent—generated event when an email related to a case object is sent or received. • FacebookPost—generated when a Facebook post is created from a case. Deprecated. • MilestoneEvent—generated when a case milestone is completed or reaches violation status. • SocialPost—generated when a social post is created from a case. **Note:** If you set Type to ContentPost, you must specify ContentData and ContentFileName.
Visibility	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort

Field Name	Details
	Description Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

- If you are using API version 23.0 or later and have "View All Data" permission, you can directly query for a FeedItem. The following example returns the 20 most recent feed items.

```
SELECT ID, CreatedDate, CreatedById, CreatedBy.FirstName, CreatedBy.LastName, ParentId,
 Parent.Name, Body,
 (SELECT ID, FieldName, OldValue, NewValue, FROM FeedTrackedChanges ORDER BY ID DESC)
FROM FeedItem
WHERE CreatedDate > LAST_MONTH
ORDER BY CreatedDate DESC, Id DESC LIMIT 20;
```

- If you are using an earlier API version than version 23.0, you have to query FeedItem objects through a feed (such as AccountFeed or OpportunityFeed). The following example returns all feed items for a given account, ordered by date descending:

```
SELECT Id, Type, FeedItem.Body
FROM AccountFeed
WHERE ParentId = AccountId ORDER BY CreatedDate DESC
```

Note: Prior to API version 23.0 you needed to provide the `ParentId`.

- A FeedItem of type `UserStatus` is automatically created when a user adds a post to update the current status. You can't explicitly create a FeedItem of type `UserStatus`.
- The FeedItem object doesn't support aggregate functions in queries.
- If the logged-in user has the "Insert System Field Values for Chatter Feeds" user permission, the `create` field property is available on `CreatedBy` and `CreatedDate` system fields for this object. This allows the logged-in user to set these fields to the original post author and creation date upon migration instead of accepting the system field value when migrated, which would be the logged-in user and the date the migration was performed, respectively. The fields can't be updated after migration.
- The size limit for an attachment on a profile or news feed is 25 MB.
- The size limit for an attachment on a record feed is 5 MB.

- You can’t use the content fields to update or delete the content.
- You can’t filter or update the content fields.
- Deleting a FeedItem via the API also deletes the associated content. Likewise, undeleting a FeedItem restores associated content.

  Note: This object is hard deleted. It isn’t sent to the Recycle Bin.

- After uploading to a feed, it is possible for an attachment or document to be deleted, marked private, or hidden by sharing rules. In this case, all content fields in a FeedItem object appear as `null` in a SOQL query.
- You can’t explicitly create or delete a FeedTrackedChange record.
- If you insert a FeedItem or FeedComment of `Type ContentPost` on a User or Group to create a new file, the `NetworkScope` field value of the FeedItem is passed to the file.
- If you use an Apex trigger to modify the `Body` of a FeedItem object, all mentions hyperlinks are converted to plain text and the mentioned users don't get email notifications.

FeedLike

Indicates that a user has liked a feed item. FeedLike records represent likes on posts and not likes on comments. Likes on comments can’t be queried via the API. A FeedLike is a child object of an associated FeedItem, FeedTrackedChange, or object feed, such as AccountFeed. This object is available in API version 21.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`

Special Access Rules

If the logged-in user has the “Insert System Field Values for Chatter Feeds” user permission, the `create` field property is available on `CreatedBy` and `CreatedDate` system fields for this object. This allows the logged-in user to set these fields to the original post author and creation date upon migration instead of accepting the system field value when migrated, which would be the logged-in user and the date the migration was performed, respectively. The fields can't be updated after migration.

Fields

Field Name	Details
`FeedItemId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the feed item that the user liked.
`FeedEntityId`	**Type** reference **Properties** Create, Filter, Group, Sort

Field Name	Details
	Description The Id of a feed item or feed comment the user liked. If the user liked a comment, FeedEntityId is set to the ID of the comment. If the user liked a feed item, FeedEntityId is set to the ID of the feed item. FeedEntityId is an optional field. The default value is the ID of the feed item.
`InsertedById`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.

Usage

You can't query FeedLike records directly. They can only be queried via the entity feed, such as AccountFeed.

FeedLike records represent likes on posts and not likes on comments. Likes on comments can’t be queried via the API.

FeedPollChoice

Shows the choices for a poll posted in the feed. This object is available in API version 29.0 and later.

Supported Calls

`query()`, `retrieve()`

Fields

Field Name	Details
`ChoiceBody`	**Type** string **Properties** Group **Description** A choice in the poll.
`FeedItemId`	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Sort **Description** ID of the feed item for the poll.
`Position`	**Type** int **Properties** Group, Sort **Description** Shows the position of the poll choice.

Usage

Use this object to query all of the choices associated with a particular poll. To view how people voted on the poll, see the FeedPollVote object.

FeedPollVote

Shows how users voted on a poll posted in the feed. This object is available in API version 29.0 and later.

Supported Calls

`query()`, `retrieve()`

Fields

Field Name	Details
`ChoiceId`	**Type** reference **Properties** Filter, Group **Description** Indicates which choice a user selected on a poll posted in a feed.
`FeedItemId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the feed item for the poll.

Usage

Use this object to query how users voted on a particular poll.

FeedPost

FeedPost represents the following types of changes in a record feed, such as AccountFeed: text posts, link posts, and content posts. This object is available in API version 18.0 through 21.0. FeedPost is no longer available in later versions. Starting with API version 21.0, use FeedItem to represent text posts, link posts, and content posts in feeds.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `search()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Only users with the "Modify All Data" permission can delete a FeedItem of `Type` `TrackedChange`.

If the logged-in user has the "Insert System Field Values for Chatter Feeds" user permission, the `create` field property is available on `CreatedBy` and `CreatedDate` system fields for this object. This allows the logged-in user to set these fields to the original post author and creation date upon migration instead of accepting the system field value when migrated, which would be the logged-in user and the date the migration was performed, respectively. The fields can't be updated after migration.

Fields

Field	Details
`Body`	**Type** string **Properties** Create, Filter, Nillable, Sort **Description** The content of the FeedPost. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`ContentData`	**Type** base64 **Properties** Create, Nillable

Field	Details
	Description The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** string **Properties** Create, Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Create, Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`FeedItemId`	**Type** reference

Field	Details
	Properties Filter, Group, Sort **Description** ID of the feed containing the FeedPost.
`InsertedById`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LinkUrl`	**Type** url **Properties** Create, Filter, Nillable, Sort **Description** The URL of a `LinkPost`.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the object type to which the FeedPost is related. For example, set this field to a `UserId` to post to someone's profile feed, or an `AccountId` to post to a specific account.
`Title`	**Type** string

Field	Details
	Properties Create, Filter, Group, Nillable, Sort **Description** The title of the FeedPost. When the `Type` is `LinkPost`, the `Body` is the URL and the `Title` is the label for the link.
`Type`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Restricted picklist, Sort **Description** The type of FeedPost: • `UserStatus`—automatically generated when a user updates their status • `TrackedChange`—ignore • `TextPost`—a direct text entry on a feed • `LinkPost`—a URL posting on a feed • `ContentPost`—an uploaded file on a feed **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.

Usage

- You can't directly query for a FeedPost. FeedPosts are always associated with a feed item, so you can query for them through the feeds. The following example returns all feed items for a given account, ordered by date descending:

```
SELECT Id, Type, FeedPost.Body
FROM AccountFeed
WHERE ParentId = AccountId ORDER BY CreatedDate DESC
```

- A FeedPost of type `UserStatus` is automatically created when a user adds a post to update the current status. You can't explicitly create a FeedPost of type `UserStatus`.
- The size limit for an attachment on a profile, news, or record feed is 2 GB.
- You can't use the content fields to update or delete the content.
- You can't filter or update the content fields.
- Deleting a FeedPost via the API also deletes the associated content and FeedPost objects. Likewise, undeleting a FeedPost restores associated content and FeedPost objects.

  Note: This object is hard deleted. It isn't sent to the Recycle Bin.

- After uploading to a feed, it is possible for an attachment or document to be deleted, marked private, or hidden by sharing rules. In this case, all content fields in FeedPost appear as `null` in a SOQL query.
- You can't explicitly create or delete a FeedTrackedChange record.

FeedTrackedChange

Represents an individual field change or set of field changes. A FeedTrackedChange is a child object of a record feed, such as AccountFeed. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`CurrencyIsoCode`	**Type** picklist **Properties** Group, Restricted picklist, Sort **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO currency code for the field, if `FieldName` is a currency field.
`FeedItemId`	**Type** reference **Properties** Group, Sort, Filter **Description** ID of the parent feed that tracks the field change.
`FieldName`	**Type** string **Properties** Group, Sort **Description** The name of the field that was changed. **Note:** This field also tracks other events that are not related to an individual field for a parent feed. These events occur as the parent record advances through its pipeline. For example, a value of `leadConverted` indicates that a lead has been converted to an opportunity. For a full list of values, see Tracking of Special Events.
`NewValue`	**Type** anyType

Field	Details
	Properties Nillable, Sort **Description** The new value of the field that was changed.
`OldCurrencyIsoCode`	**Type** string **Properties** Group, Nillable, Sort **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO currency code for the `OldValue` field, if `FieldName` is a currency field.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The last value of the field before it was changed.

Usage

A user can subscribe to a record or to another user. Changes to the record and updates from the users are displayed in the Chatter feed on the user's home page, which is a useful way to stay up-to-date with other users and with changes made to records in Salesforce. Feeds are available in API version 18.0 and later.

If you move a custom field to the Recycle Bin, all FeedTrackedChange records that track historical changes to the custom field are automatically deleted and are not restored if the custom-field is undeleted.

The following sections outline the difference between standard feeds and custom feeds.

Standard Feeds

A standard feed is a record feed, such as AccountFeed. FeedTrackedChange records for standard feeds can only be queried via the parent feed object.

Note the following when working with standard feed items:

- Feed items for standard feeds are read only in the API.
- A FeedTrackedChange record is visible when you have read access on the record feed, and when the field in the FeedTrackedChange is visible in the field-level security settings.

Custom Feeds

If you want more control over the information provided in a record feed, such as AccountFeed, you can create a custom feed. A custom feed can replace or augment an existing record feed. For example, you might want to:

- Disable the standard account record feed and use an Apex trigger to generate FeedTrackedChange records for the events that you want to track in the feed instead.

- Augment the standard contact record feed by writing an API client that inserts feed items for events that are not tracked in the standard feed.

Tracking of Special Events

The `FieldName` field also tracks other events that are not related to an individual field for a parent feed. These events occur as the parent record advances through its pipeline. For example, a value of `leadConverted` indicates that a lead has been converted to an opportunity.

Valid values for the `FieldName` field for multiple objects:

- `created`
- `ownerAccepted`
- `ownerAssignment`

Additional valid values for the `FieldName` field for individual objects:

Account

- `accountCreatedFromLead`
- `accountMerged`
- `accountUpdatedByLead`
- `personAccountUpdatedByLead`

Case

- `closed`
- `ownerEscalated`

Contact

- `contactCreatedFromLead`
- `contactMerged`
- `contactUpdatedByLead`

Contract

- `contractActivation`
- `contractApproval`
- `contractConversion`
- `contractExpiration`
- `contractTermination`

Lead

- `leadConverted`
- `leadMerged`

Opportunity

- `opportunityCreatedFromLead`

See Also:

NewsFeed
UserProfileFeed

FieldHistoryArchive

Represents field history values for all objects that retain field history. This object is only available to users with the "RetainFieldHistory" permission. This object is available in API version 29.0 and later.

Note: The FieldHistoryArchive object is currently available through a limited pilot program. Contact your salesforce.com representative to see if your organization qualifies.

Supported Calls

`describeSObjects()`, `query()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`FieldHistoryType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The name of the object containing the field history (for example, Account).
`Id`	**Type** ID **Properties** Defaulted on create, Filter, Group, idLookup, Sort **Description** The ID of the archived record. It's useful to have a field's ID for fields that you may have deleted. (Field names are not retained in history when you delete fields from Salesforce.)
`NewValue`	**Type** anyType **Properties** Nillable, Sort

Field Name	Details
	Description New value of the modified field.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** Previous value of the modified field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the object that contains the field.

FieldPermissions

Represents the enabled field permissions for the parent PermissionSet. To grant a user access to a field, associate a FieldPermissions record with a PermissionSet that's assigned to a user. FieldPermissions records are only supported in PermissionSet, not in Profile. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** The field's API name. This name must be prefixed with the `SobjectType`. For example, `Merchandise__c.Description__c`
`ParentId`	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Sort **Description** The `Id` of the field's parent PermissionSet.
`PermissionsEdit`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** If `true`, users assigned to the parent PermissionSet can edit this field. Requires `PermissionsRead` for the same field to be `true`.
`PermissionsRead`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** If `true`, users assigned to the parent PermissionSet can view this field. A FieldPermissions record must have at minimum `PermissionsRead` set to `true`, or it will be deleted.
`SobjectType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** The object's API name. For example, `Merchandise__c`.

Usage

FieldPermissions work similarly to ObjectPermissions. However, FieldPermissions includes a `Field` attribute to return the name of the field.

For example, the following query returns all FieldPermissions records that have at least the “Read” permission. The results includes the field, object, and permission set names.

```
SELECT SobjectType, Field, PermissionsRead, Parent.Name
FROM FieldPermissions
WHERE PermissionsRead = True
```

Include the field's parent object when querying FieldPermissions. For example, to find all rows that match the Account object's `Type` field, create the following query:

```
SELECT Id, SobjectType, Field
FROM FieldPermissions
WHERE Field = 'Account.Type' AND SobjectType = 'Account'
```

Both `SobjectType` and `Field` must be included in the `SELECT` line of the query. You must also provide the full API name of the field in the form of `SobjectType.Field` when querying for a field.

Special Properties for Field Permissions

The auto-number and formula fields have special rules for how field permissions work. Both have FieldPermissions records, but inserting and updating is limited to `PermissionsRead`. `PermissionsEdit` isn't allowed for either field type, since these fields must be read-only for users.

The following field types don't return a FieldPermissions record because they are assumed to always be readable.

- `Id`
- `CreatedById`
- `CreatedDate`
- `IsDeleted`
- `LastModifiedById`
- `LastModifiedDate`
- `SystemModStamp`

The following field types don't return a FieldPermissions record because they are assumed to always be readable and writable.

- `OwnerId`
- Master-detail custom (relationship) fields
- Universally required custom fields

As a result, the following query returns no records, even though users do have some access to some of the fields.

```
SELECT Field, SobjectType, PermissionsRead
FROM FieldPermissions
WHERE Field='Id'
```

To determine if a field can return a FieldPermissions record, you can call a `describeSObject()` on the field. For example, `describeSObject('Merchandise__c')`, returns all the properties of the Merchandise custom object, including field properties. If you're using a field whose `permissionable` property is `false` (such as any of the field types listed in this section), you can't query, insert, update, or delete any field permissions records, because none exist.

Working with Custom Activity Fields

While tasks and events are considered separate objects, they share a common set of activity custom fields. As a result, when a custom task field is created, a custom event field is also created, and vice versa. You can display the custom field on the event layout, task layout, or both event and task layouts.

Although custom activity fields are shared between tasks and events, you'll see separate FieldPermissions records for the task and event. However, changes made to one field permission record are automatically made to the other. For example, if you create a custom activity field, assign field permissions to it in a permission set, and run the following query, the query will return two records with the same permission value.

```
SELECT Field, Id, ParentId, PermissionsEdit, PermissionsRead, SobjectType
FROM FieldPermissions
WHERE SobjectType = 'event' OR SobjectType ='task'
```

If you then update one of the records with a different set of field permission values and run the query again, the same permission values for both records are returned.

Nesting Field Permissions

You can nest FieldPermissions in a PermissionSet query. For example, the following returns any permission sets where "Edit Read Only Fields" is `true`. Additionally, the result set will include both the "Read" and "Edit" field permission on the Merchandise object. This is done by nesting the SOQL with a field permission query using the relationship name for field permissions: `FieldPerms`.

```
SELECT PermissionsEditReadonlyFields,
(SELECT SobjectType, Field, PermissionsRead, PermissionsEdit
FROM FieldPerms
WHERE SobjectType = 'Merchandise__c')
FROM PermissionSet
WHERE PermissionsEditReadonlyFields = true
```

As a result, it's possible to traverse the relationship between the PermissionSet and any child related objects (in this case, FieldPermissions). You can do this from the PermissionSet object by using the child relationship (`ObjectPerms`, `FieldPerms`, and so on) or from the child object by referencing the PermissionSet with `Parent.`***`permission_set_attribute`***.

It's important to consider when to use a conditional `WHERE` statement to restrict the result set. To query based on an attribute on the permission set object, nest the SOQL with the child relationship. However, to query based on an attribute on the child object, you must reference the permission set parent attribute in your query.

The following two queries return the same columns with different results, based on whether you use the child relationship or parent notation.

```
SELECT PermissionsEditReadonlyFields,
(SELECT SobjectType, Field, PermissionsRead, PermissionsEdit
FROM FieldPerms
WHERE SobjectType = 'Merchandise__c')
FROM PermissionSet
WHERE PermissionsEditReadonlyFields = true
```

versus:

```
SELECT SobjectType, Field, PermissionsRead, PermissionsEdit, Parent.Name,
   Parent.PermissionsEditReadonlyFields
FROM FieldPermissions
WHERE SObjectType='Merchandise__c'
```

See Also:

PermissionSet
ObjectPermissions

FiscalYearSettings

Settings to define a custom or standard fiscal year for your organization. This object has a parent-child relationship with the Period object.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
Description	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Description of the setting.
EndDate	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** End date of the fiscal year.
IsStandardYear	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the fiscal year is a standard calendar year (`true`) or a custom fiscal year (`false`).
Name	**Type** string **Properties** Filter, Group, Sort **Description** A name for the fiscal year. Limit: 80 characters.
PeriodId	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description ID of the associated fiscal period.
`PeriodLabelScheme`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The numbering scheme used for fiscal periods.
`PeriodPrefix`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** The prefix of fiscal periods. For example, if `p` is the prefix, then the first period is “P1.”
`QuarterLabelScheme`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The numbering scheme used for fiscal quarters.
`QuarterPrefix`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** The prefix of fiscal quarters. For example, if “Q” is the prefix, then the fourth quarter would be “Q4.”
`StartDate`	**Type** date **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Start date of the fiscal year.
`WeekLabelScheme`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The numbering scheme used for weeks.
`WeekStartDay`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The name of the day that starts the week, for example `Monday` or `Sunday`
`YearType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Indicates one of two types of fiscal years, Standard or Custom. Standard denotes the standard Gregorian calendar, while Custom means a fiscal year with a custom structure.

See Also:

Period
Standard and Custom Object Basics

Folder

Represents a repository for a Document, EmailTemplate, Report, or Dashboard. Only one type of item can be contained in a folder.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- You must have the “Modify All Data” permission to create, update, or delete document folders, email template folders, or report folders.
- To query this object, no special permissions are needed.
- Customer Portal users can’t access this object.

Fields

Field	Details
`AccessType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. Indicates who can access the Folder. Available values include: • `Hidden`—Folder is hidden from everyone. • `Public`—Folder is accessible by all users. • `Shared`—Folder is accessible only by a User in a particular Group or UserRole. The API doesn’t allow you to view, insert, or update which group or Role the Folder is shared with.
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object’s name in a managed package and the changes are reflected in a subscriber’s organization. Label is **Folder Unique Name**. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`IsReadonly`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field	Details
	Description Indicates whether this Folder is read-only (`true`) or editable (`false`). Label is `Read Only`.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label of the folder as it appears in the user interface. Label is **Document Folder Label**.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects. This field can't be accessed unless the logged-in user has the "Customize Application" permission.
`Type`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** Required. Type of objects contained in the Folder. This field can't be updated. Available values include: • `Dashboard` • `Document`

Field	Details
	• `Email template` • `Report`

Usage

Only one type of item can be contained in a folder, either Document, EmailTemplate, Report, or Dashboard.

See Also:

Standard and Custom Object Basics

ForecastingAdjustment

This object represents an individual user's adjustment for a subordinate's forecast via a ForecastingItem. Available in API versions 26 and greater.

Note: This information only applies to Collaborative Forecasts.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AdjustedAmount`	**Type** double **Properties** Create, Filter, Sort, Update **Description** The revenue amount of an individual forecast item, after an adjustment.
`AdjustedQuantity`	**Type** double **Properties** Create, Filter, Sort, Update **Description** The quantity amount of an individual forecast item, after an adjustment. This field is available in API version 28 and later.

Field Name	Details
CurrencyIsoCode	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Restricted picklist, Sort, Update **Description** The currency code of the adjustment. If omitted, the default is the importing user's personal currency.
ForecastCategoryName	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** The category within the sales cycle that an opportunity is assigned to based on its opportunity stage. The standard forecast categories are Pipeline, Best Case, Commit, Omitted, and Closed. You can customize forecast category names. The forecast categories display information for that specific category; for example, Best Case only reflects amounts in the Best Case category.
ForecastingTypeId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the related ForecastingType.
ForecastingItemId	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the related ForecastingItem.
IsAmount	**Type** boolean

Field Name	Details
	Properties Defaulted on create, Filter, Group, Sort **Description** True indicates that the adjustment is made in a revenue amount. If false, then `IsQuantity` must be true. This field is available in API version 28 and later.
`IsQuantity`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** True indicates that the adjustment is made in a quantity amount. If false, then `IsAmount` must be true. This field is available in API version 28 and later.
`OwnerId`	**Type** reference **Properties** Create,Defaulted on create, Filter, Group, Sort **Description** The ID of the forecast owner.
`PeriodId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Period ID for the adjustment. Read only.
`ProductFamily`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort

Field Name	Details
	Description The Product Family for the adjustment. Read only. This field is available in API version 29 and later.
`StartDate`	**Type** date **Properties** Create, Filter, Group, Sort **Description** The start of the adjustment, expressed as month and year. The date can include any day in a given month. Stored using the first date of the month.

Usage

Use this object to obtain a user's adjustment detail for a specified ForecastingItem.

Note: Beginning with API version 30.0, organizations can have more than one forecasting type enabled. The `ForecastingQuota`, `ForecastingAdjustment`, `ForecastingItem`, and `ForecastingFact` objects can all have records with different `ForecastingTypeId` values. Use the ForecastingType object to determine the ID for each forecast type and then filter `ForecastingQuota`, `ForecastingAdjustment`, `ForecastingItem`, or `ForecastingFact` records as necessary.

See Also:

ForecastingFact
ForecastingItem
ForecastingQuota

ForecastingFact

This is a read-only object linking a ForecastingItem with its opportunities, such as opportunities that share the same owner or forecast category and have a closing date within the period of the forecasting item. Available in API versions 26 and greater.

Note: This information only applies to Collaborative Forecasts.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
ForecastCategoryName	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A forecast category is the category within the sales cycle that an opportunity is assigned to based on its opportunity stage. The standard forecast categories are Pipeline, Best Case, Commit, Omitted, and Closed. You can customize forecast category names for your organization. The forecast categories display information for that specific category; for example, Best Case only reflects amounts in the Best Case category.
ForecastedObjectId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Contains the Split ID of the forecasted OpportunitySplit object if the forecast data source is opportunity splits or the OpportunityLineItem ID of the forecasted opportunity if the data source is product families. If the data source is product families and the opportunity has no line item, this field is Null. If the forecast data source is opportunities, this field is Null. This field is available in API version 29 and later. Read only.
ForecastingItemId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the ForecastingItem.
ForecastingTypeId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the related ForecastingType.

Field Name	Details
OpportunityId	**Type** reference **Properties** Filter, Group, Sort **Description** The opportunity ID.
OwnerId	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the opportunity owner.
PeriodId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Period ID for the forecast.

Usage

Use this object to get information about opportunities linked to forecasting items.

Note: Beginning with API version 30.0, organizations can have more than one forecasting type enabled. The `ForecastingQuota`, `ForecastingAdjustment`, `ForecastingItem`, and `ForecastingFact` objects can all have records with different `ForecastingTypeId` values. Use the ForecastingType object to determine the ID for each forecast type and then filter `ForecastingQuota`, `ForecastingAdjustment`, `ForecastingItem`, or `ForecastingFact` records as necessary.

s

See Also:

ForecastingAdjustment
ForecastingItem
ForecastingQuota

ForecastingItem

This is a read-only object used for individual forecast amounts. Users see amounts based on their perspectives and forecast roles. The amounts users see include one of the following when forecasting in revenue: `AmountWithoutAdjustments`, `AmountWithoutManagerAdjustment`, `ForecastAmount`, `OwnerOnlyAmount`. The amounts users see include one of the following when forecasting in quantity: `QuantityWithoutAdjustments`, `QuantityWithoutManagerAdjustments`, `ForecastQuantity`, `OwnerOnlyQuantity`. Additionally, note that users:

- with the “View All Forecasts” permission have access to all ForecastingItem fields.
- without the “View All Forecasts” permission have access to all fields for their own subordinates.

Available in API versions 26 and greater.

Note: This information only applies to Collaborative Forecasts.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`AmountWithoutAdjustments`	**Type** double **Properties** Filter, Sort, Nillable **Description** The sum of all of a person’s owned revenue opportunities and also his or her subordinates’ opportunities, without adjustments. Subordinates include everyone reporting up to a person in the forecast hierarchy.
`AmountWithoutManagerAdjustment`	**Type** double **Properties** Filter, Sort, Nillable **Description** The forecast number as seen by the forecast owner. This is the sum of the owner’s revenue opportunities and his or her subordinates’ opportunities, including adjustments made on the subordinates’ forecasts. *It doesn’t include adjustments made by forecast managers above the owner in the forecast hierarchy.*

Field Name	Details
ForecastAmount	**Type** double **Properties** Filter, Sort, Nillable **Description** The revenue forecast from the forecast manager's perspective and the sum of the owner's and subordinates' opportunities, including all forecast adjustments.
ForecastCategoryName	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A forecast category is the category within the sales cycle that an opportunity is assigned to based on its opportunity stage. The standard forecast categories are Pipeline, Best Case, Commit, Omitted, and Closed. You can customize forecast category names for your organization. The forecast categories display information for that specific category; for example, Best Case only reflects amounts in the Best Case category.
ForecastQuantity	**Type** double **Properties** Filter, Sort, Nillable **Description** The quantity forecast from the forecast manager's perspective and the sum of the owner's and subordinates' opportunities, including all forecast adjustments. This field is available in API version 28 and later.
ForecastingTypeId	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the related ForecastingType.

Field Name	Details
HasAdjustment	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** A flag that indicates if an adjustment has been made based on the point of view of the viewer.
IsAmount	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** True indicates that the adjustment is made in a revenue amount. If false, then `IsQuantity` must be true. This field is available in API version 28 and later.
IsQuantity	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** True indicates that the adjustment is made in a quantity amount. If false, then `IsAmount` must be true. This field is available in API version 28 and later.
IsUpToDate	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** A flag indicating whether or not a specific forecasting item reflects current information. For example, if users are making adjustments which are in process, the item won't be up-to-date.
OwnerId	**Type** reference

Field Name	Details
	Properties Filter, Group, Sort **Description** The ID of the forecast owner.
`OwnerOnlyAmount`	**Type** double **Properties** Filter, Sort, Nillable **Description** The sum of all of a person's revenue opportunities, without adjustments.
`OwnerOnlyQuantity`	**Type** double **Properties** Filter, Sort, Nillable **Description** The sum of all of a person's quantity opportunities, without adjustments. This field is available in API version 28 and later.
`ParentForecastingItemId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the ForecastingItem that the current item rolls up to.
`PeriodId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Period ID for the forecast.

Field Name	Details
`ProductFamily`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** The product family of the forecast item. This field is available in API version 29 and later. Read only.
`QuantityWithoutAdjustments`	**Type** double **Properties** Filter, Sort, Nillable **Description** The sum of a person's owned quantity opportunities and also his or her subordinates' opportunities, without adjustments. Subordinates include everyone reporting up to a person in the forecast hierarchy. This field is available in API version 28 and later.
`QuantityWithoutManagerAdjustments`	**Type** double **Properties** Filter, Sort, Nillable **Description** The forecast number as seen by the forecast owner. This is the sum of the owner's quantity opportunities and his or her subordinates' opportunities, including adjustments made on the subordinates' forecasts. *It doesn't include adjustments made by forecast managers above the owner in the forecast hierarchy.* This field is available in API version 28 and later.

Usage

Use this object to obtain individual forecast amounts, either with or without adjustments, based on a user's perspective and forecast role.

Note: Beginning with API version 30.0, organizations can have more than one forecasting type enabled. The `ForecastingQuota`, `ForecastingAdjustment`, `ForecastingItem`, and `ForecastingFact` objects can all have records with different `ForecastingTypeId` values. Use the ForecastingType object to determine the ID for each forecast type and then filter `ForecastingQuota`, `ForecastingAdjustment`, `ForecastingItem`, or `ForecastingFact` records as necessary.

See Also:

ForecastingAdjustment
ForecastingFact
ForecastingQuota

ForecastingQuota

This object represents an individual user's quota for a specified time period. The "Manage Quotas" user permission is required for creating, updating, or deleting quotas. (Users can only edit their subordinates' quotas, not their own.) The "View All Forecasts" permission is required to View any user's forecast, regardless of the forecast role hierarchy. Available in API versions 25 and greater. Forecast managers can view the forecasts of subordinates who report to them in the forecast hierarchy.

Note: This information only applies to Collaborative Forecasts.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`CurrencyIsoCode`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Restricted picklist, Sort, Update **Description** The currency code of the quota. If omitted, the default is the importing user's personal currency.
`ForecastingTypeId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the related ForecastingType.
`IsAmount`	**Type** boolean

Field Name	Details
	Properties Defaulted on create, Filter, Group, Sort **Description** True indicates that the adjustment is made in a revenue amount. If false, then `IsQuantity` must be true. This field is available in API version 28 and later.
`IsQuantity`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** True indicates that the adjustment is made in a quantity amount. If false, then `IsAmount` must be true. This field is available in API version 28 and later.
`PeriodId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Period ID for the quota. Read only.
`ProductFamily`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** The product family for the quota. This field is available in API version 29 and later.
`QuotaAmount`	**Type** currency **Properties** Create, Filter, Sort, Update **Description** The revenue quota amount for an individual user and for a specific period.

Field Name	Details
`QuotaOwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID that identifies the quota owner.
`QuotaQuantity`	**Type** double **Properties** Create, Filter, Sort, Update **Description** The quantity quota amount for an individual user and for a specific period. This field is available in API version 28 and later.
`StartDate`	**Type** date **Properties** Create, Filter, Group, Sort, Update **Description** The start of the quota, expressed as month and year. The date can include any day in a given month. Stored using the first date of the month.

Usage

Use this object to get an individual user's quota for a specified time period.

Note: Beginning with API version 30.0, organizations can have more than one forecasting type enabled. The `ForecastingQuota`, `ForecastingAdjustment`, `ForecastingItem`, and `ForecastingFact` objects can all have records with different `ForecastingTypeId` values. Use the ForecastingType object to determine the ID for each forecast type and then filter `ForecastingQuota`, `ForecastingAdjustment`, `ForecastingItem`, or `ForecastingFact` records as necessary.

See Also:

ForecastingAdjustment
ForecastingFact
ForecastingItem

ForecastingType

This object is used to identify the forecast type associated with `ForecastingAdjustment`, `ForecastingQuota`, `ForecastingFact`, and `ForecastingItem` objects.

Available in API versions 30 and greater.

Note: This information only applies to Collaborative Forecasts.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`DeveloperName`	**Type** string **Properties** Filter, Group, Sort **Description** The name of the forecasting type. Allowed values include: • `OpportunityRevenue` : Opportunities - Revenue • `OpportunityQuantity` : Opportunities - Quantity • `OpportunitySplitRevenue` : Opportunity Revenue Splits - Revenue • `OpportunityLineItemRevenue` : Product Families - Revenue • `OpportunityLineItemQuantity` : Product Families - Quantity The `DeveloperName` is called `name` in the Metadata API and Forecasting Type in custom reports.
`IsActive`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the forecasting type is currently enabled in the organization.
`IsAmount`	**Type** boolean

Field Name	Details
	Properties Defaulted on create, Filter, Group, Sort **Description** Indicates whether the forecasting type is based on the revenue measurement.
`IsQuantity`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the forecasting type is based on the quantity measurement.
`Language`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The language of the forecasting type.
`MasterLabel`	**Type** string **Properties** Filter, Group, Sort **Description** Master label for this Forecasting Type value. This display value is the internal label that does not get translated.

Usage

Use this object to identify the forecast type of `ForecastingAdjustment`, `ForecastingQuota`, `ForecastingFact`, and `ForecastingItem` objects.

ForecastShare

Represents the sharing of a customizable forecast at a given role and territory.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can’t access this object.

Fields

Field	Details
`AccessLevel`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A value (`Read`, `Edit`, or `All`) that represents the type of sharing being allowed.
`CanSubmit`	**Type** boolean **Properties** Filter, Group, Sort **Description** Indicates whether the user or group can submit forecasts (`True`) or not (`False`).
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. There are many possible values, including: • `Manual`—The User or Group has access because a User has manually shared the forecast with them. • `Owner`—The User is the owner of the forecast.
`UserOrGroupId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID representing the User or Group being granted access.

Field	Details
UserRoleId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the UserRole associated with this object.

Usage

This object allows you to determine which users and groups can view or submit forecasts owned by other users.

GoogleDoc

Represents a link to a Google Document. This object is available in API version 14.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

This object is available in **All** Editions except **Database.com** for Google Apps Premier Edition accounts. See the Salesforce online help for more information.

Fields

Field	Details
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The name of the Google document.
Owner	**Type** reference **Properties** Create, Defaulted on create, Filter, Nillable, Update **Description** The ID of the user who currently owns this Google Document. Default value is the user logged in to the API to perform the create.

Field	Details
ParentId	**Type** reference **Properties** Create, Filter **Description** Required. ID of the attachment's parent object. The following objects are supported as parents of Google documents: Account, Asset, Campaign, Case, Contact, Contract, Custom Objects, Lead, Opportunity, Product2, and Solution.
Url	**Type** string **Properties** Create, Filter, Nillable, Update **Description** The URL of the Google document.

Group

Groups are sets of users. They can contain individual users, other groups, the users in a particular role or territory, or the users in a particular role or territory plus all of the users below that role or territory in the hierarchy.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `search()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can’t access this object.

Fields

Field	Details
DefaultDivision	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** This record’s default division. Only applicable if divisions are enabled for your organization.

Field	Details
DeveloperName	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Group Name** in the user interface. This field is available in API version 24.0 and later. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
DoesIncludeBosses	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the managers have access (`true`) or do not have access (`false`) to records shared with members of the group. This field is only available for public groups. This field is available in API version 18.0 and later.
DoesSendEmailToMembers	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the email is sent (`true`) or not sent (`false`) to the group members. The email is sent to queue members as well.
Email	**Type** email **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Email address for a group of type Case. Applies only for a case queue.

Field	Details
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of the group. Corresponds to **Label** on the user interface.
OwnerId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the user who owns the group.
RelatedId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** For Groups of type "Role," the ID of the associated UserRole.
Type	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** Required. Type of the Group. One of the following values: • `Regular`—Standard Public Group. When you `create()` a Group, its type must be `Regular`, unless a partner portal is enabled for the organization, in which case the type can be `Regular` or `PRMOrganization`. • `Role`—Public Group that includes all of the User records in a particular UserRole. • `RoleAndSubordinates`—Public Group that includes all of the User records in a particular UserRole and all of the User records in any subordinateUserRole. • `Organization`—Public Group that includes all of the User records in the organization. This Group is read-only. • `Case`—Public group of users, members of a queue that can own a Case. • `Lead`—Public group of users, members of a queue that can own a Lead. • `PRMOrganization`—Public Group that includes all of the partners in an organization that has the partner portal feature enabled.

Field	Details
	• `Queue`—Public Group that includes all of the User records that are members of a queue. • `Territory`—Public Group that includes all of the User records in an organization that has the territory feature enabled. • `TerritoryAndSubordinates`—Public Group that includes all of the User records in a particular UserRole and all of the User records in any subordinateUserRole. • `Collaboration`—Chatter group. Only `Regular`, `Case`, and `Lead` can be used when creating a group. The other values are reserved.

Usage

Unlike users, this object can be deleted. Any User can access this object—no special permissions are needed.

Only public groups are accessible via the API. Personal groups are not available.

Note: In API version 13.0 and later, if you delete a public group, it will be deleted even if it has been used in sharing, consistent with the behavior for UserRole. In versions before 13.0, such sharing prevents the record from being deleted.

See Also:

GroupMember
Standard and Custom Object Basics

GroupMember

Represents a User or Group that is a member of a public group.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users cannot access this object.

Fields

Field	Details
`GroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Group.

Field	Details
UserOrGroupId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the User or Group that is a direct member of the group.

Usage

A record exists for every User or Group who is a direct member of a public group whose `Type` field is set to Regular. User records that are indirect members of Regular public groups are not listed as group members. A User can be an indirect member of a group if he or she is in a UserRole above the direct group member in the hierarchy, or if he or she is a member of a group that is included as a subgroup in that group.

If you attempt to create a record that matches an existing record, system simply returns the existing record.

See Also:

Standard and Custom Object Basics

HashtagDefinition

HashtagDefinition represents hashtag (#) topics in public Chatter posts and comments. Public posts and comments include those on profiles and in public groups, but not those on records or in private groups. This object is available in API version 26.0 and later.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
HashtagCount	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of times a hashtag topic is used.
Name	**Type** string **Properties** Filter, Nillable, Sort

Field Name	Details
	Description The string of characters following the hashtag (#) in a hashtag topic.
`NameNorm`	**Type** string **Properties** Filter, Nillable, Sort **Description** The string of characters following the hashtag (#) in a hashtag topic, normalized to remove capitalization and punctuation.
`NetworkId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Identifier of the community to which the HashtagDefinition belongs. This field is available only if Salesforce Communities is enabled in your organization.

Usage

Use this object to identify public hashtag topics and see how often they're used.

See Also:

FeedItem
FeedComment

Holiday

Represents a period of time during which your customer support team is unavailable. Business hours and escalation rules associated with business hours are suspended during any holidays with which they are affiliated.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
ActivityDate	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** If the Holiday `IsAllDay` flag is set to `true` (indicating that it is an all-day holiday), then the holiday due date information is contained in the `ActivityDate` field. This field is a date field with a timestamp that is always set to midnight in the Coordinated Universal Time (UTC) time zone. The timestamp is not relevant, and you should not attempt to alter it to account for any time zone differences.
Description	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Text description of the holiday.
EndTimeInMinutes	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The end time of the holiday in minutes.
IsAllDay	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the duration of the holiday is all day (`true`) or not (`false`).
IsRecurrence	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field	Details
	Description Indicates whether the holiday is scheduled to repeat itself (`true`) or only occurs once (`false`). This is a read only field on update, but not on create. If this field value is `true`, then any recurrence fields associated with the given recurrence type must be populated.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The name of the holiday.
`RecurrenceDayOfMonth`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The day of the month on which the holiday repeats.
`RecurrenceDayOfWeekMask`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The day or days of the week on which the holiday repeats. This field contains a bitmask. For each day of the week, the values are as follows: • Sunday = `1` • Monday = `2` • Tuesday = `4` • Wednesday = `8` • Thursday = `16` • Friday = `32` • Saturday = `64` Multiple days are represented as the sum of their numerical values. For example, Tuesday and Thursday = 4 + 16 = 20.
`RecurrenceEndDateOnly`	**Type** date

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The last date on which the holiday repeats. For multiday recurring events, this is the day on which the last occurrence starts.
`RecurrenceInstance`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The frequency of the recurring holiday. For example, `2nd` or `3rd`.
`RecurrenceInterval`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The interval between recurring holidays.
`RecurrenceMonthOfYear`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The month of the year on which the event repeats.
`RecurrenceStartDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The date when the recurring holiday begins. Must be a date and time before `RecurrenceEndDateOnly`.
`RecurrenceType`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update

Field	Details
	Description Indicates how often the holiday repeats. For example, daily, weekly, or every Nth month (where “Nth” is defined in `RecurrenceInstance`).
`StartTimeInMinutes`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The start time of the holiday in minutes.

Usage

Use this object to view and update holidays, which specify dates and times at which associated business hours and escalation rules are suspended.

Idea

Represents an idea on which users are allowed to comment and vote, for example, a suggestion for an enhancement to an existing product or process. This object is available in API version 12 and later.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Note: For other standard objects, the `describeLayout()` call returns the `recordTypeMappings` section that contains the layout ID and picklist values for each record type. However, the `recordTypeMappings` section and the fields it includes are not available for the Idea object.

When performing a SOSL search on Idea objects, IdeaComment objects are also searched.

Fields

Field	Details
`AttachmentBody`	**Type** base64 **Properties** Create, Nillable, Update **Description** File data for the attachment. This field is available in API version 28.0 and later.

Field	Details
AttachmentContentType	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Type of the attachment. This field is available in API version 28.0 and later.
AttachmentLength	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Size of the attachment in bytes. This field is available in API version 28.0 and later.
AttachmentName	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Name of the attachment. This field is available in API version 28.0 and later.
Body	**Type** textarea **Properties** Create, Nillable, Update **Description** Description of the Idea.
Categories	**Type** multipicklist **Properties** Create, Filter, Nillable, Update **Description** Customizable multi-select picklist used to organize Ideas into logical groupings.

Field	Details
	Note: This field is only available if your organization has the `Categories` field enabled. This field is enabled by default in organizations created after API version 14 was released. If the `Categories` field is enabled, API versions 13 and earlier do not have access to either the `Categories` or `Category` fields.
`Category`	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** Customizable picklist of values used to organize Ideas into logical groupings. **Note:** This field is not available if your organization has the multi-select `Categories` field enabled.
`CommunityId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The zone ID associated with the idea. Once you create an idea, you can't change the zone ID associated with that idea. **Note:** API version 12 does not support zone ID. If you create an idea in version 12, your idea is automatically posted to the oldest zone that you have permission to access.
`CreatorFullPhotoUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** URL of the user's profile photo. This field is available in API version 28.0 and later.
`CreatorName`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Name of the user who posted the idea or commented on the idea. This field is available in API version 28.0 and later.
`CreatorSmallPhotoUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** URL of the user's thumbnail photo. This field is available in API version 28.0 and later.
`CurrencyIsoCode`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.
`IdeaThemeID`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Identifies the idea theme associated with the idea.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsHtml`	**Type** boolean

Field	Details
	Properties Defaulted on create, Filter, Group, Sort **Description** Read-only. If this value is `true`, your organization has the Ideas HTML editor enabled, and the Idea `Body` may contain HTML. If this value is `false`, the HTML editor is disabled and the Idea `Body` only contains regular text.
`IsMerged`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Read only. Indicates whether the idea has been merged with a parent idea (`true`) or not (`false`). You can't vote for or add comments to a merged idea. **Note:** In API version 27, `IsMerged` replaces `IsLocked`. Existing formula fields that use `IsLocked` must be edited to use `IsMerged`.
`LastCommentDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date and time the last comment (child IdeaComment object) was added.
`LastCommentId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Read only. The ID of the last comment (child IdeaComment object).
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.

Field	Details
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`NumComments`	**Type** int **Properties** Defaulted on create, Filter, Group, Nillable, Sort **Description** The number of comments (child IdeaComment objects) that users have submitted for the given idea.
`ParentIdeaId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID associated with this idea's parent idea. When multiple ideas are merged together, one idea becomes the parent (master) of the other ideas. The `ParentIdeaId` is automatically set when you merge ideas.
`RecordTypeId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the record type assigned to this object.
`Status`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Customizable picklist of values used to specify the status of an idea.

Field	Details
`Title`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The descriptive title of the idea.
`VoteScore`	**Type** double **Properties** Filter, Nillable, Sort **Description** The internal score of the Idea, used to sort Ideas on the Popular tab in the application user interface. The internal algorithm that determines the score gives older votes less weight than newer votes, simulating exponential decay. The score itself does not display in the application user interface. **Note:** Unlike other fields of type double, you can't use a SOQL aggregate function with this field.
`VoteTotal`	**Type** double **Properties** Filter, Nillable, Sort **Description** An Idea's total number of points. Each vote a user makes is worth ten points, therefore the value of this field is ten times the number of votes an idea has received. **Note:** Unlike other fields of type double, you can't use a SOQL aggregate function with this field.

Note: If you are importing Idea data and need to set the value for an audit field, such as `CreatedDate`, contact salesforce.com. Audit fields are automatically updated during API operations unless you request to set these fields yourself..

Usage

Use this object to track ideas, which are written suggestions on which users can vote and comment.

See Also:

IdeaComment
Vote

IdeaComment

Represents a comment that a user has submitted in response to an idea.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Note:

When performing a SOSL search on IdeaComment objects, Idea objects are also searched.

Fields

Field	Field Type
`CommentBody`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Body of the submitted comment.
`CommunityId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The zone ID associated with the idea. Once you create an idea, you can't change the zone ID associated with that idea. **Note:** API version 12 does not support zone ID. If you create an idea in version 12, your idea is automatically posted to the oldest zone that you have permission to access.
`CreatorFullPhotoUrl`	**Type** string

Field	Field Type
	Properties Filter, Group, Nillable, Sort **Description** URL of the user's profile photo. This field is available in API version 28.0 and later.
`CreatorName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Name of the user who posted the idea or commented on the idea. This field is available in API version 28.0 and later.
`CreatorSmallPhotoUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** URL of the user's thumbnail photo. This field is available in API version 28.0 and later.
`IdeaId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the idea on which this comment was made.
`IsHtml`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Read-only. If this value is `true`, your organization has the Ideas HTML editor enabled, and the `CommentBody` field may contain HTML. If this value is `false`, the HTML editor is disabled and the `CommentBody` field only contains regular text.

Field	Field Type
`UpVotes`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Total number of up votes for the question.

Note: If you import these records, and need to set the value for an audit field, such as `CreatedDate`, contact salesforce.com. Audit fields are automatically updated during API operations unless you request to set these fields yourself.

Usage

Use this object to track comments on ideas, which are users' text responses to ideas.

See Also:

Idea
Vote

IdeaReputation

Represents a collection of statistics and scores derived from a user's activity within an Ideas zone or internal organization. This object is available in API version 28.0 and later.

Supported Calls

`query()`, `retrieve()`,

Fields

Field	Details
`CommentCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of comments a user has created in a zone or the internal organization. This number excludes comments the user creates on his or her own idea.
`CommentsReceivedCount`	**Type** int

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** The number of comments a user has received in a zone or the internal organization.
`ContextId`	**Type** reference **Properties** Filter, Group, Namepointing, Nillable, Sort **Description** The ID of the zone or internal organization.
`DownVotesGivenCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of down votes a user has given in a zone or the internal organization.
`DownVotesReceivedCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of down votes a user has received in a zone or the internal organization.
`IdeaCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of ideas a user has created in a zone or the internal organization.
`ReputationLevel`	**Type** string **Properties** Nillable

Field	Details
	Description The reputation level that a user has achieved based on their score in a zone or within an organization.
`Score`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total score of a user's activity within a zone or within an organization.
`UpVotesGivenCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of up votes a user has given in a zone or the internal organization. This number doesn't include the default vote the system applies when the user creates the idea.
`UpVotesReceivedCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of up votes a user has received in a zone or the internal organization.
`UserId`	**Type** reference **Properties** Filter, Group, Sort **Description** The user ID associated with the reputation.

Usage

Use to query a user's reputation within a zone.

IdeaReputationLevel

Represents a reputation level within an Ideas zone or internal organization and is used by the system to calculate reputation. You can create up to 25 levels per zone or internal organization. This object is available in API version 28.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`ContextId`	**Type** reference **Properties** Create, Filter, Group, Namepointing, Sort, Update **Description** The ID of the zone or internal organization.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Name of the reputation level. The name must be unique within the zone or internal organization. Maximum size is 50 characters.
`Threshold`	**Type** int **Properties** Create, Filter, Group, Sort, Update **Description** Minimum number of points for this level. The threshold must be unique within the zone or internal organization and must be greater than or equal to zero.

Usage

Use to create or edit reputation levels for an Ideas zone or internal organization.

IdeaTheme

Represents an invitation to community members to submit ideas that are focused on a specific topic. This object is available in API version 26 and later.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`,

Fields

Field Name	Details
`Categories`	**Type** multipicklist **Properties** Create, Filter, Nillable, Update **Description** Customizable multi-select picklist used to organize ideas and idea themes into logical groupings.
`CommunityId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, **Description** The zone ID associated with the idea theme.
`CurrencyIsoCode`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.
`Description`	**Type** textarea **Properties** Create, Nillable, Update

Field Name	Details
	Description Description of the idea theme.
`EndDate`	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** Date marking the end of the idea theme.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`StartDate`	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** Date that the idea theme begins.
`Status`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Customizable picklist of values used to specify the status of the idea theme.
`Title`	**Type** string **Properties** Create, Filter, Group, idLookup, Namefield, Sort, Update

Field Name	Details
	Description Title of the idea theme.

Usage

Use the object to track ideas that are submitted to an idea theme.

KnowledgeableUser

Represents a user identified as knowledgeable about a specific topic, and ranks them relative to other knowledgeable users. This object is available in API version 31.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`NetworkId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the community the topic exists in. This field is available only if Salesforce Communities is enabled for your organization.
`RawRank`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Rank of this user's knowledge on the topic relative to other users.
`TopicId`	**Type** reference **Properties** Filter, Group, Sort **Description** Unique ID for the topic in Salesforce.

Field Name	Details
`UserId`	**Type** reference **Properties** Filter, Group, Sort **Description** Unique ID for the user in Salesforce.

KnowledgeArticle

Provides read-only access to an article and the ability to delete the master article. Unlike KnowledgeArticleVersion, the ID of a KnowledgeArticle record is identical irrespective of the article's version (status). For more information on articles and article types, see “Managing Articles and Translations” and “Managing Article Types” in the Salesforce online help.

Supported Calls

`delete()`, `describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

A user must have the “View Articles” permission enabled. Salesforce Knowledge users, unlike Customer Portal and partner portal users, must also be granted the `Knowledge User` feature license.

Fields

Field Name	Details
`ArchivedById`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the user who archived the article.
`ArchivedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date when the article was archived.
`ArticleNumber`	**Type** string

Field Name	Details
	Properties Autonumber, Defaulted on create, Filter, Sort **Description** The unique number automatically assigned to the article when it's created. You can't change the format or value for this field.
`CaseAssociationCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of cases attached to the article.
`FirstPublishedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date when the article was first published.
`LastPublishedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date when the article was last published.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update

Field Name	Details
	Description The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`MasterLanguage`	**Type** picklist **Properties** Filter, Restricted picklist **Description** The article's original language. Only accessible if your knowledge base supports multiple languages.

Usage

Use this object to query or retrieve articles. KnowledgeArticle can be used in a SOQL clause, but doesn't provide access to the fields from the article. Provides read-only access to an article and the ability to delete the master article.

See Also:

Articles
Data Categories
KnowledgeArticleVersion

KnowledgeArticleVersion

Provides a global view of standard article fields across all article types depending on their version. Use this object to:

- Query or search generically across multiple article types.
- Filter on a specific version.
- Update custom fields and updateable standard fields in draft versions.

When you query on the archived article, the results include both the article and the article's archived versions.

Note: You can't update draft translations with the API.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `query()`, `retrieve()`, `search()`, and `update()`.

Note:

- You can only update draft versions.
- To create, update, or delete a Knowledge article version, use the call on *ArticleType*_kav, where *ArticleType* is the name of the article's type. For example, to delete use *ArticleType*_kav.delete().

Special Access Rules

A user must have the “View Articles” permission enabled. Salesforce Knowledge users, unlike Customer Portal and partner portal users, must also be granted the `Knowledge User` feature license in order to view and edit article versions.

Fields

Field Name	Details
`ArchivedById`	**Type** reference **Properties** Filter, Nillable **Description** The ID of the user who archived the article.
`ArchivedDate`	**Type** dateTime **Properties** Filter, Nillable **Description** The date when the article was archived.
`ArticleNumber`	**Type** string **Properties** Autonumber, Defaulted on create, Filter **Description** The unique number automatically assigned to the article when it's created. You can't change the format or value for this field.
`ArticleType`	**Type** string **Properties** Defaulted on create,Filter **Description** Indicates the API Name of the article type. The `ArticleType` is assigned to the article when it's created. You can't change the value of this field. This field is available in API version 26.0 and later.
`FirstPublishedDate`	**Type** dateTime

Field Name	Details
	Properties Filter, Nillable **Description** The date when the article was first published.
`IsLatestVersion`	**Type** boolean **Properties** Defaulted on create, Group, Filter, Sort **Description** Indicates whether the article is the most current version. (`true`) or not (`false`). This field is available in API version 24.0 and later.
`IsMasterLanguage`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the article has one or more translations associated with it (`true`) or not (`false`). Only accessible if your knowledge base supports multiple languages.
`IsOutOfDate`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the source article has been updated since this translated version was created (`true`) or not (`false`). Only accessible if your knowledge base supports multiple languages.
`IsVisibleInApp`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Required. Indicates whether the article is visible in the Articles tab (`true`) or not (`false`).

Field Name	Details
IsVisibleInCsp	**Type** boolean **Properties** Defaulted on create, Filter **Description** Required. Indicates whether the article is visible in the Customer Portal (`true`) or not (`false`).
IsVisibleInPkb	**Type** boolean **Properties** Defaulted on create, Filter **Description** Required. Indicates whether the article is visible in the public knowledge base (`true`) or not (`false`).
IsVisibleInPrm	**Type** boolean **Properties** Defaulted on create, Filter **Description** Required. Indicates whether the article is visible in the partner portal (`true`) or not (`false`).
KnowledgeArticleId	**Type** reference **Properties** Filter **Description** The ID of the article independent from its version. The value for this field is retrieved from the `Id` field of the KnowledgeArticle object.
Language	**Type** picklist **Properties** Filter, Restricted picklist **Description** The language that the article is written in, such as `French` or `Chinese (Traditional)`.

Field Name	Details
	Querying or searching articles in SOQL or SOSL requires that you specify the `Language` field in the WHERE clause. The language must be the same for all article types.
`LastPublishedDate`	**Type** dateTime **Properties** Filter, Nillable **Description** The date when the article was last published.
`MasterVersionId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the source article, if this is the translation of a source article. Only accessible if your knowledge base supports multiple languages.
`OwnerId`	**Type** reference **Properties** Filter **Description** The ID of the article's owner.
`PublishStatus`	**Type** picklist **Properties** Filter, Restricted picklist **Description** The publication status for the article: • `Draft`: any draft articles. • `Online`: articles published in Salesforce Knowledge. • `Archived`: archived articles. A user must have the “Manage Articles” permission enabled to use `Online`. Querying or searching articles in SOQL or SOSL requires that you specify either the `PublishStatus` or the `Id` field in the WHERE clause. You can search for only one publication status per article type in a single SOSL query.

Field Name	Details
	When searching for articles with a `PublishStatus` of `Archived`, also check that `IsLatestVersion` equals `false` in your WHERE clause.
`SourceId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the source from which the article was created (Case or Reply).
`Summary`	**Type** textarea **Properties** Filter, Nillable **Description** Summary of the article. Maximum size is 1000 characters.
`Title`	**Type** string **Properties** Defaulted on create, Filter, idLookup **Description** Required. Article's title. Maximum size is 255 characters.
`TranslationCompletedDate`	**Type** dateTime **Properties** Filter, Nillable **Description** Date and time when the article was last translated. Only accessible if your knowledge base supports multiple languages.
`TranslationExportedDate`	**Type** dateTime **Properties** Filter, Nillable **Description** Date and time when the article was last exported for translation. Only accessible if your knowledge base supports multiple languages.

Field Name	Details
TranslationImportedDate	**Type** dateTime **Properties** Filter, Nillable **Description** Date and time when the article was last imported for translation. Only accessible if your knowledge base supports multiple languages.
UrlName	**Type** string **Properties** Filter **Description** Required. Represents the article's URL. Can contain alphanumeric characters and hyphens but can't begin or end with a hyphen. `UrlName` is case-sensitive and its maximum size is 255 characters.
ValidationStatus	**Type** picklist **Properties** Filter, Group **Description** Shows whether the content of the article has been validated. Default values are `Validated` and `Not Validated`. This field is available in API version 24.0 or later.
VersionNumber	**Type** int **Properties** None **Description** The number assigned to a version of an article. This field is available in API version 24.0 and later.

Usage

Use this object to query, retrieve, or search for articles across all article types depending on their version. You can update draft master articles. Additionally, you can delete articles that are not drafts. Client applications can use KnowledgeArticleVersion with `describeDataCategoryGroups()` and `describeDataCategoryGroupStructures()` to return the category groups and the category structure associated with Salesforce Knowledge.

Alternatively, client applications can use the article type `API Name` followed by `__kav` to query, retrieve, or search for articles for a specific article type.

To access an article independent of its version, use the KnowledgeArticle object.

SOQL Samples

The following SOQL clause uses KnowledgeArticleVersion to query all published articles from **all article types** complying with the classification specified in the WITH DATA CATEGORY clause:

```
SELECT Title, Summary
FROM KnowledgeArticleVersion
WHERE PublishStatus='Online'
AND Language = 'en_US'
WITH DATA CATEGORY Geography__c ABOVE_OR_BELOW europe__c AND Product__c BELOW All__c
```

The following SOQL clause uses the article type `API Name` to limit the query to all draft articles from the `Offer__kav` article type:

```
SELECT Id, Title
FROM Offer__kav
WHERE PublishStatus='Draft'
AND Language = 'en_US'
WITH DATA CATEGORY Geography__c AT (france__c,usa__c) AND Product__c ABOVE dsl__c
```

The following SOQL clause uses KnowledgeArticleVersion to query the Ids of all archived versions of a particular article:

```
SELECT Id
FROM KnowledgeArticleVersion
WHERE PublishStatus='Archived'
AND IsLatestVersion=false
AND KnowledgeArticleId='kA1D00000001PQ6KAM'
```

SOQL and SOSL with KnowledgeArticleVersion

- Always filter on a single value of `PublishStatus` unless the query filters on one or more primary key IDs. To support security, only users with the “Manage Articles” permission see articles whose `PublishStatus` value is `Draft`.
- Archived article versions are stored in the articletype_kav object. To query archived article versions, specify the article `Id` and set `sLatestVersion='0'`.
- Always filter on a single value of `Language`. However, in SOQL, you can filter on more than one `Language` if there is a filter on `Id` or `KnowledgeArticleId`.

See Also:

Articles
Data Categories
KnowledgeArticle
KnowledgeArticleViewStat
KnowledgeArticleVoteStat

KnowledgeArticleVersionHistory

Enables read-only access to the full history of an article. This object is available in API version 25.0 and later.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

This object respects field, entity, and record-level security. You must have at least "Read" permission on the article type or the field to access its history. For data category security, Salesforce determines access based on the categorization of the online version of an article. If there is no online version, then security is applied based on the archived version, followed by the security of the draft version.

Fields

Field Name	Details
`EventType`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The type of event that is tracked in the history table.
`FieldName`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Name of the tracked field.
`Language`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The language that the article is written in, such as `French` or `Chinese (Traditional)`. Querying or searching articles in SOQL or SOSL requires that you specify the `Language` field in the WHERE clause. The language must be the same for all article types.
`NewValue`	**Type** anyType **Properties** Nillable, Sort

Field Name	Details
	Description The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The most recent value of the field before it was changed.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the article.
`ParentSobjectType`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The type of object that contains the field.
`VersionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID assigned to a version of the article.
`VersionNumber`	**Type** int **Properties** Filter, Group, Sort

Field Name	Details
	Description The number assigned to a version of an article. This field is available in API version 24.0 and later.

Usage

Use this object to query events in the history of an article. For example, you can retrieve the number of edits a particular user has made to an article, how many times the article has been published, and so on.

KnowledgeArticleViewStat

Provides statistics on the number of views for the specified article across all article types. This object is read-only and available in API version 20 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Users must have access to the published version of an article to retrieve its views. For more information on published article version, see the `PublishStatus` field in KnowledgeArticleVersion.

Fields

Field Name	Details
`Channel`	**Type** picklist **Properties** Filter, Restricted picklist **Description** The channel where the article is viewed: • `AllChannels` for article views across all channels. • `App` for the internal Salesforce Knowledge application. • `Pkb` for article views in public knowledge base. • `Csp` for Customer Portal. • `Prm` for article view in partner portal.
`NormalizedScore`	**Type** double **Properties** Filter, Nillable

Field Name	Details
	Description Article's weighted views in the selected channel. The article with most views has a score of 100. Other article views are then calculated relative to this highest view score. For example, if the best read article has 2000 views and another has 1000. The first one gets a score of 100 while the second gets 50.
`ParentId`	**Type** reference **Properties** Filter **Description** ID of the viewed article. This corresponds to a KnowledgeArticle record.
`ViewCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of unique views an article has received in the selected channel. An article with a high number of views may not always have a high normalized score. The normalized score for an article is calculated based on views over time, with more recent views earning a higher score. This field is available in API version 27.0 and later.

Usage

Use this object to query or retrieve statistics for article views.

Alternatively, client applications can use the article type `API Name` followed by `__ViewStat` to query or retrieve most viewed articles from a specific article type.

SOQL Samples

The following SOQL clause uses KnowledgeArticleViewStat to query all the article views in Salesforce Knowledge and return the related articles:

```
SELECT Id, NormalizedScore, Parent.Id
FROM KnowledgeArticleViewStat where Channel = 'App'
ORDER BY NormalizedScore
```

Use the following clause to restrict your query to Offer articles for the `Offer` article type:

```
SELECT Id, NormalizedScore, Parent.Id
FROM Offer__ViewStat where Channel = 'App'
ORDER BY NormalizedScore
```

See Also:

Articles
Data Categories
KnowledgeArticle
KnowledgeArticleVersion
KnowledgeArticleVoteStat

KnowledgeArticleVoteStat

Provides the weighted rating for the specified article on a scale of 1 to 5 across all article types. This object is read-only and available in API version 20 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Users must have access to the published version of an article to retrieve its votes. For more information on published article version, see the `PublishStatus` field in KnowledgeArticleVersion

Fields

Field Name	Details
`Channel`	**Type** picklist **Properties** Filter, Restricted picklist **Description** The channel where the article is rated: • `AllChannels` for article views across all channels. • `App` for the internal Salesforce Knowledge application. • `Pkb` for article views in public knowledge base. • `Csp` for Customer Portal. • `Prm` for article view in partner portal.
`NormalizedScore`	**Type** double

Field Name	Details
	Properties Filter, Nillable **Description** Article's weighted score on a scale of 1 to 5. A higher score means more votes. Articles without recent votes trend towards an average rating of three stars.
`ParentId`	**Type** reference **Properties** Filter **Description** The rated article. This corresponds to a KnowledgeArticle record.

Usage

Use this object to query or retrieve the rating for an article.

Alternatively, client applications can use the article type `API Name` followed by `__VoteStat` to query or retrieve the rating for an article for a specific article type.

SOQL Samples

See KnowledgeArticleViewStat.

See Also:

Articles
Data Categories
KnowledgeArticle
KnowledgeArticleVersion
KnowledgeArticleViewStat

Lead

Represents a prospect or potential Opportunity.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `merge()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`Address` (beta)	**Type** address

Field	Details
	Properties Filter, Nillable **Description** The compound form of the address. Read-only. See Address Compound Fields for details on compound address fields.
`AnnualRevenue`	**Type** currency **Properties** Create, Filter, Nillable, Sort, Update **Description** Annual revenue for the company of the lead.
`City`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** City for the address of the lead.
`CleanStatus`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates the record's clean status as compared with Data.com. Values are: `In Sync`, `Different`, `Reviewed`, `Not Found`, `Inactive`, `Not Compared`, `Select Match`, or `Skipped`.
`Company`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Company of the lead. **Note:** If person account record types have been enabled, and if the value of `Company` is null, the lead converts to a person account.

Field	Details
CompanyDunsNumber	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The Data Universal Numbering System (D-U-N-S) number is a unique, nine-digit number assigned to every business location in the Dun&Bradstreet database that has a unique, separate, and distinct operation. D-U-N-S numbers are used by industries and organizations around the world as a global standard for business identification and tracking. Maximum size is 9 characters. **Note:** This field is only available to organizations that use Data.com Prospector or Data.com Clean.
ConnectionReceivedId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the PartnerNetworkConnection that shared this record with your organization. This field is only available if you have enabled Salesforce to Salesforce.
ConnectionSentId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the PartnerNetworkConnection that you shared this record with. This field is only available if you have enabled Salesforce to Salesforce. Beginning with API version 15.0, the `ConnectionSentId` field is no longer supported. The `ConnectionSentId` field is still visible, but the value is null. You can use the new PartnerNetworkRecordConnection object to forward records to connections.
ConvertedAccountId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Object reference ID that points to the Account into which the Lead has been converted.

Field	Details
ConvertedContactId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Object reference ID that points to the Contact into which the Lead has been converted.
ConvertedDate	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Date on which this Lead was converted.
ConvertedOpportunityId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Object reference ID that points to the Opportunity into which the Lead has been converted.
Country	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Country for the address of the lead.
CountryCode	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO country code for the lead's address.
CurrencyIsoCode	**Type** picklist

Field	Details
	Properties Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.
`Description`	**Type** textarea **Properties** Create, Nillable, Update **Description** Description of the lead.
`Division`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** A logical segment of your organization's data. Available only if the organization has the Division permission enabled.
`Email`	**Type** email **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Email address for the lead.
`EmailBouncedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort, Update **Description** If bounce management is activated and an email sent to the lead bounced, the date and time the bounce occurred.
`EmailBouncedReason`	**Type** string

Field	Details
	Properties Filter, Group, Nillable, Sort, Update **Description** If bounce management is activated and an email sent to the lead bounced, the reason the bounce occurred.
`Fax`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Fax number for the lead.
`FirstName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** First name of the lead. Limited to 40 characters.
`HasOptedOutOfEmail`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Update **Description** Indicates whether the lead has opted out of email (`true`) or not (`false`). Label is **Email Opt Out**.
`Industry`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Industry the lead works in.
`IsConverted`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates whether the Lead has been converted (`true`) or not (`false`). Label is **Converted**.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsUnreadByOwner`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** If true, lead has been assigned, but not yet viewed. See Unread Leads for more information. Label is **Unread By Owner**.
`Jigsaw`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** References the ID of a contact in Data.com. If a lead has a value in this field, it means that a contact was imported as a lead from Data.com. If the contact (converted to a lead) was not imported from Data.com, the field value is null. Maximum size is 20 characters. Available in API version 22.0 and later. Label is **Data.com Key**. **Important:** The `Jigsaw`field is exposed in the API to support troubleshooting for import errors and reimporting of corrected data. Do not modify the value in the `Jigsaw` field.
`LastActivityDate`	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Value is one of the following, whichever is the most recent:

Field	Details
	• Due date of the most recent event logged against the record. • Due date of the most recently closed task associated with the record.
`LastName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Last name of the lead. Limited to 80 characters.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`Latitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `Longitude` to specify the precise geolocation of an address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`Longitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update

Field	Details
	Description Used with `Latitude` to specify the precise geolocation of an address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`LeadSource`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Source from which the lead was obtained.
`MasterRecordId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** If this object was deleted as the result of a merge, this field contains the ID of the record that was kept. If this object was deleted for any other reason, or has not been deleted, the value is `null`. **Note:** When using Apex triggers to determine which record was deleted in a merge event, this field's value is the ID of the record that was kept only in `Trigger.old`. In `Trigger.new`, the value is `null`.
`MiddleName` (beta)	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Middle name of the lead. Maximum size is 40 characters. Contact salesforce.com Customer Support to enable this field.
`MobilePhone`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Mobile phone number for the lead.

Field	Details
`Name`	**Type** string **Properties** Filter, Group, Sort **Description** Concatenation of `FirstName`, `MiddleName` (beta), `LastName`, and `Suffix` (beta). Maximum size is 121 characters.
`NumberOfEmployees`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Number of employees at the lead's company. Label is **Employees**.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the owner of the lead.
`PartnerAccountId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the partner account for the partner user that owns this lead. Available only if Partner Relationship Management is enabled OR Communities is enabled and you have partner portal licenses. **Note:** If you are uploading leads using API version 15.0 or earlier, and one of the leads in the batch has a partner user as the owner, the `Partner Account` field on all leads in the batch is set to that partner user's account regardless of whether the partner user is the owner. In version 16.0, the `Partner Account` field is set to the appropriate account for the partner user that owns the lead. If the owner of the lead is not a partner user, this field remains blank.
`Phone`	**Type** phone

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** Phone number for the lead.
`PhotoUrl`	**Type** url **Properties** Filter, Group, Sort **Description** Path to be combined with the URL of a Salesforce instance (for example, https://na1.salesforce.com/) to generate a URL to request the social network profile image associated with the lead. Generated URL returns an HTTP redirect (code 302) to the social network profile image for the lead. Blank if Social Accounts and Contacts isn't enabled for the organization or if Social Accounts and Contacts has been disabled for the requesting user.
`PostalCode`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Postal code for the address of the lead. Label is **Zip/Postal Code**.
`Rating`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Rating of the lead.
`RecordTypeId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the record type assigned to this object.

Field	Details
`Salutation`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Salutation for the lead.
`State`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** State for the address of the lead.
`StateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO state code for the lead's address.
`Status`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Status code for this converted lead. Status codes are defined in `Status` and represented in the API by the LeadStatus object.
`Street`	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Street number and name for the address of the lead.
`Suffix` (beta)	**Type** string

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** Name suffix of the lead. Maximum size is 40 characters. Contact salesforce.com Customer Support to enable this field.
`Title`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Title for the lead, for example CFO or CEO.
`Website`	**Type** url **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Website for the lead.

Note: If you import Lead data and need to set the value for an audit field, such as `CreatedDate`, contact salesforce.com. Audit fields are automatically updated during API operations unless you request to set these fields yourself..

Converted Leads

Leads have a special state to indicate that they have been converted into an Account, Contact, and optionally, an Opportunity. Your client application can convert leads via the `convertLead()` call. Users can also convert leads through the user interface. Once a lead has been converted, it is read-only. You can't update or delete a converted lead. However, you can query converted lead records.

Leads have several fields that indicate their converted status. These special fields are set when converting the lead in the user interface.

- `ConvertedAccountId`
- `ConvertedContactId`
- `ConvertedDate`
- `ConvertedOpportunityId`
- `IsConverted`
- `Status`

Note: If person account record types have been enabled, and if the value of `Company` is null, the lead converts to a person account.

Unread Leads

Leads have a special state to indicate that they have not been viewed or edited by the lead owner. In the user interface, this is helpful for users to know which leads have been assigned to them but which they have not touched yet. `IsUnreadByOwner` is `true` if the lead owner has not yet viewed or edited the lead, and `false` if the lead owner has viewed or edited the lead at least once.

Lead Status Picklist

Each `Status` value corresponds to either a converted or unconverted status in the lead status picklist, as defined in the user interface. To obtain the lead status values in the picklist, a client application can query LeadStatus.

You can't convert a lead via the API by changing `Status` to one of the converted lead status values. When you convert qualified leads into an account, contact, and opportunity, you can select one of the converted status types for the lead. Leads with a converted status type are no longer available in the Leads tab, although you can include them in reports.

Usage

To update a Lead or to convert one with `convertLead()`, your client application must log in with the "Edit" permission on leads.

When you create, update, or upsert a lead, your client application can have the lead automatically assigned to one or more User records based on assignment rules that have been configured in the user interface.

To use this feature, your client application needs to set either of the following options (but not both) in the AssignmentRuleHeader used in create or update:

Field	Field Type	Details
`assignmentRuleId`	reference	ID of the assignment rule to use. Can be an inactive assignment rule. If unspecified and `useDefaultRule` is `true`, then the default assignment rule is used. To find the ID for a given assignment rule, query the AssignmentRule object (specifying `RuleType="leadAssignment"`), iterate through the returned AssignmentRule records, find the one you want to use, retrieve its ID, and then specify its ID in this field in the AssignmentRuleHeader.
`useDefaultRule`	boolean	Specifies whether to use the default rule for rule-based assignment (`true`) or not (`false`). Default rules are assigned in the user interface.

Java Sample

The following Java sample shows how to automatically assign a newly created lead.

```
package com.doc.samples;
import java.net.MalformedURLException;
import java.net.URL;
import java.rmi.RemoteException;
import javax.xml.rpc.ServiceException;

import com.sforce.soap.enterprise.LoginResult;
import com.sforce.soap.enterprise.QueryResult;
import com.sforce.soap.enterprise.SaveResult;
import com.sforce.soap.enterprise.SforceServiceLocator;
import com.sforce.soap.enterprise.SoapBindingStub;
import com.sforce.soap.enterprise._AssignmentRuleHeader;
import com.sforce.soap.enterprise._SessionHeader;
import com.sforce.soap.enterprise.fault.LoginFault;
import com.sforce.soap.enterprise.fault.UnexpectedErrorFault;
```

```
import com.sforce.soap.enterprise.sobject.Lead;
import com.sforce.soap.enterprise.sobject.SObject;

public class LeadAssignment
{

  static LeadAssignment _leadAssignment;

  public static void main(String[] args)
  {
    _leadAssignment = new LeadAssignment();
    try {
      _leadAssignment.CreateLead();
    } catch (Exception e) {
      e.printStackTrace();
    }
  }

  public void CreateLead() throws UnexpectedErrorFault, LoginFault,
    RemoteException, ServiceException
  {
    //Create the proxy binding and login
    SoapBindingStub binding = (SoapBindingStub) new SforceServiceLocator().getSoap();
    LoginResult lr = binding.login("user@domain.net", "secret");

    //Reset the binding to use the endpoint returned from login
    binding._setProperty(SoapBindingStub.ENDPOINT_ADDRESS_PROPERTY,
        lr.getServerUrl());

    //Create the session id header, and add it to the proxy binding
    _SessionHeader sh = new _SessionHeader();
    sh.setSessionId(lr.getSessionId());
    binding.setHeader(
        new SforceServiceLocator().getServiceName().getNamespaceURI(),
        "SessionHeader", sh );

    //Create a new case and assign various properties
    Lead lead = new Lead();

    lead.setFirstName("Joe");
    lead.setLastName("Smith");
    lead.setCompany("ABC Corporation");
    lead.setLeadSource("API");
    //The lead assignment rule will assign any new leads that
    //have "API" as the LeadSource to a particular user

    //Create the assignment rule header and add it to the proxy binding
    _AssignmentRuleHeader arh = new _AssignmentRuleHeader();

    //In this sample we will look for a particular rule and if found
    //use the id for the lead assignment. If it is not found we will
    //instruct the call to use the current default rule. You can't use
    //both of these values together.
    QueryResult qr = binding.query("Select Id From AssignmentRule where Name = " +
       "'Mass Mail Campaign' and RuleType = 'leadAssignment'");
    if (qr.getSize() == 0) {
      arh.setUseDefaultRule(new Boolean(true));
    } else {
      arh.setAssignmentRuleId(qr.getRecords(0).getId());
    }

    binding.setHeader(
        new SforceServiceLocator().getServiceName().getNamespaceURI(),
            "AssignmentRuleHeader", arh);

    // Every operation that results in a new or updated case, will
    // use the specified rule until the header is removed from the
    // proxy binding.
    SaveResult[] sr = binding.create(new SObject[] {lead});
```

```
    for (int i=0;i<sr.length;i++) {
      if (sr[i].isSuccess()) {
        System.out.println("Successfully created lead with id of: " +
          sr[i].getId().getValue() + ".");
      }
      else {
        System.out.println("Error creating lead: " +
          sr[i].getErrors(0).getMessage());
      }
    }

    // This call effectively removes the header, the next lead will
    // be assigned to the default lead owner. Remember to add the
    // session header back in.
    binding.clearHeaders();
    binding.setHeader(
        new SforceServiceLocator().getServiceName().getNamespaceURI(),
            "SessionHeader", sh);

  }
}
```

C# Sample

The following C# sample shows how to automatically assign a newly created lead.

```
using System;
using System.Collections.Generic;
using System.Text;
using LeadAssignment.sforce;

namespace LeadAssignment
{
    class LeadAssignment
    {
        private SforceService binding;

        private static readonly string Username = "ENTERUSERNAME";
        private static readonly string Password = "ENTERPASSWORD";

        /// <summary>
        /// Create the proxy binding and login
        /// </summary>
        private LeadAssignment()
        {
            this.binding = new SforceService();
          LoginResult lr = binding.login(LeadAssignment.Username, LeadAssignment.Password);


            // Reset the binding to use the endpoint returned from login
            this.binding.Url = lr.serverUrl;

            // Create the session ID header and add it to the proxy binding
            this.binding.SessionHeaderValue = new SessionHeader();
            this.binding.SessionHeaderValue.sessionId = lr.sessionId;
        }

        [STAThread]
        static void Main(string[] args)
        {
            LeadAssignment leadAssignment = new LeadAssignment();
            try
            {
                leadAssignment.CreateLead();
            }
            catch (Exception e)
            {
                Console.WriteLine(e.Message);
```

```
                Console.WriteLine(e.StackTrace);
                Console.WriteLine(e.InnerException);
            }
        }

        public void CreateLead()
        {
            // Create a new Lead and assign various properties
            Lead lead = new Lead();

            lead.FirstName = "John";
            lead.LastName = "Brown";
            lead.Company = "ABC Corporation";
            lead.LeadSource = "Advertisement";
            // Setting the lead source for a pre-existing lead assignment rule. This
            // rule was created outside of this sample and will assign any new leads
            // that have "Advertisement" as the LeadSource to a particular user

            // Create the assignment rule header and add it to the proxy binding
            AssignmentRuleHeader arh = new AssignmentRuleHeader();

            // In this sample we will look for a particular rule and if found
            // use the id for the lead assignment. If it is not found we will
            // instruct the call to use the current default rule. Both these
            // values can't be used together.
            QueryResult qr = binding.query("Select Id from AssignmentRule where Name = "
+
                "'Mass Mail Campaign' and SobjectType = 'lead'");
            if (qr.size == 0)
            {
                arh.useDefaultRule = true;
            }
            else
            {
                arh.assignmentRuleId = qr.records[0].Id;
            }
            binding.AssignmentRuleHeaderValue = arh;

            // Every operation that results in a new or updated lead will use the
            // specified rule until the header is removed from the proxy binding
            SaveResult[] sr = binding.create(new sObject[] { lead });
            foreach (SaveResult s in sr)
            {
                if (s.success)
                {
                    Console.WriteLine("Successfully created Lead with ID: {0}", s.id);
                }
                else
                {
                    Console.WriteLine("Error creating Lead: {0}", s.errors[0].message);
                }
            }

            // This call effectively removes the header. The next lead will be assigned
            // to the default lead owner.
            binding.AssignmentRuleHeaderValue = null;
        }
    }
}
```

See Also:

LeadOwnerSharingRule
LeadShare
LeadStatus
PartnerNetworkConnection

LeadCleanInfo

Stores the metadata Data.com Clean uses to determine a lead record's clean status. Helps you automate the cleaning or related processing of lead records. LeadCleanInfo includes a number of bit vector fields.

- `CleanedBy` indicates who (a user) or what (a Clean job) cleaned the lead record.
- `IsDifferent` indicates whether or not a field on the lead record has a value that differs from the corresponding field on the matched Data.com record.
- `IsFlaggedWrong` indicates whether or not a field on the lead record has a value that is flagged as wrong to Data.com.
- `IsReviewed` indicates whether or not a field on the lead record is in a `Reviewed` state, which means that the value was reviewed by a user but not accepted.

Their individual bits are defined here

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`

Fields

Field Name	Details
`Address` (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the address. Read-only. See Address Compound Fields for details on compound address fields.
`AnnualRevenue`	**Type** currency **Properties** Filter, Nillable, Sort **Description** Estimated annual revenue of the lead.
`City`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the lead.

Field Name	Details
CleanedByJob	**Type** boolean **Properties** Filter **Description** Indicates whether the lead record was cleaned by a Data.com Clean job (`true`) or not (`false`).
CleanedByUser	**Type** boolean **Properties** Filter **Description** Indicates whether the lead record was cleaned by a Salesforce user (`true`) or not (`false`).
CompanyDunsNumber	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The Data Universal Numbering System (D-U-N-S) number is a unique, nine-digit number assigned to every business location in the Dun & Bradstreet database that has a unique, separate, and distinct operation. D-U-N-S numbers are used by industries and organizations around the world as a global standard for business identification and tracking.
CompanyName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The name of the company.
ContactStatusDataDotCom	**Type** picklist **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description The status of the contact associated with the lead per Data.com. Values are: `Contact is Active per Data.com`, `Phone is Wrong per Data.com` , `Email is Wrong per Data.com`, `Phone and Email are Wrong per Data.com`, `Contact Not at Company per Data.com`, `Contact is Inactive per Data.com`, `Company this contact belongs to is out of business per Data.com`, `Company this contact belongs to never existed per Data.com` or `Email address is invalid per Data.com`.
`Country`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the lead.
`DandBCompanyDunsNumber`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The D-U-N-S Number on the D&B Company record (if any) that is linked to the lead.
`DataDotComCompanyId`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The ID Data.com maintains for the company associated with the lead.
`DataDotComId`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The ID Data.com maintains for the contact associated with the lead.

Field Name	Details
`Email`	**Type** email **Properties** Filter, Group, Nillable, Sort **Description** The lead’s email address.
`FirstName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The lead’s first name.
`Industry`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** The industry the lead belongs to.
`IsDifferentAnnualRevenue`	**Type** boolean **Properties** Filter **Description** Indicates whether the lead’s `AnnualRevenue` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentCity`	**Type** boolean **Properties** Filter **Description** Indicates whether the lead’s `City` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).

Field Name	Details
IsDifferentCompanyDunsNumber	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `Company D-U-N-S Number` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentCompanyName	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `Company Name` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentCountry	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `Country` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentCountryCode	**Type** boolean **Properties** Filter **Description** Indicates whether the account's `Country Code` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentDandBCompanyDunsNumber	**Type** boolean **Properties** Filter

Field Name	Details
	Description Indicates whether the lead's `D&B Company D-U-N-S Number` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentEmail`	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `Email` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentFirstName`	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `First Name` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentIndustry`	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `Industry` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
`IsDifferentLastName`	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `Last Name` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).

Field Name	Details
IsDifferentNumberOfEmployees	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `No. of Employees` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentPhone	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `Phone` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentPostalCode	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `Postal Code` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentState	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `State` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentStateCode	**Type** boolean **Properties** Filter

Field Name	Details
	Description Indicates whether the account's `State Code` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentStreet	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `Street` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsDifferentTitle	**Type** boolean **Properties** Filter **Description** Indicates whether the lead's `Title` field value is different from the corresponding value on its matched Data.com record (`true`) or not (`false`).
IsFlaggedWrongAddress	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Address` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongAnnualRevenue	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Annual Revenue` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongCompanyDunsNumber	**Type** boolean

Field Name	Details
	Properties Filter, Update **Description** Indicates whether the lead's `Company D-U-N-S Number` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongCompanyName	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Company Name` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongEmail	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Email` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongIndustry	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Industry` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongName	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Name` field value is flagged as wrong to Data.com (`true`) or not (`false`).
IsFlaggedWrongNumberOfEmployees	**Type** boolean

Field Name	Details
	Properties Filter, Update **Description** Indicates whether the lead's `No. of Employees` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongPhone`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Phone` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsFlaggedWrongTitle`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Title` field value is flagged as wrong to Data.com (`true`) or not (`false`).
`IsInactive`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the lead has been reported to Data.com as `Inactive` (`true`) or not (`false`).
`IsReviewedAddress`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Address` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedAnnualRevenue`	**Type** boolean

Field Name	Details
	Properties Filter, Update **Description** Indicates whether the lead's `Annual Revenue` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedCompanyDunsNumber`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Company D-U-N-S Number` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedCompanyName`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Company Name` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedDandBCompanyDunsNumber`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `D&B Company D-U-N-S Number` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedEmail`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Email` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedIndustry`	**Type** boolean

Field Name	Details
	Properties Filter, Update **Description** Indicates whether the lead's `Industry` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedName`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Name` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedNumberOfEmployees`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `No. of Employees` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedPhone`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Phone` field value is in a `Reviewed` state (`true`) or not (`false`).
`IsReviewedTitle`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether the lead's `Title` field value is in a `Reviewed` state (`true`) or not (`false`).
`LastMatchedDate`	**Type** dateTime

Field Name	Details
	Properties Filter, Sort **Description** The date the lead record was last matched and linked to a Data.com record.
`LastName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The lead’s last name.
`LastStatusChangedById`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of who or what last changed the record’s `Clean Status` field value: a Salesforce user or a Clean job.
`LastStatusChangedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date on which the record’s `Clean Status` field value was last changed.
`Latitude` (beta)	**Type** double **Properties** Filter, Nillable, Sort **Description** Used with `Longitude` to specify the precise geolocation of a billing address. Data not currently provided.
`LeadId`	**Type** reference

Field Name	Details
	Properties Filter, Group, Sort **Description** The unique, system-generated ID assigned when the lead record was created.
`Longitude` (beta)	**Type** double **Properties** Filter, Nillable, Sort **Description** Used with `Latitude` to specify the precise geolocation of a billing address. Data not currently provided.
`Name`	**Type** string **Properties** Filter, Group, Sort, Update **Description** Field label is **Lead Clean Info Name**. The name of the lead. Maximum size is 255 characters.
`NumberOfEmployees`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of employees working at the lead.
`Phone`	**Type** phone **Properties** Filter, Group, Nillable, Sort **Description** The phone number for the lead.
`PostalCode`	**Type** string

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** Details for the billing address of the lead.
`State`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the lead.
`Street`	**Type** textarea **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address of the lead.
`Title`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The lead's title.

Usage

Use triggers to map values from fields on imported or cleaned records onto a standard set of values.

LeadFeed

Represents a single feed item in the feed displayed on the detail page for a lead record.

A lead feed shows recent changes to a lead record for any fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to leads in Salesforce. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the Lead object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
Body	**Type** textarea **Properties** Nillable, Sort **Description** The content of LeadFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
CommentCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
ConnectionId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
ContentData	**Type** base64 **Properties** Nillable

Field	Details
	Description The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime

Field	Details
	Properties Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort

Field	Details
	Description Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`.

Field	Details
	• For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the lead record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence.

Field	Details
	• `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field	Details
	Description Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a lead record. If a lead record is converted, use this object to query and retrieve the associated lead feed items.

See Also:

Lead

EntitySubscription

NewsFeed

UserProfileFeed

LeadHistory

Represents the history of changes to the values in the fields of a lead.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort,

Field	Details
	Description The name of the field that was changed.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`LeadId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Lead. Label is **Lead ID**.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.

Usage

Use this object to identify changes to a lead.

This object respects field level security on the parent object.

See Also:

LeadShare
LeadStatus

LeadOwnerSharingRule

Represents the rules for sharing a lead with users other than the owner.

Note: Contact salesforce.com customer support to enable access to this object for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
`DeveloperName`	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Rule Name** in the user interface. This field is available in API version 24.0 and later. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`GroupId`	**Type** reference

Field	Details
	Properties Create, Filter, Group, Sort **Description** The ID representing the source group. Leads owned by users in the source group trigger the rule to give access.
`LeadAccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of sharing being allowed. The possible values are: • `Read` • `Edit`
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to **Label** on the user interface.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the target user or group. The target user or group is being given access.

Usage

Use these objects to manage the sharing rules for leads. General sharing and Territory-related sharing use this object.

See Also:

Lead
LeadShare
LeadStatus

LeadShare

Represents a sharing entry on a Lead.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
LeadAccessLevel	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Level of access that the User or Group has to the Lead. The possible values are: • `Read` • `Edit` • `All` This value is not valid when creating or updating these records. This field must be set to an access level that is higher than the organization's default access level for leads.
LeadId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Lead associated with this sharing entry. This field can't be updated.
RowCause	**Type** picklist

Field	Details
	Properties Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Values may include: • `Manual Sharing`—The User or Group has access because a user with "All" access manually shared the Lead with them. • `Owner`—The User is the owner of the Lead or is in a Role above the Case owner in the role hierarchy. • `ImplicitChild`—User or Group has access to the Lead on the Account associated with this Lead. • `Sharing Rule`—The User or Group has access via a Lead sharing rule.
`UserOrGroupId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the User or Group that has been given access to the Lead. This field can't be updated.

Usage

This object allows you to determine which users and groups can view or edit leads owned by other users.

If you attempt to create a record that matches an existing record, the existing record is returned.

See Also:

AccountShare
Case
CaseShare
OpportunityShare

LeadStatus

Represents the status of a Lead, such as Open, Qualified, or Converted.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
IsConverted	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this lead status value represents a converted lead (`true`) or not (`false`). Multiple lead status values can represent a converted lead.
IsDefault	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this is the default lead status value (`true`) or not (`false`) in the picklist.
MasterLabel	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Master label for this lead status value. This display value is the internal label that does not get translated.
SortOrder	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Number used to sort this value in the lead status picklist. These numbers are not guaranteed to be sequential, as some previous lead status values might have been deleted.

Usage

This object represents a value in the lead status picklist (see Lead Status Picklist). The lead status picklist provides additional information about the status of a Lead, such as whether a given status value represents a converted Lead. Query this object to retrieve the set of values in the lead status picklist, and then use that information while processing Lead objects to determine more information about a given lead. For example, the application could test whether a given lead is converted based on its `Status` value and the value of the `IsConverted` property in the associated LeadStatus record.

See Also:

LeadOwnerSharingRule
LeadShare

LeadTag

Associates a word or short phrase with a Lead.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.

Field Name	Details
Type	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

LeadTag stores the relationship between its parent TagDefinition and the Lead being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

LimitAllocationPerApp

Represents a connected app quota for an API limit. This object is available in API version 30.0 and later.

Note: LimitAllocationPerApp is currently available through a pilot program. For information on enabling LimitAllocationPerApp for your organization, contact salesforce.com.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
ConnectedAppName	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The name of the connected app.
DeveloperName	**Type** string

Field Name	Details
	Properties Create, Filter, Group, Sort, Update **Description** The unique name of the record in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. This field is automatically generated but you can supply your own value if you create the record using the API. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`Language`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The language of the `MasterLabel`.
`LimitType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The type of API limit for which you're setting a quota. The available values are: • TotalRequests—Total API requests per 24-hour period • ApiBatchItems—Number of Bulk API batches per 24-hour period • StreamEventsPerDay—Number of Streaming API events per 24–hour period • GenStreamingEventsPerDay—Number of generic streaming events per 24–hour period **Note:** Generic streaming is currently available through a pilot program. For information on enabling generic streaming for your organization, contact salesforce.com.

Field Name	Details
MasterLabel	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Master label for the API limit quota. This is the internal label that does not get translated.
Percentage	**Type** int **Properties** Create, Filter, Group, Sort, Update **Description** The percentage of the API limit reserved for the connected app. The app can't consume more than this percentage of the API limit. You can find the available percentage for each limit type through the user interface or by using the `query()` call.

Usage

App quotas for API limits enable you to reserve API capacity for mission-critical connected apps or set a ceiling for API usage of non-critical connected apps.

LineitemOverride

A forecast override of a line item on an Opportunity. This read-only object for customizable forecasting has a child-parent relationship with OpportunityOverride.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

This object is only accessible if your organization has enabled the customizable forecasting feature, which can be done in the user interface. Requires the "View All Data" permission.

Fields

Field	Details
AmountInherited	**Type** boolean

Field	Details
	Properties Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the overridden amount rolls up through the forecast hierarchy (`true`), or was overridden by the owner of this record (`false`).
`ForecastCategoryInherited`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the overridden forecast category rolls up through the forecast hierarchy (`true`), or was overridden by the owner of this record (`false`).
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`OpportunityId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the associated Opportunity.
`OpportunityLineItemId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the associated OpportunityLineItem.
`OverrideAmount`	**Type** currency

Field	Details
	Properties Create, Filter, Sort, Update **Description** The total monetary amount of the line item, which may be overridden.
`OverrideForecastCategory`	**Type** picklist **Properties** Create, Filter, Group, Sort, Update **Description** The forecast category of the line item. Can be overridden.
`OverrideQuantity`	**Type** double **Properties** Create, Filter, Sort, Update **Description** The quantity of the line item, which may be overridden.
`OverrideUnitPrice`	**Type** currency **Properties** Create, Filter, Sort, Update **Description** The monetary amount of the unit price. Can be overridden.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** ID of the owner of this record.
`QuantityInherited`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field	Details
	Description Indicates whether the overridden quantity rolls up through the forecast hierarchy (`true`), or is overridden by the owner of this record (`false`).
`UnitPriceInherited`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the overridden unit price rolls up through the forecast hierarchy (`true`), or was overridden by the owner of this record (`false`).

See Also:

OpportunityOverride

LiveAgentSession

This object is automatically created for each Live Agent session and stores information about the session. This object is available in API versions 28.0 and later.

Note: Standard fields for the LiveAgentSession object can only be modified if your administrator has given you editing permissions for these records.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`

Fields

Field Name	Details
`AgentId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the agent associated with the session.
`ChatReqAssigned`	**Type** int

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The number of chat requests that were assigned to an agent during a session.
`ChatReqDeclined`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The number of chat requests that were declined by an agent during a session.
`ChatReqEngaged`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The number of chats in which an agent was engaged during a session.
`ChatReqTimedOut`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The number of chat requests that timed out in an agent's queue during a session.
`LastReferencedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date and time that the session record was last referenced.
`LastViewedDate`	**Type** dateTime

Field Name	Details
	Properties Filter, Nillable, Sort **Description** The date and time that the session record was last viewed.
`LoginTime`	**Type** dateTime **Properties** Create, Filter, Sort, Update **Description** The date and time an agent logged in during the session.
`LogoutTime`	**Type** dateTime **Properties** Create, Filter, Sort, Update **Description** The date and time an agent logged out during a session.
`Name`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookupSort **Description** The name of the session.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the owner of the session record.
`TimeAtCapacity`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description The amount of time an agent spent with the maximum number of chats in his or her queue.
`TimeIdle`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The amount of time an agent spent idle during the session.
`TimeInAwayStatus`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The amount of time an agent spent with a status of "Away" during a session.
`TimeInChats`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The amount of time an agent spent engaged in chats during a session.
`TimeInOnlineStatus`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The amount of time an agent spent with a status of "Online" during a session.

Usage

Use this object to query and manage live chat session records.

LiveAgentSessionHistory

This object is automatically created for each Live Agent session and stores information about changes made to the session. This object is available in API versions 28.0 and later.

Note: Standard fields for the LiveAgentSession object can only be modified if your administrator has given you editing permissions for these records.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed in a session record.
`LiveAgentSessionId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the session record that was changed.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort

Field Name	Details
	Description The original value of the field that was changed.

Usage

Use this object to identify changes to live chat session records.

LiveAgentSessionOwnerSharingRule

Represents the rules for sharing a Live Agent session record with users other than the record owner.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, restrictedPicklist, Sort, Update **Description** Determines the level of access users have to session records. Specifies whether or not users can read, edit, or transfer session records. Corresponds to the Default Access column in the UI.
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your

Field	Details
	organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to Rule Name in the UI. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`GroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the source group. Session records owned by users in the source group trigger the rule to give access.
`Name`	**Type** string **Properties** Create, Filter, Group, idLookup,Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to Label in the UI.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the target user or group. The target user or group is being given access.

Usage

Use this object to manage the sharing rules for Live Agent session records.

LiveAgentSessionShare

This object is automatically created for each Live Agent session and stores information about the session. This object is available in API versions 28.0 and later.

Note: Standard fields for the LiveAgentSession object can only be modified if your administrator has given you editing permissions for these records.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Level of access that the User or Group has to the LiveAgentSession. The possible values are: • `Read` • `Edit` • `All` (This value is not valid for `create()` or `update()` calls.) This value must be set to an access level that is higher than the organization's default access level for live chat transcripts.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the parent object, if any.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Values can include: • `Manual`—The user or group has access because a user with "All" access manually shared the LiveAgentSession with them. • `Owner`—The user is the owner of the LiveAgentSession or is in a role above the QuickText owner in the role hierarchy.

Field Name	Details
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the user or group that has been given access to the LiveAgentSession.

Usage

This object lets you determine which users and groups can view and edit LiveAgentSession records owned by other users.

If you attempt to create a new record that matches an existing record, the `create()` call updates any modified fields and returns the existing record.

LiveChatButton

Represents a button that allows visitors to request chats with Live Agent users. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`Animation`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The type of animation used when an automated chat invitation appears on-screen. For automated chat invitations only. Available in API version 29.0 and later.
`AutoGreeting`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** The text that is automatically sent from an agent to a visitor when a chat session starts.

Field Name	Details
	Note: A greeting message in the `AutoGreeting` field of the LiveChatButton object overrides individual users' greeting messages in the `AutoGreeting` field in the LiveChatUserConfig object.
`ChatPageId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The record ID of the custom Visualforce page that contains the custom chat window code.
`CustomAgentName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The custom name of the agent associated with the button. Available in API version 29.0 and later. **Note:** A custom agent name in the `CustomAgentName` field of the LiveChatButton object overrides individual users' custom agent name in the `CustomAgentName` field in the LiveChatUserConfig object.
`CustomRoutingClassId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the Apex class you customize to create your custom routing rules if `RoutingType` equals `Custom`. Available in API version 29.0 and later.
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Sort, Update

Field Name	Details
	Description The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`HasQueue`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether to allow incoming chat requests to queue until an agent is available.
`InviteEndPosition`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The position on-screen where an automated chat invitation's animation ends. **Note:** You don't need to select an end position for your automated chat invitation if you use a custom animation. For automated chat invitations only. Available in API version 29.0 and later.
`InviteImageId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description The record ID of the static image resource displayed on your automated chat invitation. For automated chat invitations only. Available in API version 29.0 and later.
`InviteStartPosition`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The position on-screen where an automated chat invitation's animation begins. **Note:** You don't need to select a start position for your automated chat invitation if you use a custom animation. For automated chat invitations only. Available in API version 29.0 and later.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** For automated chat invitations, specifies whether an automated chat invitation is active and able to be sent to customers (`true`) or not (`false`). For chat buttons, this is set to `true` by default.
`Language`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The language of the chat.
`MasterLabel`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label for the live chat button.

Field Name	Details
`NumberOfReroutingAttempts`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Specifies the number of times a chat request can be rerouted to available agents if all agents reject the chat request.
`OfflineImageId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The record ID of the static image resource that is displayed when the button is offline (inactive).
`OnlineImageId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The record ID of the static image resource that is displayed when the button is online (active).
`OptionsHasInviteAfterAccept`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether an automated chat invitation can be sent to a customer after that customer has accepted a prior automated chat invitation (`true`) or not (`false`). For automated chat invitations only. Available in API version 29.0 and later.
`OptionsHasInviteAfterReject`	**Type** boolean **Properties** Create, Filter, Update

Field Name	Details
	Description Specifies whether an automated chat invitation can be sent to a customer after that customer has rejected a prior automated chat invitation (`true`) or not (`false`). For automated chat invitations only. Available in API version 29.0 and later.
`OptionsHasRerouteDeclinedRequest`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether a chat request that has been rejected by all available agents should be rerouted to available agents again (`true`) or not `false`).
`OptionsIsAutoAccept`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether a chat request should be automatically accepted by the agent it's assigned to (`true`) or not `false`). For chat buttons and automated chat invitations with `RoutingType` set to `Most Available` or `Least Active`. Available in API version 30.0 and later.
`OptionsIsInviteAutoRemove`	**Type** boolean **Properties** Create, Filter, Update **Description** Specifies whether an automated chat invitation should be automatically removed from the screen after a certain amount of time (`true`) or not (`false`). For automated chat invitations only. Available in API version 29.0 and later.
`OverallQueueLength`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The maximum number of chat requests allowed to queue.

Field Name	Details
`PerAgentQueueLength`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The maximum number of chat requests allowed to queue for each agent with the required skill.
`PostchatPageId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The record ID of the custom Visualforce page displayed when the chat ends.
`PostchatUrl`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The URL the user is directed to after the chat ends.
`PrechatFormPageId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The record ID of the custom Visualforce page displayed before the chat begins.
`PrechatFormUrl`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The URL the user is directed to before the chat begins.

Field Name	Details
PushTimeout	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The number of seconds an agent has to answer a chat request before it's routed to the next available agent.
RoutingType	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** How chat requests are routed to agents. The values are: • `Choice`—Incoming chat requests are added to the queue in the Live Agent console or Live Agent in the Salesforce console and are available to any agent with the required skill. • `Least Active`—Incoming chats are routed to the agent with the required skill who has the fewest active chats. • `Most Available`—Incoming chats are routed to the agent with the required skill and the greatest difference between chat capacity and active chat sessions. For example, if Agents A and B each have a chat capacity of five, and Agent A has three active chat sessions while Agent B has only one, incoming chats will be routed to Agent B. • `Custom`—Incoming chats are routed to agents based on the custom routing rules specified in the Apex class you use to create them.
SiteId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The record ID of the site used for loading static resources and custom Visualforce pages.
SkillId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description The record ID of the skill used to route incoming chat requests. To associate multiple skills with a live chat button, reference one skill in the `SkillId` field and use LiveChatButtonSkill junction objects for the remaining skills.
`TimeToRemoveInvite`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The number of seconds an automated invitation stays on-screen before it is automatically removed. For automated chat invitations only. Available in API version 29.0 and later.
`Type`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The type of element to display to customers (either a chat button or an automated invitation).
`WindowLanguage`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The language used for standard chat windows. Custom chat windows use the language of the user's browser.

Usage

Use this object to query and manage live chat buttons and automated chat invitations.

LiveChatButtonDeployment

Associates a Live Agent automated chat invitation with a specific deployment. This object is available in API versions 28.0 and later.

Supported Calls

`create()`, `delete()` `query()`, `retrieve()`, `update()`

Fields

Field Name	Details
ButtonId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the automated invitation associated with the deployment.
DeploymentId	**Type** reference **Properties** Create, Filter, Group, NillableSort **Description** The ID of the deployment that will feature the automated invitation.

Usage

Use this object to associate automated chat invitations with specific deployments.

LiveChatButtonSkill

Represents a join between LiveChatButton and Skill.

Supported Calls

`create()`, `delete()`, `query()`, `update()`

Fields

Field Name	Details
ButtonID	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The record ID of the button.
SkillID	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort **Description** The record ID of the skill.

Usage

Use this object to assign a specific skill to a specific button for multi-skill routing.

LiveChatDeployment

Represents the general settings for deploying Live Agent on a website. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`BrandingId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The record ID of the static image resource that's displayed in the chat window
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is

Field Name	Details
	specified, performance may slow while Salesforce generates one for each record.
`Domains`	**Type** textarea **Properties** Create, Filter (unavailable in API version 25.0 and later), Nillable, Sort (unavailable in API version 25.0 and later), Update **Description** A comma-separated list of domains the deployment is whitelisted for. Leave this blank to allow the deployment to be used on any domain.
`HasTranscriptSave`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether visitors can download and save transcripts from the chat window
`Language`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The language of the deployment
`MasterLabel`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The name of the deployment
`MobileBrandingId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description The record ID of the static image resource displayed in the mobile version of the chat window
`OptionsHasPrechatApi`	**Type** boolean **Properties** Create, Filter, Update **Description** Determines whether developers can access the Pre-Chat API
`SiteId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The record ID of the site used for loading static resources
`WindowTitle`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The text displayed in the title bar of the browser window used to launch the chat window

Usage

Use this object to query and manage live chat deployments.

LiveChatTranscript

This object is automatically created for each Live Agent chat session and stores information about the session. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
`Abandoned`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The amount of time in seconds an incoming chat request remained unanswered by an agent before the chat was disconnected by the customer.
`AccountId`	**Type** ID **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the account associated with the chat transcript.
`AverageResponseTimeOperator`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The agent's average response time (in seconds) to chat messages from the visitor.
`AverageResponseTimeVisitor`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The visitor's average response time (in seconds) to chat messages from the agent.
`Body`	**Type** textarea **Properties** Create, Nillable, Update **Description** The contents of the chat.

Field Name	Details
`Browser`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The browser the visitor used for the chat.
`BrowserLanguage`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The language of the visitor's browser.
`CaseID`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the case associated with the chat transcript.
`ChatDuration`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The total duration of the chat in seconds.
`ChatKey`	**Type** string **Properties** Create, Filter, Group, idLookup, Nillable, Sort **Description** The session ID of the chat before it is persisted. `ChatKey` can be used with advanced integrations in the Salesforce console. This field is available in API version 25.0 and later.
`ContactID`	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the contact associated with the chat transcript.
EndedBy	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The way the chat was ended: by the operator, the visitor, or the system.
EndTime	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** The time the chat ended.
IpAddress	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** The visitor's IP address.
LastReferencedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
LastViewedDate	**Type** date **Properties** Filter, Nillable, Sort, Update

Field Name	Details
	Description The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`LeadID`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the lead associated with the chat transcript.
`LiveChatButtonID`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the LiveChatButton the chat session originated from.
`LiveChatDeploymentID`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Type** The ID of the LiveChatDeployment the chat session originated from.
`LiveChatVisitorID`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the visitor associated with the chat transcript.
`Location`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The best-guess approximation of the visitor's location.

Field Name	Details
MaxResponseTimeOperator	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The maximum time in seconds it took an agent to respond to a chat visitor's message.
MaxResponseTimeVisitor	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The maximum time in seconds it took a customer to respond to an agent's message.
Name	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** The name of the transcript.
OperatorMessageCount	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The number of messages sent by agent(s) during the chat.
OwnerID	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The ID of the operator who participated in the chat last; for missed chats, this is a system user.

Field Name	Details
Platform	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The visitor's operating system platform.
ReferrerUri	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Where the chat request originated.
RequestTime	**Type** dateTime **Properties** Create, Filter, Nillable. Sort, Update **Description** The time the visitor requested the chat.
ScreenResolution	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The visitor's screen resolution.
SkillId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The record ID of the primary Skill associated with the LiveChatButton the chat session originated from. To associate multiple skills with a LiveChatTranscript, use LiveChatTranscriptSkill junction objects.
StartTime	**Type** dateTime

Field Name	Details
	Properties Create, Filter, Nillable, Sort, Update **Description** The time the chat started.
Status	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The final status of the chat: completed or missed.
SupervisorTranscriptBody	**Type** textarea **Properties** Create, Nillable, Update **Description** The text body of the supervisor's chat transcript.
UserAgent	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The visitor's user agent string.
VisitorMessageCount	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The number of messages sent by the visitor during the chat.
WaitTime	**Type** int **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description The total amount of time in seconds a chat request was waiting to be accepted by an agent.

Usage

Use this object to query and manage live chat transcripts.

LiveChatTranscriptEvent

Captures specific events that occur over the lifetime of a chat. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
`AgentId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** The ID of the agent associated with the event.
`Detail`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort **Description** Details associated with the event.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.

Field Name	Details
LastViewedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (LastReferencedDate) and not viewed.
LiveChatTranscriptId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the live chat transcript associated with the event.
Name	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** The name of the event.
Time	**Type** dateTime **Properties** Create, Filter, Sort **Description** The time at which the event happened.
Type	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** The kind of event that occurred.

Usage

Use this object to query and manage live chat transcript events.

LiveChatTranscriptHistory

Represents changes to field values on a LiveChatTranscript object. This object is available in API version 24.0 and later.

Supported Calls

`getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed
`LiveChatTranscriptID`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the transcript that was changed
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The old value of the field that was changed

Usage

Use this object to identify changes to a live chat transcript.

LiveChatTranscriptOwnerSharingRule

Represents the rules for sharing a Live Agent chat transcript record with users other than the record owner.

Supported Calls

`create()`, `delete()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`

Fields

Field	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, restrictedPicklist, Sort, Update **Description** Determines the level of access users have to chat transcript records. Specifies whether or not users can read, edit, or transfer chat transcript records. Corresponds to the Default Access column in the UI.
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to Rule Name in the UI.

Field	Details
	Note: When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`GroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the source group. Chat transcript records owned by users in the source group trigger the rule to give access.
`Name`	**Type** string **Properties** Create, Filter, Group, idLookup,Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to Label in the UI.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the target user or group. The target user or group is being given access.

Usage

Use this object to manage the sharing rules for Live Agent chat transcript records.

LiveChatTranscriptShare

Represents a sharing entry on a LiveChatTranscript object. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `query()`, `retrieve()` `update()`, `upsert()`

Fields

Field Name	Details
AccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Level of access that the User or Group has to the LiveChatTranscript. The possible values are: • Read • Edit • All (This value is not valid for create() or update() calls.) This value must be set to an access level that is higher than the organization's default access level for live chat transcripts.
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the parent object, if any
RowCause	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Values may include: • Manual—The User or Group has access because a user with "All" access manually shared the LiveChatTranscript with them. • Owner—The User is the owner of the LiveChatTranscript or is in a role above the QuickText owner in the role hierarchy.
UserOrGroupID	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the User or Group that has been given access to the LiveChatTranscript.

Usage

This object lets you determine which users and groups can view and edit LiveChatTranscript records owned by other users.

If you attempt to create a new record that matches an existing record, the `create()` call updates any modified fields and returns the existing record.

LiveChatTranscriptSkill

Represents a join between LiveChatTranscript and Skill.

Supported Calls

`create()`, `delete()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`

Fields

Field Name	Details
`Name`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** The name of the transcript.
`SkillID`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** The record ID of the skill.
`TranscriptID`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The record ID of the transcript.

Usage

Use this object to assign a specific skill to a specific transcript for multi-skill routing.

LiveChatUserConfig

Represents a setting that controls the console settings for Live Agent users. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AutoGreeting`	**Type** textarea **Properties** Create, Nillable, Update **Description** The text that is automatically sent from an agent to a visitor when a chat session starts.
`Capacity`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Limits the number of active chat session an agent can engage in.
`CriticalWaitTime`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The amount of time before a chat flashes to alert an agent to answer it.
`CustomAgentName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The custom name of the agent associated with the Live Agent configuration.

Field Name	Details
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`HasCrmChatlet`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether the CRM chatlet appears in the agent console.
`HasDetailsChatlet`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether the Details chatlet appears in the agent console.
`HasHistoryChatlet`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether the History chatlet appears in the agent console.
`HasKnowledgeChatlet`	**Type** boolean

Field Name	Details
	Properties Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether the Knowledge chatlet appears in the agent console.
HasLivetrackChatlet	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether the Livetrack chatlet appears in the agent console.
HasLogoutSound	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether a sound plays when an agent logs out of the console.
HasMapChatlet	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether the Map chatlet appears in the agent console.
HasNotifications	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether desktop notifications are enabled for the configuration. Available in API version 25.0 and later.
HasQueueChatlet	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field Name	Details
	Description Determines whether the Queue chatlet appears in the agent console.
`HasRequestSound`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether a sound plays when a chat request comes in.
`HasSneakPeek`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether an agent sees a real-time preview of the messages a visitor types.
`IsAutoAwayOnDecline`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Determines whether agents' status is automatically changed to Away when they decline a chat request. Available in API version 26.0 and later.
`Language`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The language of the configuration.
`MasterLabel`	**Type** string **Properties** Create, Filter, Group, Sort, Update

Field Name	Details
	Description The name of the configuration.
OptionsHasAgentSneakPeek	**Type** boolean **Properties** Create, Filter, Update **Description** Determines whether Sneak Peek is enabled for agents. Available in API version 29.0 and later.
OptionsHasChatMonitoring	**Type** boolean **Properties** Create, Filter, Update **Description** Determines whether supervisors can view agents' ongoing chats. Available in API version 29.0 and later.
OptionsHasChatTransfer	**Type** boolean **Properties** Create, Filter, Update **Description** Determines whether an agent can transfer a chat to other agents. Available in API version 31.0 and later.
OptionsHasWhisperMessage	**Type** boolean **Properties** Create, Filter, Update **Description** Determines whether supervisors can send private messages to agents within an agent's chat with a customer. Available in API version 29.0 and later.
SupervisorDefaultAgentStatus	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update

Field Name	Details
	Description The default agent status by which to filter agents in the Agent Status list in the supervisor panel.
SupervisorDefaultButtonId	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The default button ID by which to filter agents in the Agent Status list in the supervisor panel.
SupervisorDefaultSkillId	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The default skill ID by which to filter agents in the Agent Status list in the supervisor panel.

Usage

Use this object to query and manage agent configurations in Live Agent.

LiveChatUserConfigProfile

Represents a join between LiveChatUserConfig and Profile. This object is available in API version 24.0 and later.

Supported Calls

`create()` `delete()` `query()` `retrieve()` `update()` `upsert()`

Fields

Field Name	Details
LiveChatUserConfigId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The record ID of the agent configuration

Field Name	Details
`ProfileId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The record ID of the profile

Usage

Use this object to assign specific agent configurations to specific user profiles.

LiveChatUserConfigUser

Represents a join between LiveChatUserConfig and User. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`LiveChatUserConfigId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The record ID of the agent configuration
`UserId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** The record ID of the user

Usage

Use this object to assign specific agent configurations to specific users.

LiveChatVisitor

Represents a website visitor who has started or tried to start a chat session. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`Name`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, Sort **Description** The name of the visitor
`SessionKey`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description The session key used to uniquely identify the visitor

Usage

Use this object to query and manage live chat visitors.

LoginIp

Represents a validated IP address. This object is available in version 28.0 and later.

Supported Calls

`describeSObjects()`, `delete()`, `query()`, `retrieve()`

Fields

Field	Details
`ChallengeSentDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date and time that the user was authenticated.
`IsAuthenticated`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** If `true`, the user has already been authenticated.
`SourceIp`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The IP address the user logged in from.
`UsersId`	**Type** reference

Field	Details
	Properties Filter, Group, Sort **Description** The ID of the user associated with this item.

Usage

At every login, the IP address of the login request is checked against the validated IP addresses using LoginIp. A match means the login IP address is a known IP address. If there's no match, the address is unknown, and the user is asked to confirm their identity.

LoginHistory

Represents the login history for all successful and failed login attempts for organizations and enabled portals.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Only users with Manager Users permissions can access this object.

Fields

Field	Details
`ApiType`	**Type** string **Properties** Group, Nillable, Sort **Description** Indicates the API type, for example `Soap Enterprise`. Label is **API Type**.
`APIVersion`	**Type** string **Properties** Group, Nillable, Sort **Description** Displays the API version used by the client. Label is **API Version**.
`Application`	**Type** string

Field	Details
	Properties Group, Nillable, Sort **Description** The application used to access the organization. Label is **Application**.
`Browser`	**Type** string **Properties** Group, Nillable, Sort **Description** The current browser version. Label is**Browser**.
`ClientVersion`	**Type** string **Properties** Group, Nillable, Sort **Description** Version of the API client. Label is**Client Version**.
`LoginTime`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Time zone is based on GMT. Label is **Login Time**.
`LoginType`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The type of login, for example, Application. Label is **Login Type**.
`LoginUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description URL from which the login request is coming. Label is **Login URL**.
`NetworkId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the community that the user is logging in to. This field is available in API version 31.0 and later, if Salesforce Communities is enabled for your organization.
`Platform`	**Type** string **Properties** Group, Nillable, Sort **Description** Operating system on the login machine. Label is **Platform**.
`SourceIp`	**Type** string **Properties** Group, Nillable, Sort **Description** IP address of the machine from which the login request is coming. This can be an IPv4 or IPv6 address in API version 23.0 or later. In API version 22.0 or earlier, this is an IPv4 address, and IPv6 addresses are null. Label is **Source IP**.
`Status`	**Type** string **Properties** Group, Nillable, Sort **Description** Displays the status of the attempted login. Status is either success or a reason for failure. Label is **Status**.
`UserId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description ID of the user logging in. Label is **User ID**.

Usage

Not all fields are filterable. You can only filter on the following fields:

- `Id`
- `UserId`
- `LoginTime`
- `LoginType`
- `LoginUrl`
- `NetworkId`

The API allows you to do many powerful queries. A few examples are:

Sample Query	Query String
Simple query showing UserId & LoginTime for each user	`SELECT UserId, LoginTime from LoginHistory;`
Query showing logins only after a specified date and time	`SELECT UserId, LoginTime from LoginHistory WHERE LoginTime > 2010-09-20T22:16:30.000Z;`
Query showing logins for a specific time interval	`SELECT UserId, LoginTime from LoginHistory WHERE LoginTime > 2010-09-20T22:16:30.000Z AND LoginTime < 2010-09-21T22:16:30.000Z;`

MailmergeTemplate

Represents a mail merge template (a Microsoft Word document) used for performing mail merges for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

- All users can view this object, but you need the "Customize Application" permission to modify it.
- Customer Portal users can't access this object.

Fields

Field	Details
`Body`	**Type** base64 **Properties** Create

Field	Details
	Description Required. Microsoft Word document to use as a mail merge template. Due to limitations with Microsoft Word mail merge templates, your client application can specify the Body field when creating these records, but not when updating them. Limit: 5 MB.
`BodyLength`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Length of the Microsoft Word document.
`Description`	**Type** string **Properties** Create, Filter,Group, Nillable, Sort, Update **Description** Required. Text description of this mail merge template. Limit: 255 characters.
`Filename`	**Type** string **Properties** Create, Filter, Group, Sort **Description** Required. File name of the Microsoft Word document that was uploaded as a mail merge template. Limit: 255 characters in length.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`LastUsedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort

Field	Details
	Description Date and time when this MailmergeTemplate was last used.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of this mail merge template.

Usage

Use this object to manage mail merge templates for your organization.

See Also:

Standard and Custom Object Basics

MetricsDataFile

Represents a data file containing usage metrics on all installations of a managed package in a Salesforce instance. This object is available in API version 30.0 and later.

Supported Calls

`query()`, `delete()`

Fields

Field Name	Details
`MetricsDataFile`	**Type** base64 **Properties** Filter, Query, Sort **Description** A text file containing the usage data encoded in Base 64.
`MetricsDataFileContentType`	**Type** string **Properties** Filter, Query, Sort

Field Name	Details
	Description The format of the data file. Currently, the only allowed value is `text/csv`.
`MetricsDataFileLength`	**Type** int **Properties** Filter, Query, Sort **Description** The size of the data file in bytes.
`MetricsEndDate`	**Type** dateTime **Properties** Filter, Query, Sort **Description** The end time and date for the data collection.
`MetricsStartDate`	**Type** dateTime **Properties** Filter, Query, Sort **Description** The start time and date for the data collection.
`MetricsType`	**Type** picklist **Properties** Filter, Query, Sort **Description** The type of data being collected. The possible values are `CustomObject` and `Visualforce`.
`NamespacePrefix`	**Type** string **Properties** Filter, Query, Sort **Description** The namespace prefix of the package for which data is being collected.

Field Name	Details
`SendingInstance`	**Type** string **Properties** Filter, Query, Sort **Description** The server instance from which this data was collected, for example, “na8.”

Usage

Use this object to access customer usage metrics for a managed package. Each record contains one day’s data, on either custom objects or Visualforce pages, for all organizations in a Salesforce instance that have the package installed. The following data is collected each day.

- **Custom objects** — the number of records stored in each custom object.
- **Visualforce pages** — the number of times each Visualforce page was accessed, the number of unique users who accessed it, and the average loading time (in milliseconds).

The custom objects data is a snapshot that reflects the state of the organization at the time the database was sampled, while the Visualforce data covers usage over a 24-hour period.

This feature is intended for API access only. The owner of the package must write a secondary process to retrieve the metrics data from the reporting organization, and export it to another system for analysis.

The usage metrics data for all production organizations in a given instance is merged and written into a text file, in a specified format, once a day. If an instance doesn’t have any organizations with the package installed or any organizations that accessed Visualforce pages in the package, a blank record is created for that day, with `MetricsDataFileLength` set to zero.

In a record for custom objects, each row of the text file contains usage data in the following order.

- Organization ID
- Organization name
- Organization edition
- Organization status
- Name of the custom object
- Number of records of the custom object on the specified day

The custom object count is a snapshot captured once each day. Here’s a section of a sample data file for custom objects. It shows there were 3500 and 1500 records in the `Alpha` and `Beta` custom objects, respectively, in the specified customer organization on the specified day.

```
"00Dxx0000001gbk","org1","Enterprise Edition","TRIAL","Alpha", "3500"
"00Dxx0000001gbk","org1","Enterprise Edition","TRIAL","Beta", "1500"
```

In a record for Visualforce pages, each row of the text file contains usage data in the following order.

- Organization ID
- Organization name
- Organization edition
- Organization status
- Name of the Visualforce page
- Number of times the page was accessed

- Number of unique users who accessed the page
- Average loading time of the page, in milliseconds

The Visualforce counts for each organization measure the number of times the page was viewed in the duration between the start and end times. Here's a section of a sample data file for Visualforce pages.

```
"00Dxx0000001gbk","org1","Enterprise Edition","TRIAL","/apex/gm12__f1","1","1","66.0"
"00Dxx0000001gbk","org1","Enterprise Edition","TRIAL","/apex/gm12__f2","1","1","128.0"
"00Dxx0000001gbk","org1","Enterprise Edition","TRIAL","/apex/gm12__f3","1","1","107.0"
"00Dxx0000001gbf","org1","Enterprise Edition","TRIAL","/apex/gm12__f1","5","1","73.6"
"00Dxx0000001gbf","org1","Enterprise Edition","TRIAL","/apex/gm12__f2","1","1","72.0"
"00Dxx0000001gbf","org1","Enterprise Edition","TRIAL","/apex/gm12__f3","7","1","50.8"
```

MilestoneType

Represents a milestone (required step in a customer support process).

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Update **Description** A description of the milestone.
`Name`	**Type** string **Properties** Create, Filter, idLookup, Update **Description** The name of the milestone.
`Name`	**Type** string **Properties** Create, Filter, idLookup, Update **Description** The name of the milestone.

Field	Details
`RecurrenceType`	**Type** picklist **Properties** Create,Update **Description** The type of recurrence for the milestone.

Usage

Use this object to query and manage the milestone type for CaseMilestone records.

See Also:

CaseMilestone
SlaProcess

Name

This object is used to retrieve information from related records where the related record may be from more than one object type (a polymorphic foreign key). For example, the owner of a case can be either a user or a group (queue). This object allows retrieval of the owner name, whether the owner is a user or a group (queue). You can use a describe call to access the information about parents for an object, or you can use the `who`, `what`, or `owner` fields (depending on the object) in SOQL queries. This object cannot be directly accessed.

Supported Calls

`describeSObjects()`

Fields

Field	Details
`Alias`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The user alias. This field contains a value only if the related record is a user.
`Email`	**Type** email **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description The email address of the user. This field contains a value only if the related record is a user.
FirstName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The first name of the user, contact, or lead.
IsActive	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the related record is an active user (`true`) or not (`false`). This field contains a value only if the related record is a user.
LastName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The last name of the user, contact, or lead.
LastReferencedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
LastViewedDate	**Type** date **Properties** Filter, Nillable, Sort, Update

Field	Details
	Description The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`MiddleName` (beta)	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The middle name of the user, contact, or lead.
`Name`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Name of the parent of the object queried. If the parent is a user, contact, or lead, the value is a concatenation of the `FirstName`, `MiddleName` (beta), `LastName`, and `Suffix` (beta) fields of the related record.
`Phone`	**Type** phone **Properties** Filter, Group, Nillable, Sort **Description** The phone number of the user. This field contains a value only if the related record is a user.
`Profile`	**Type** reference **Properties** Filter, Nillable **Description** The Profile of the user. Only populated if the related record is a user.
`ProfileId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description ID of the user's Profile. Only populated if the related record is a user.
`Suffix` (beta)	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The name suffix of the user, contact, or lead.
`Title`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The title of the user, for example CFO or CEO.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** A list of the types of sObjects that can be an owner of this object. You can use this field to filter on a type of owner, for example, return only the leads owned by a user.
`Username`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Contains the name that a user enters to log into the API or the user interface. The value for this field is in the form of an email address, and is only populated if the related record is a user.
`UserRole`	**Type** picklist **Properties** Filter, Nillable

Field	Details
	Description Name of the `Role` played by the user. Only populated for user rows.
`UserRoleId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the user role associated with this object.

Usage

To query on relationships where the parent may be more than one type of object, use `who`, `what`, or `owner` relationship fields.

See Also:

Standard and Custom Object Basics

Network

Represents a community, which is a customizable public or private space where employees, end-customers, and partners can collaborate on best practices and business processes. Communities give you the opportunity to share information, records, and files with coworkers and related external stakeholders all in one place. This object is available in API version 26.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`, `update()`

Special Access Rules

This object is available only if Salesforce Communities is enabled in your organization.

Fields

Field Name	Details
`CaseCommentEmailTemplateId`	**Type** reference **Properties** Filter, Group, Nillable, Sort, Update **Description** ID of the email template used when submitting a comment on a case. This field is available in API version 28.0 and later.

Field Name	Details
ChangePasswordEmailTemplateId	**Type** reference **Properties** Filter, Group, Sort, Update **Description** ID of the email template used when notifying a user that their password has been reset.
Description	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** Description of the community.
EmailFooterLogoId	**Type** reference **Properties** Filter, Group, Nillable, Sort, Update **Description** ID of the Document object that displays as an image in the footer of community Chatter emails.
EmailFooterText	**Type** string **Properties** Filter, Nillable, Sort, Update **Description** Text that displays in the footer of community Chatter emails.
EmailSenderAddress	**Type** email **Properties** Filter, Group, Sort **Description** Read only. Email address from which community emails are sent. **Note:** To change the `EmailSenderAddress` value, you must first specify `NewSenderAddress`, which triggers the sending of

Field Name	Details
	an address change verification email. After you complete the address verification process, `EmailSenderAddress` changes to the specified `NewSenderAddress`.
`EmailSenderName`	**Type** string **Properties** Filter, Group, Sort, Update **Description** Name from which community emails are sent.
`ForgotPasswordEmailTemplateId`	**Type** reference **Properties** Filter, Group, Sort, Update **Description** ID of the email template used when a user forgets their password.
`Name`	**Type** string **Properties** Filter, Group, Sort, Update **Description** The name of the community.
`NewSenderAddress`	**Type** email **Properties** Filter, Group, Nillable, Sort, Update **Description** Email address that has been entered as the new value for `EmailSenderAddress` but has not been verified yet. After a user has requested to change the sender email address and has successfully responded to the verification email, the `NewSenderAddress` value overwrites the value in `EmailSenderAddress`. This becomes the email address from which community emails are sent.

Field Name	Details
	Note: • If verification is pending for a new email address and you set `NewSenderAddress` to null, this cancels the verification request. • `NewSenderAddress` is automatically set to null after `EmailSenderAddress` has been set to the new verified address. • If verification is pending for a new email address and you specify a different new address for this field, only the latest value is retained and used for verification.
`OptionsAllowMembersToFlag`	**Type** boolean **Properties** Filter, Update **Description** Determines whether users can flag posts, comments, or files as inappropriate in the community. This field is available in API version 29.0 and later. The ability to flag files is available in version 30.0 and later.
`OptionsGuestChatterEnabled`	**Type** boolean **Properties** Filter, Update **Description** Specifies whether guest users can access public Chatter groups in the community without logging in.
`OptionsInvitationsEnabled`	**Type** boolean **Properties** Filter, Update **Description** Determines whether users can invite others to the community.
`OptionsKnowledgeableEnabled`	**Type** boolean **Properties** Filter

Field Name	Details
	Description Determines whether users can see knowledgeable people for topics and endorse people for topics.
`OptionsPrivateMessagesEnabled`	**Type** boolean **Properties** Filter **Description** Determines whether users can send and receive Chatter messages in the community.
`OptionsReputationEnabled`	**Type** boolean **Properties** Filter, Update **Description** Determines if reputation is calculated and displayed for community members. This field is available in API version 31.0 and later.
`OptionsSelfRegistrationEnabled`	**Type** boolean **Properties** Filter, Update **Description** Determines whether external users can self-register to join the community. External users are users with Community, Customer Portal, or partner portal licenses. If `true`, displays a **Not a member?** link on the login page that points to the default self-registration page. This field is available in API version 28.0 and later.
`OptionsSendWelcomeEmail`	**Type** boolean **Properties** Filter, Update **Description** Determines whether a welcome email is sent when a new user is added to the community.
`OptionsSiteAsContainerEnabled`	**Type** boolean

Field Name	Details
	Properties Filter, Update **Description** Determines whether the community uses Site.com pages instead of tabs. This field is available in API version 29.0 and later.
`SelfRegProfileId`	**Type** reference **Properties** Filter, Group, Nillable, Sort, Update **Description** ID of the profile assigned to users who self register. Only applies if self registration is enabled for the community. This field is available in API version 29.0 and later.
`Status`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update **Description** Status of the community. Available values are: • `Live`—The community is online and members can access it. Label is `Published`. • `DownForMaintenance`—The community was previously published, but was taken offline. Members with “Create and Customize Communities” can still access the setup for offline communities regardless of profile or membership. Members are not able to access offline communities, but they still appear in the user interface drop-down as `CommunityName (Offline)`. Label is `Offline`. • `UnderConstruction`—The community has not yet been published. Users with “Create and Customize Communities” can access communities in this status if their profile is associated with the community. Once a community is published, it can never be in this status again. Label is `Preview`.
`UrlPathPrefix`	**Type** string **Properties** Filter, Group, Sort, Update

Field Name	Details
	Description The `UrlPathPrefix` is a unique string at the end of the URL for this community. For example, in the community URL `UniversalTelco.force.com/customers`, `customers` is the `UrlPathPrefix`.
`WelcomeEmailTemplateId`	**Type** reference **Properties** Filter, Group, Sort, Update **Description** ID of the email template used when sending welcome emails to new community members.

Usage

Use this object to find, view, and update communities in your organization. If you have "Modify All Data," "View All Data," or "Create and Customize Communities," you can view all communities in the organization. Users without these permissions only see the `Preview` or `Published` communities that they're members of. If you have "Create and Customize Communities," you can customize community settings.

NetworkActivityAudit

Represents an audit trail of moderation actions in Communities. This object is available in API version 30.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

This object is available only if Salesforce Communities is enabled in your organization.

Fields

Field Name	Details
`Action`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The moderation action a user performed on a post, comment, or file in a community. Values are:

Field Name	Details
	• Flagged—A community member flagged a post, comment or file. • Unflagged—A community member removed the flag from a post, comment or file. • RemovedFlags—A moderator removed all flags from a post, comment, or file. • DeletedFlaggedItem—A moderator deleted a flagged post, comment, or file.
`Description`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Notes entered by the user. If the entity being moderated is a file, records the version number of the file when it was flagged.
`EntityCreatedById`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the user that created the entity being moderated.
`EntityId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the entity that is being moderated.
`EntityType`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The key prefix of entity being moderated.
`Name`	**Type** string

Field Name	Details
	Properties Create, Filter, Group, Sort, Update **Description** The ID of the item being moderated.
`NetworkId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the network (community) where the moderation action was performed.
`ParentEntityId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the parent of the entity on which an action was performed.
`ParentEntityType`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The key prefix of the parent of the entity being audited.

Usage

Use this object to view an audit trail of moderation activity for your communities. You must have the “Modify All Data” permission to access this object.

Users with “Moderate Communities Feeds”, “Moderate Communities Files” or “View All Data” can view the audit trail using reports in the Salesforce user interface.

NetworkMember

Represents a member of a community. Members can be either users in your company or external users with portal profiles. This object is available in API version 26.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`, `update()`

Special Access Rules

This object is available only if Salesforce Communities is enabled in your organization.

Fields

Field Name	Details
`DefaultGroupNotificationFrequency`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update **Description** Required. The default frequency for sending the member's group email notifications when the member joins groups in the community. The valid values are: • `P`—Email on each post • `D`—Daily digests • `W`—Weekly digests • `N`—Never The default value is `W`. However, this field is not currently enabled. These values are reserved for future use.
`DigestFrequency`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update **Description** Required. The frequency for sending the member's personal email digest for the community. The valid values are: • `D`—Daily • `W`—Weekly • `N`—Never The default value is `D`. However, daily and weekly personal digests aren't currently available in community. These values are reserved for future use.
`MemberId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of a person who is a member of a community.

Field Name	Details
`NetworkId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the community that the member is part of.
`PreferencesDisableAllFeedsEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member can automatically receive email for updates in the community, based on the types of feed emails and digests the member has enabled.
`PreferencesDisableBookmarkEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member automatically receives email every time someone comments on a feed item after the member has bookmarked it.
`PreferencesDisableChangeCommentEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member automatically receives email every time someone comments on a change the member has made, such as an update to their profile.
`PreferencesDisableEndorsementEmail`	**Type** boolean **Properties** Filter, Update

Field Name	Details
	Description When `false`, the member automatically receives email every time someone endorses them for a topic.
`PreferencesDisableFollowersEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member automatically receives email every time someone in the community starts following the member.
`DisableItemFlaggedEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the user automatically receives email every time a community member flags a post or comment. This setting only applies for community moderators (with the "Moderate Communities Feeds" permission) and group owners or managers. This field is available in API version 29.0 and later.
`PreferencesDisableLaterCommentEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member automatically receives email every time someone comments on a feed item after the member has commented on the feed item.
`PreferencesDisableLikeEmail`	**Type** boolean **Properties** Filter, Update

Field Name	Details
	Description When `false`, the member automatically receives email every time someone comments on a feed item after the member has liked the feed item.
`PreferencesDisableMentionsPostEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member automatically receives email every time the member is mentioned in posts.
`PreferencesDisableMessageEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member automatically receives email every time the member is sent a Chatter message.
`PreferencesDisableProfilePostEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member automatically receives email every time someone posts to the member's profile.
`PreferencesDisableSharePostEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member automatically receives email every time the member's post is shared.
`PreferencesDisCommentAfterLikeEmail`	**Type** boolean

Field Name	Details
	Properties Filter, Update **Description** When `false`, the member automatically receives email every time someone comments on a post the member has liked.
`PreferencesDisMentionsCommentEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member automatically receives email every time the member is mentioned in comments.
`PreferencesDisProfPostCommentEmail`	**Type** boolean **Properties** Filter, Update **Description** When `false`, the member automatically receives email every time someone comments on posts on the member's profile.
`ReputationPoints`	**Type** double **Properties** Filter, Sort **Description** The number of reputation points the user has accumulated by performing actions in the community.

Usage

Use this object to query members of a certain community and to update their email notification settings. If you have "Modify All Data," "View All Data," or "Create and Customize Communities," you can view all members of any community, regardless of your own membership. If you have "Modify All Data" or "Create and Customize Communities," you can also update any member's email settings. Users without these permissions can update their own email settings and can see members of the communities that they're also members of.

NetworkMemberGroup

Represents a group of members in a community. Members can be either users in your internal organization or external users assigned portal profiles. An administrator adds members to a community by adding a profile or a permission set, and any user with the profile or permission set becomes a member of the community.

Note: If a Chatter customer (from a customer group) is assigned a permission set that is also associated with a community, the Chatter customer won't be added to the community.

This object is available in API version 26.0 and later.

Note: Prior to API version 27.0, this object was called NetworkProfile.

Supported Calls

`create()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`

Note: The `upsert()` call is not supported for this object.

Special Access Rules

This object is available only if Salesforce Communities is enabled in your organization.

Fields

Field Name	Details
`AssignmentStatus`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update **Description** The status of a profile or permission set within a community. Values are: • `Added`—Users with this profile or permission set are members of the community. • `Waiting for Add`—The profile or permission set was added to the community, but the async process hasn't completed yet. After the process is complete, the status is updated to `Added`. • `Waiting for Remove`—Use this status to remove all the community members belonging to a profile or permission set and remove a profile or permission set from a community.
`NetworkId`	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Sort **Description** The ID of the community that this group of members is associated with.
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the profile or permission set associated with the community.

Usage

Use this object to view the profiles or permission sets associated with a particular community. Profiles and permission sets are added and removed asynchronously, so you can also check the status of a profile or permission set that was updated in a community.

If you have "Modify All Data," "View All Data," or "Create and Customize Communities," you can view all profiles or permission sets for any community in the organization, regardless of your membership. If you have "Modify All Data" or "Create and Customize Communities," you can also add profiles or permission sets. Users without these permissions can only find profiles and permission sets for communities that they're members of.

Sample Code

```
// Create a new NetworkMemberGroup with a profile as the ParentId
NetworkMemberGroup nmgInsert = new NetworkMemberGroup();
nmg.setNetworkId("0DBD0000000029o");
nmg.setParentId("00eD0000000z1Ww");
SaveResult[] results = connection.create(new SObject[] { nmgInsert });

// Update an existing NetworkMemberGroup to be removed from the Network
NetworkMemberGroup nmgUpdate = new NetworkMemberGroup();
nmg.setId("0DLD000000003enOAA");
nmg.setAssignmentStatus("WaitingForRemove");
SaveResult[] results = connection.update(new SObject[] { nmgUpdate });
```

NetworkModeration

Represents a flag on an item in a community. This object is available in API version 30.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

This object is available only if Salesforce Communities is enabled in your organization.

Fields

Field Name	Details
`EntityId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** ID of the post, comment, or file that was flagged.
`NetworkId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the community in which the item was flagged.
`Visibility`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Nillable, Sort **Description** Determines visibility of a flagged item. Values are: • SelfAndModerators—The user who flagged the item and any moderators can see the flagged item. This is the default value. • ModeratorsOnly—Only moderators can see the flagged item. If ModeratorsOnly is selected, only moderators can set flags using the API.

Usage

Use this object to view the items flagged for moderation within a community. Additionally, users with “Moderate Feeds” and “Modify All Data” can remove flags.

Flags on items are created either when a member manually flags an item in a community (if flagging is enabled for that community), or when a trigger automatically flags an item because the item met the trigger criteria.

NewsFeed

Represents a single feed item on a user's home tab. A Chatter feed shows recent changes to records that the user is following.

NewsFeed is available in API version 18.0 through API version 26.0. In API version 27.0 and later, NewsFeed is no longer available in the SOAP API. Use the Chatter REST API to access NewsFeed.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of NewsFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea

Field	Details
	Properties Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.

Field	Details
FeedPostId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
InsertedById	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
LastModifiedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.

Field	Details
LikeCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
NetworkScope	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
ParentId	**Type** reference **Properties** Filter, Group, Sort

Field	Details
	Description ID of the newsfeed record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file.

Field	Details
	• `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`:

Field	Details
	• For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to query and retrieve record changes tracked in a news feed. Note the following when working with news feeds:

- This object is read only in the API.
- Users can only query their own news feed.
- Queries retrieve feed items that include mentions from other users.
- Use this object to query and retrieve lead feed items that were associated with a converted lead record.

Users that do not have the “View All Data” permission have the following limitations when querying records:

- Must specify a `LIMIT` clause and the limit must be less than or equal to 1000.
- May include a `WHERE` clause that references NewsFeed fields, but may not include references to fields in related objects. For example, you can filter by `CreatedDate` or `ParentId`, but not by `Parent.Name`.
- May include an `ORDER BY` clause that references NewsFeed fields, but may not include references to fields in related objects. For example, you can `ORDER BY CreatedDate` or `ParentId`, but not by `Parent.Name`.

Tip: To query for the most recent feed items, you should `ORDER BY CreatedDate DESC, Id DESC`.

Note the following SOQL restrictions.

- No SOQL limit if logged-in user has “View All Data” permission. If not, specify a LIMIT clause of 1,000 records or fewer.
- SOQL ORDER BY on fields using relationships is not available. Use ORDER BY on fields on the root object in the SOQL query.

See Also:

EntitySubscription
FeedComment
FeedTrackedChange

Note

Represents a note, which is text associated with a custom object or a standard object, such as a Contact, Contract, or Opportunity.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`Body`	**Type** textarea **Properties** Create, Nillable, Update **Description** Body of the note. Limited to 32 KB.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsPrivate`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** If `true`, only the note owner or a user with the “Modify All Data” permission can view the note or query it via the API. Note that if a user who does not have the “Modify All Data” permission sets this field to `true` on a note that they do not own, then they can no longer query, delete, or update the note. Label is **Private**.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the user who owns the note.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort

Field	Details
	Description Required. ID of the object associated with the note.
`Title`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Title of the note.

Usage

Use this object to manage notes for an object.

See Also:

Standard and Custom Object Basics

NoteAndAttachment

This read-only object contains all notes and attachments associated with an object.

Supported Calls

`describeSObjects()`

Fields

Field	Details
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsNote`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates whether the object contains a note (`true`) or an attachment (`false`).
`IsPrivate`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** If `true`, only the note owner or a user with the "Modify All Data" permission can view the note or query it via the API. Note that if a regular user who does not have "Modify All Data" permission sets this field to `true` on a note that they do not own, then they can no longer query, delete, or update that note. Label is **Private**.
`OwnerId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the user who owns the note and attachment.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the parent object.
`Title`	**Type** string **Properties** Filter, Nillable, Group, Sort **Description** Title of the note.

Usage

Use this object to list all notes and attachments for an object.

To retrieve notes and attachments, issue a describe call on an object, which returns a query result for each activity since the record was created. You can't directly query this object.

See Also:

Note
Attachment

NoteTag

Associates a word or short phrase with a Note.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.

Field Name	Details
Type	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

NoteTag stores the relationship between its parent TagDefinition and the Note being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

ObjectPermissions

Represents the enabled object permissions for the parent PermissionSet. To grant a user access to an object, associate an ObjectPermissions record with a PermissionSet that's assigned to a user. ObjectPermissions records are only supported in PermissionSet, not in Profile. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The `Id` of this object's parent PermissionSet.
PermissionsCreate	**Type** boolean

Field Name	Details
	Properties Create, Filter, Update **Description** If `true`, users assigned to the parent PermissionSet can create records for this object. Requires `PermissionsRead` for the same object to be `true`.
`PermissionsDelete`	**Type** boolean **Properties** Create, Filter, Update **Description** If `true`, users assigned to the parent PermissionSet can delete records for this object. Requires `PermissionsRead` and `PermissionsEdit` for the same object to be `true`.
`PermissionsEdit`	**Type** boolean **Properties** Create, Filter, Update **Description** If `true`, users assigned to the parent PermissionSet can edit records for this object. Requires `PermissionsRead` for the same object to be `true`.
`PermissionsModifyAllRecords`	**Type** boolean **Properties** Create, Filter, Update **Description** If `true`, users assigned to the parent PermissionSet can edit all records for this object, regardless of sharing settings. Requires `PermissionsRead`, `PermissionsDelete`, `PermissionsEdit`, and `PermissionsViewAllRecords` for the same object to be `true`.
`PermissionsRead`	**Type** boolean **Properties** Create, Filter, Update **Description** If `true`, users assigned to the parent PermissionSet can view records for this object.

Field Name	Details
PermissionsViewAllRecords	**Type** boolean **Properties** Create, Filter, Update **Description** If `true`, users assigned to the parent PermissionSet can view all records for this object, regardless of sharing settings. Requires `PermissionsRead` for the same object to be `true`.
SobjectType	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** The object's API name. For example, `Merchandise__c`.

Permission Dependencies

Some user permissions have dependencies on object permissions. For example, if a permission set has the "Transfer Leads" permission, it also has "Read" and "Create" on the leads object.

You can query from ObjectPermissions up to the parent PermissionSet object. For example:

```
SELECT Parent.Name, Parent.PermissionsTransferAnyLead, PermissionsRead, PermissionsCreate
FROM ObjectPermissions
WHERE SobjectType = 'Lead'
```

Determining Object Access with "Modify All Data"

When using SOQL to query object permissions, be aware that some object permissions are enabled because a user permission requires them.

The exception to this rule is when "Modify All Data" is enabled. While it enables all object permissions, it doesn't physically store any object permission records in the database. As a result, unlike object permissions that are required by a user permission—such as "View All Data" or "Import Leads"—the query still returns permission sets with "Modify All Data," but the object permission record will contain an invalid ID that begins with "000". This ID indicates that the object has full access due to "Modify All Data" and the object permission record can't be updated or deleted. To remove full access from these objects, disable "Modify All Data" and then delete the resulting object permission record. This ensures that when using SOQL to find all the objects that have full access, it returns all objects that have this access regardless of whether it's due to "Modify All Data" or because an administrator set full access.

For example, the following will return all permission sets that have "Read" on the Merchandise__c object, regardless of whether it's explicitly defined on the object or implicitly defined through "Modify All Data."

```
SELECT Id, Parent.label, SobjectType, PermissionsRead,
   Parent.PermissionsModifyAllData, ParentId
FROM ObjectPermissions
WHERE PermissionsRead = true and SobjectType = 'Merchandise__c'
```

Nesting Object Permissions

You can nest ObjectPermissions in a PermissionSet query. For example, the following returns any permission sets where "Transfer Leads" is true. Additionally, the result set will include the "Read" object permission on leads. This is done by nesting the SOQL with an object permission query using the relationship name for object permissions: `ObjectPerms`.

```
SELECT Id,Name,PermissionsTransferAnyLead,
(SELECT Id, PermissionsRead from ObjectPerms where SobjectType='Lead')
FROM PermissionSet
WHERE PermissionsTransferAnyLead = true
```

As a result, it's possible to traverse the relationship between the PermissionSet and any child related objects (in this case, ObjectPermissions). You can do this from the PermissionSet object by using the child relationship (`ObjectPerms`, `FieldPerms`, and so on) or from the child object by referencing the PermissionSet with `Parent.`***`permission_set_attribute`***.

It's important to consider when to use a conditional `WHERE` statement to restrict the result set. To query based on an attribute on the permission set object, nest the SOQL with the child relationship. However, to query based on an attribute on the child object, you must reference the permission set parent attribute in your query.

The following two queries return the same columns with different results, based on whether you use the child relationship or parent notation.

```
SELECT Id, Name, PermissionsModifyAllData,
(SELECT Id, SobjectType, PermissionsRead from Objectperms)
FROM PermissionSet
WHERE PermissionsModifyAllData=true
```

versus:

```
SELECT Id, SObjectType, PermissionsRead, Parent.Id, Parent.Name,
Parent.PermissionsModifyAllData
FROM ObjectPermissions
WHERE SObjectType='Merchandise__c'
```

See Also:

PermissionSet
FieldPermissions

ObjectTerritory2AssignmentRule

Represents a territory assignment rule that's associated with an object, such as Account. ObjectTerritory2AssignmentRuleItem can only be created or deleted if the BooleanFilter field on its corresponding ObjectTerritory2AssignmentRule is `null`. Available only if Territory Management 2.0 has been enabled for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

If a territory model is in `Active` state, any user can view that model, including its territories, assignment rules, assigned records, and assigned users. Users cannot view territory models in other states (such as `Planning` or `Archived`).

Fields

Field Name	Details
`BooleanFilter`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Represents advanced filter conditions that were specified for the rule in the online application. For example, “(1 AND 2) OR 3.”
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. The unique name of the object in the API. This name can contain only underscores and alphanumeric characters and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. The field label in the user interface is `Unique Name`. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the rule is active (true) or inactive (false). Via the API, active rules run automatically when object records are created and edited. The exception is when the value of the IsExcludedFromRealign field on an object record is `true`, which prevents record assignment rules from evaluating that record. The exception is when the value of the IsExcludedFromRealign field on an object record is set to `True`, which prevents record assignment rules from evaluating that record.
`Language`	**Type** picklist

Field Name	Details
	Properties Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The language of the master label in the user interface.
`MasterLabel`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. The user interface label for the territory type.
`ObjectType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** The object that the rule is defined for. For API version 31, Account only.
`Territory2ModelId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the territory model.

ObjectTerritory2AssignmentRuleItem

A single row of selection criteria for an ObjectTerritory2AssignmentRule object. ObjectTerritory2AssignmentRuleItem can only be created or deleted if the `BooleanFilter` field on its corresponding ObjectTerritory2AssignmentRule object is a `null` value. Available only if Territory Management 2.0 has been enabled for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

If a territory model is in `Active` state, any user can view that model, including its territories and assignment rules. For territories in an active model, any user can view assigned records and assigned users subject to your organization's sharing settings. Users cannot view territory models in other states (such as `Planning` or `Archived`).

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The standard or custom object field that the rule item will operate on.
`Operation`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The criterion to apply for the rule item. For example: `equals` or `starts with`.
`RuleId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the associated ObjectTerritory2AssignmentRule.
`SortOrder`	**Type** int **Properties** Create, Filter, Group, Sort, Update **Description** The order in which this row is evaluated in relation to other ObjectTerritoryAssignmentRuleItem objects for the given ObjectTerritoryAssignmentRule.
`Value`	**Type** string

Field Name	Details
	Properties Create, Filter, Nillable, Sort, Update **Description** The field value or values to evaluate. For example: if the field is `Billing ZIP/Postal Code`, a value might be `94105`..

ObjectTerritory2Association

Represents an association (by assignment) between a territory and an object record, such as an account. Available only if Territory Management 2.0 has been enabled for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

If a territory model is in `Active` state, any user can view that model, including its territories and assignment rules. For territories in an active model, any user can view assigned records and assigned users subject to your organization's sharing settings. Users cannot view territory models in other states (such as `Planning` or `Archived`).

Fields

Field Name	Details
`AssociationCause`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** The means by which the record was associated with the territory: by rules or manually. User interface field label is `Method`. Manual cause can only be set by a user or via the API. Rules cause is used by the rules engine when the object is assigned.
`ObjectId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the object. For API version 31, Account only.

Field Name	Details
`SobjectType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of the record. For API version 31, Account only.
`Territory2Id`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the territory that the record is assigned to.

OpenActivity

This read-only object is displayed in a related list of open activities—future events and open tasks—related to an object. It includes activities for all contacts related to the object. OpenActivity fields for phone calls are only available if your organization uses Salesforce CRM Call Center.

Supported Calls

`describeSObjects()`

Fields

Field	Details
`AccountId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the related account, which is determined as follows: • The account associated with the `WhatId`, if it exists; or • The account associated with the `WhoId`, if it exists; otherwise • `null` For information on IDs, see ID Field Type.

Field	Details
ActivityDate	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Indicates one of the following: • The due date of a task • The date of an event if IsAllDayEvent is set to true This field has a time stamp that is always set to midnight in the Universal Time Coordinated (UTC) time zone. The time stamp doesn't represent the time of the activity; don't attempt to alter it to accommodate time zone differences. Label is Date.
ActivityType	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Represents one of the following values: Call, Meeting, or Other. Label is Type.
CallDisposition	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Represents the result of a given call, for example, "we'll call back," or "call unsuccessful." Limit is 255 characters.
CallDurationInSeconds	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Duration of the call in seconds.
CallObject	**Type** string

Field	Details
	Properties Filter, Group,Nillable, Sort **Description** Name of a call center. Limit is 255 characters.
`CallType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of call being answered: Inbound, Internal, or Outbound.
`ConnectionReceivedId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the PartnerNetworkConnection that shared this record with your organization. This field is available only if your organization has enabled Salesforce to Salesforce and only in API versions 28.0 and later.
`ConnectionSentId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the PartnerNetworkConnection that your organization shared this record with. This field is available only if your organization has enabled Salesforce to Salesforce, and only in API versions 28.0 and later. The value is always `null`. You can use the PartnerNetworkRecordConnection object to forward records to connections.
`Description`	**Type** textarea **Properties** Nillable **Description** Contains a description of event or task. Limit is 32 KB.

<table>
<tr><th>Field</th><th>Details</th></tr>
<tr><td><code>DurationInMinutes</code></td><td>Type
int

Properties
Filter, Group, Nillable, Sort

Description
Indicates the duration of the event or task.</td></tr>
<tr><td><code>EndDateTime</code></td><td>Type
dateTime

Properties
Filter, Nillable, Sort

Description
Indicates the end date and time of the event or task. Available in versions 27.0 and later. This field is optional, depending on the following:
• If <code>IsAllDayEvent</code> is true, you can supply a value for either <code>DurationInMinutes</code> or <code>EndDateTime</code>. Supplying values in both fields is allowed if the values add up to equal time spans. If both fields are <code>null</code>, the duration defaults to one day.
• If <code>IsAllDayEvent</code> is false, a value must be supplied for either <code>DurationInMinutes</code> or <code>EndDateTime</code>. Supplying values in both fields is allowed if the values add up to the same amount of time.</td></tr>
<tr><td><code>IsAllDayEvent</code></td><td>Type
boolean

Properties
Defaulted on create, Filter, Group, Sort

Description
If the value of this field is set to <code>true</code>, then the activity is an all-day event, and <code>ActivityDate</code> defines the date of the event. Label is <code>All Day Event</code>.</td></tr>
<tr><td><code>IsClosed</code></td><td>Type
boolean

Properties
Defaulted on create, Filter, Group, Sort

Description
Indicates whether a task is closed; value is always <code>false</code>). This field is set indirectly by setting <code>Status</code> on the task—each picklist value has a corresponding <code>IsClosed</code> value. Label is <code>Closed</code>.</td></tr>
</table>

Field	Details
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the activity has been moved to the Recycle Bin (`true`) or not (`false`). Label is `Deleted`.
`IsHighPriority`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates a high-priority task. This field is derived from the `Priority` field.
`IsReminderSet`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether a reminder is set for an activity (`true`) or not (`false`).
`IsTask`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** If the value of this field is set to `true`, then the activity is a task; if the value is set to `false`, then the activity is an event. Label is `Task`.
`IsVisibleInSelfService`	**Type** boolean **Properties** Defaulted on create, Filter **Description** If the value of this field is set to `true`, then the activity can be viewed in the self-service portal. Label is `Visible in Self-Service`.

Field	Details
`Location`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** If the activity is an event, this field represents the location of the event. If the activity is a task, then the value is `null`.
`OwnerId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the user who owns the activity.
`PrimaryAccountId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Contains the `AccountId` value from the activity record. Available in API versions 30.0 and later to organizations that have enabled Shared Activities.
`PrimaryWhoId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Contains the `WhoId` value from the activity record. Available in API versions 30.0 and later to organizations that have enabled Shared Activities.
`Priority`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Indicates the priority of a task, such as high, normal, or low.

Field	Details
`ReminderDateTime`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Represents the time at which a reminder is scheduled to fire if `IsReminderSet` is set to `true`. If `IsReminderSet` is set to `false`, then either the user has deselected the reminder checkbox in the user interface or the reminder has already fired at the time indicated by the value.
`Status`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Indicates the current status of a task, such as In Progress or Complete. Each predefined status field sets a value for `IsClosed`. To obtain picklist values, query TaskStatus.
`Subject`	**Type** combobox **Properties** Filter, Nillable, Sort **Description** Contains the subject of the task or event.
`WhatId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the related object (Campaign, Case, Opportunity, or custom object). Label is `Opportunity/Account ID`.
`WhoId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Indicates the ID of the related lead or contact. If the ID in this field refers to a lead, then the `WhatId` field must be `null`. If your organization uses Shared Activities, when you query activities in API version 30.0 or later, then the returned value of the `WhoId` field matches the value in the queried object, not necessarily in the activity record itself. Label is `Contact/Lead ID`.

Usage

Query activities related to an object

1. Optionally, issue a describe call against the object whose activities you wish to query, to get a suggestion of the correct SOQL to use.
2. Issue a SOQL relationship query with a main clause that references the object, and an inner clause that references the activity history, for example:

```
SELECT (SELECT ActivityDate, Description FROM ActivityHistories)
FROM Account
WHERE Name Like 'XYZ%'
```

or

```
SELECT (SELECT ActivityDate, Description FROM OpenActivities)
FROM Account
WHERE Name Like 'XYZ%'
```

The user interface enforces sharing rules, filtering out related-list items that a user doesn't have permission to see.

The following restrictions on users who don't have "View All Data" permission help prevent performance issues:

- In the main clause of the relationship query, you can reference only one record. For example, you can't filter on all records where the account name starts with 'A'; instead, you must reference a single account record.
- You can't use `WHERE` clauses.
- You must specify a limit of 499 or fewer on the number of rows returned in the list.
- You must sort on `ActivityDate` in descending order and `LastModifiedDate` in descending order; you can display nulls last. For example: `ORDER BY ActivityDate DESC NULLS LAST, LastModifiedDate DESC`.

See Also:

Task

Opportunity

Represents an opportunity, which is a sale or pending deal.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Field Type
`AccountId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the account associated with this opportunity.
`Amount`	**Type** currency **Properties** Create, Filter, Nillable, Sort, Update **Description** Estimated total sale amount. For opportunities with products, the amount is the sum of the related products. Any attempt to update this field, if the record has products, will be ignored. The update call will not be rejected, and other fields will be updated as specified, but the Amount will be unchanged.
`CampaignId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of a related Campaign. This field is defined only for those organizations that have the campaign feature Campaigns enabled. The User must have read access rights to the cross-referenced Campaign object in order to create or update that campaign into this field on the opportunity.
`CloseDate`	**Type** date **Properties** Create, Filter, Group, Sort, Update **Description** Required. Date when the opportunity is expected to close.
`ConnectionReceivedId`	**Type** reference **Properties** Filter, Nillable

Field	Field Type
	Description ID of the PartnerNetworkConnection that shared this record with your organization. This field is only available if you have enabled Salesforce to Salesforce.
`ConnectionSentId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that you shared this record with. This field is only available if you have enabled Salesforce to Salesforce. Beginning with API version 15.0, the `ConnectionSentId` field is no longer supported. The `ConnectionSentId` field is still visible, but the value is null. You can use the new PartnerNetworkRecordConnection object to forward records to connections.
`ContractId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the contract that's associated with this opportunity.
`CurrencyIsoCode`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Nillable, Restricted picklist, Update **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization. If the organization has multicurrency and a Pricebook2 is specified on the opportunity (i.e., the `Pricebook2Id` field is not blank), then the currency value of this field must match the currency of the PricebookEntry records that are associated with any opportunity line items it has.
`Description`	**Type** textarea **Properties** Create, Nillable, Update **Description** Text description of the opportunity. Limit: 32,000 characters.

Field	Field Type
`ExpectedRevenue`	**Type** currency **Properties** Filter, Nillable, Sort **Description** Read-only field that is equal to the product of the opportunity `Amount` field and the `Probability`. You can't directly set this field, but you can indirectly set it by setting the `Amount` or `Probability` fields.
`Fiscal`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** If fiscal years are not enabled, the name of the fiscal quarter or period in which the opportunity `CloseDate` falls. Value should be in YYY Q format, for example, '2006 1' for first quarter of 2006.
`FiscalQuarter`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Represents the fiscal quarter. Valid values are 1, 2, 3, or 4.
`FiscalYear`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Represents the fiscal year, for example, 2006.
`ForecastCategory`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Restricted picklist field. It is implied, but not directly controlled, by the `StageName` field. You can override this field to a different value than is implied by the `StageName`

Field	Field Type
	value. The values of this field are fixed enumerated values. The field labels are localized to the language of the user performing the operation, if localized versions of those labels are available for that language in the user interface. In API version 12.0 and later, the value of this field is automatically set based on the value of the `ForecastCategoryName` and can't be updated any other way. The field properties Create, Defaulted on create, Nillable, and Update are not available in version 12.0.
`ForecastCategoryName`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Available in API version 12.0 and later. The name of the forecast category. It is implied, but not directly controlled, by the `StageName` field. You can override this field to a different value than is implied by the `StageName` value.
`HasOpportunityLineItem`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Read-only field that indicates whether the opportunity has associated line items. A value of `true` means that Opportunity line items have been created for the opportunity. An opportunity can have opportunity line items only if the opportunity has a price book. The opportunity line items must correspond to PricebookEntry objects that are listed in the opportunity Pricebook2. However, you can insert opportunity line items on an opportunity that does not have an associated Pricebook2. For the first opportunity line item that you insert on an opportunity without a Pricebook2, the API automatically sets the `Pricebook2Id` field, if the opportunity line item corresponds to a PricebookEntry in an active Pricebook2 that has a `CurrencyIsoCode` field that matches the `CurrencyIsoCode` field of the opportunity. If the Pricebook2 is not active or the `CurrencyIsoCode` fields do not match, then the API returns an error. You can't update the `Pricebook2Id` or `PricebookId` fields if opportunity line items exist on the Opportunity. You must delete the line items before attempting to update the `PricebookId` field.
`IsClosed`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Field Type
	Description Directly controlled by `StageName`. You can query and filter on this field, but you can't directly set it in a create, upsert, or update request. It can only be set via `StageName`. Label is **Closed**.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsSplit`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Read-only field that indicates whether credit for the opportunity is split between opportunity team members. Label is `IsSplit`.This field is available in versions 14.0 and later for organizations that enabled Opportunity Splits during the pilot period. **Warning:** This field should not be used. However, it's documented for the benefit of pilot customers who find references to `IsSplit` in code.
`IsWon`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Directly controlled by `StageName`. You can query and filter on this field, but you can't directly set the value. It can only be set via `StageName`. Label is **Won**.
`LastActivityDate`	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Value is one of the following, whichever is the most recent: • Due date of the most recent event logged against the record.

Field	Field Type
	• Due date of the most recently closed task associated with the record.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`LeadSource`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Source of this opportunity, such as Advertisement or Trade Show.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. A name for this opportunity. Limit: 120 characters.
`NextStep`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Description of next task in closing opportunity. Limit: 255 characters.

Field	Field Type
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the User who has been assigned to work this opportunity. If you update this field, the previous owner's access becomes Read Only or the access specified in your organization-wide default for opportunities, whichever is greater. If you have set up opportunity teams in your organization, updating this field has different consequences depending on your version of the API: • For API version 12.0 and later, sharing records are kept, as they are for all objects. • For API version before 12.0, sharing records are deleted. • For API version 16.0 and later, users must have the “Transfer Record” permission in order to update (transfer) account ownership using this field.
`Pricebook2Id`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Nillable, Sort, Update **Description** ID of a related Pricebook2 object. The `Pricebook2Id` field indicates which Pricebook2 applies to this opportunity. The `Pricebook2Id` field is defined only for those organizations that have products enabled as a feature. You can specify values for only one field (`Pricebook2Id` or `PricebookId`)—not both fields. For this reason, both fields are declared nillable.
`PricebookId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Nillable, Update **Description** Unavailable as of version 3.0. As of version 8.0, the Pricebook object is no longer available. Use the `Pricebook2Id` field instead, specifying the ID of the Pricebook2 record.
`Probability`	**Type** percent **Properties** Create, Defaulted on create, Filter, Nillable, Sort, Update

Field	Field Type
	Description Percentage of estimated confidence in closing the opportunity. It is implied, but not directly controlled, by the `StageName` field. You can override this field to a different value than what is implied by the `StageName`. **Note:** If you're changing the `Probability` field through the API using a partner WSDL call, or an Apex `before` trigger, and the value may have several decimal places, we recommend rounding the value to a whole number. For example, the following Apex in a `before` trigger uses the `round` method to change the field value: `o.probability = o.probability.round();`
`RecordTypeId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the record type assigned to this object.
`StageName`	**Type** picklist **Properties** Create, Filter, Group, Sort, Update **Description** Required. Current stage of this record. The `StageName` field controls several other fields on an opportunity. Each of the fields can be directly set or implied by changing the `StageName` field. In addition, the `StageName` field is a picklist, so it has additional members in the returned describeSObjectResult to indicate how it affects the other fields. To obtain the stage name values in the picklist, query the OpportunityStage object. If the `StageName` is updated, then the `ForecastCategoryName`, `IsClosed`, `IsWon`, and `Probability` are automatically updated based on the stage-category mapping.
`SyncedQuoteID`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** Read only in an Apex trigger. The ID of the Quote that syncs with the opportunity. Setting this field lets you start and stop syncing between the opportunity and a quote. The ID has to be for a quote that is a child of the opportunity.

Field	Field Type
TotalOpportunityQuantity	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Number of items included in this opportunity. Used in quantity-based forecasting.
Type	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Type of opportunity. For example, Existing Business or New Business. Label is **Opportunity Type**.

Note: If you are importing Opportunity data and need to set the value for an audit field, such as CreatedDate, contact salesforce.com. Audit fields are automatically updated during API operations unless you request to set these fields yourself..

Usage

Use the Opportunity object to manage information about a sale or pending deal. You can also sync this object with a child Quote. To update an Opportunity, your client application needs “Edit” permission on opportunities. You can create, update, delete, and query Attachment records associated with an opportunity via the API. To split credit for an opportunity among multiple opportunity team members, use the OpportunitySplit object.

Client applications can also create or update opportunity objects by converting a Lead with convertLead().

Note: On opportunities and opportunity products, the workflow rules, validation rules, and Apex triggers fire when an update to a child opportunity product or schedule causes an update to the parent record. This means your custom application logic is enforced when there are updates to the parent record, ensuring higher data quality and compliance with your organization’s business policies.

Sample Code—Java

This code starts the sync between an object and a child quote.

```
public void startQuoteSync() {
         Opportunity opp = new Opportunity();
         opp.setId(new ID("006D000000CpOSy"));
         opp.setSyncedQuoteId(new ID("0Q0D000000002OZ"));
   // Invoke the update call and save the results
   try {
      SaveResult[] saveResults = binding.update(new SObject[] {opp});
      // check results and do more processing after the update call ...
   }
   catch (Exception ex) {
      System.out.println("An unexpected error has occurred." + ex.getMessage());
      return;
```

```
 }
}
```

This code stops the sync between an object and a child quote.

```
public void stopQuoteSync() {
         Opportunity opp = new Opportunity();
         opp.setId(new ID("006D000000CpOSy"));
         opp.setFieldsToNull(new String[]  {"SyncedQuoteId"} );
   // Invoke the update call and save the results
   try {
      SaveResult[] saveResults = binding.update(new SObject[] {opp});
      // check results and do more processing after the update call ...
   }
   catch (Exception ex) {
      System.out.println("An unexpected error has occurred." + ex.getMessage());
      return;
 }
}
```

See Also:

- *OpportunityCompetitor*
- *OpportunityHistory*
- *OpportunityLineItem*
- *OpportunityLineItemSchedule*
- *OpportunityFieldHistory*
- *Quote*
- *QuoteLineItem*
- *PartnerNetworkConnection*

OpportunityCompetitor

Represents a competitor on an Opportunity.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`CompetitorName`	**Type** combobox **Properties** Create, Filter, Nillable, Sort, Update **Description** Name of the competitor.

Field	Details
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`OpportunityId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the associated Opportunity.
`Strengths`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** Description of the competitor's strengths. Limit: 1,000 characters.
`Weaknesses`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** Description of the competitor's weaknesses. Limit: 1,000 characters.

Usage

Use this object to manage competitors on an Opportunity, associating multiple competitors on a opportunity and specifying the strengths and weaknesses of each competitor.

See Also:

Opportunity

OpportunityContactRole

Represents the role that a Contact plays on an Opportunity.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`ContactId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of an associated Contact. The API applies user access rights to the associated Opportunity for this object, but not to the associated Contact. The API may return rows from a query on this object that include this field's values for contacts to which the user does not have sufficient access rights. It may also return values for this field for contacts that have been deleted. In either case, the client must perform a query on the contact table for this field's value to determine whether the Contact is accessible to the user and has not been deleted.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsPrimary`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the associated Contact plays the primary role on the Opportunity (`true`) or not (`false`). Each Opportunity has only one primary contact. Label is **Primary**.
`OpportunityId`	**Type** reference

Field	Details
	Properties Create, Filter, Group, Sort, **Description** Required. ID of an associated Opportunity. This field is non-nullable, and it cannot be updated. You must provide a value for this field when creating new records. You can't change it after it has been created.
`Role`	**Type** picklist **Properties** Create, Filter, Group, Sort, Nillable, Update **Description** Name of the role played by the associated Contact on the Opportunity, such as Business User or Decision Maker.

Usage

Records of this object type appear in the user interface on the Opportunity detail page. Like most other objects, records of this object type have their own unique ID that you use when updating or deleting records.

Although allowed, we do not recommend that you create multiple relationships between the same Opportunity and a Contact.

See Also:

Standard and Custom Object Basics

OpportunityFeed

Represents a single feed item in the feed displayed on the detail page for an opportunity record. An opportunity feed shows recent changes to an opportunity record for any fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to opportunities. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the Opportunity object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
Body	**Type** textarea **Properties** Nillable, Sort **Description** The content of OpportunityFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
CommentCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
ConnectionId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
ContentData	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
ContentDescription	**Type** textarea

Field	Details
	Properties Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.

Field	Details
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.

Field	Details
LikeCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
NetworkScope	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
ParentId	**Type** reference **Properties** Filter, Group, Sort

Field	Details
	Description ID of the opportunity record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file.

<table>
<tr><th>Field</th><th>Details</th></tr>
<tr><td></td><td>

<code>CreatedRecordEvent</code>—generated when a user creates a record from the publisher.
<code>DashboardComponentAlert</code>—generated when a dashboard metric or gauge exceeds a user-defined threshold.
<code>DashboardComponentSnapshot</code>—created when a user posts a dashboard snapshot on a feed.
<code>LinkPost</code>—a post with an attached URL.
<code>PollPost</code>—a poll posted on a feed.
<code>ProfileSkillPost</code>—generated when a skill is added to a user's Chatter profile.
<code>QuestionPost</code>—generated when a user posts a question.
<code>ReplyPost</code>—generated when Chatter Answers posts a reply.
<code>RypplePost</code>—generated when a user creates a Thanks badge in Work.com.
<code>TextPost</code>—a direct text entry on a feed.
<code>TrackedChange</code>—a change or group of changes to a tracked field.
<code>UserStatus</code>—automatically generated when a user adds a post. Deprecated.

The following values appear in the <code>Type</code> picklist for all feed objects but apply only to CaseFeed:

<code>CaseCommentPost</code>—generated event when a user adds a case comment for a case object
<code>EmailMessageEvent</code>—generated event when an email related to a case object is sent or received
<code>CallLogPost</code>—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls.
<code>ChangeStatusPost</code>—generated event when a user changes the status of a case
<code>AttachArticleEvent</code>—generated event when a user attaches an article to a case

Note: If you set <code>Type</code> to <code>ContentPost</code>, you must specify <code>ContentData</code> and <code>ContentFileName</code>.
</td></tr>
<tr><td><code>Visibility</code></td><td>
Type
picklist
Properties
Filter, Group, Nillable, Restricted picklist, Sort
Description
Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization.
<code>Visibility</code> can have the following values:

<code>AllUsers</code>—The feed item is available to all users who have permission to see the feed item.
<code>InternalUsers</code>—The feed item is available to internal users only.

Note the following exceptions for <code>Visibility</code>:
</td></tr>
</table>

Field	Details
	• For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for an opportunity record.

See Also:

Opportunity
EntitySubscription
NewsFeed
UserProfileFeed

OpportunityFieldHistory

Represents the history of changes to the values in the fields of an opportunity. This object is available in versions 13.0 and later.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.

Field	Details
`OpportunityId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Opportunity. Label is **Opportunity ID**.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.

Usage

Use this object to identify changes to any fields on an Opportunity. The OpportunityHistory object represents the history of a change to the `Amount`, `Probability`, `Stage`, or `Close Date` fields of an Opportunity.

This object respects field level security on the parent object.

See Also:

Opportunity

OpportunityHistory

Represents the stage history of an Opportunity.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Amount`	**Type** currency **Properties** Filter, Nillable, Sort **Description** Estimated total sale amount.
`CloseDate`	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Date when the opportunity is expected to close.
`ExpectedRevenue`	**Type** currency **Properties** Filter, Nillable, Sort **Description** Calculated revenue based on the `Amount` and `Probability` fields.
`ForecastCategory`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Category that determines the column in which an opportunity is totaled in a forecast. Label is **To ForecastCategory**.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.

Field	Details
`OpportunityId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the associated Opportunity.
`Probability`	**Type** percent **Properties** Filter, Nillable, Sort **Description** Percentage of estimated confidence in closing the opportunity.
`StageName`	**Type** picklist **Properties** Filter, Group, Sort **Description** Name of the current stage of the opportunity (for example, Prospect or Proposal).

Usage

This object represents the history of a change to the `Amount`, `Probability`, `Stage`, or `Close Date` fields of an Opportunity. The OpportunityFieldHistory object represents the history of a change to any of the fields of an Opportunity. To obtain information about how a particular opportunity is progressing, query the OpportunityHistory records associated with a given Opportunity. Please note that if an opportunity's `Amount`, `Probability`, `Stage`, or `Close Date` fields have not changed, nothing will be returned in the OpportunityHistory objects. In this case, query the OpportunityFieldHistory records associated with a given Opportunity to get more information about changes to the opportunity.

This object is read-only. The system generates a new record whenever a user or client application changes the value of any of the above fields; the then-current values of all of these major fields are saved in the newly-generated object.

This object respects field-level security on the parent object.

Note: The record is automatically deleted if its parent Opportunity is deleted.

See Also:

Opportunity

OpportunityLineItem

Represents an opportunity line item, which is a member of the list of Product2 products associated with an Opportunity.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

The user must have the "Edit" permissions on Opportunity records in order to create or update opportunity line items on an opportunity.

Fields

Field	Details
`ConnectionReceivedId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that shared this record with your organization. This field is only available if you have enabled Salesforce to Salesforce.
`ConnectionSentId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that you shared this record with. This field is only available if you have enabled Salesforce to Salesforce. Beginning with API version 15.0, the `ConnectionSentId` field is no longer supported. The `ConnectionSentId` field is still visible, but the value is null. You can use the new PartnerNetworkRecordConnection object to forward records to connections.
`CurrencyIsoCode`	**Type** picklist **Properties** Filter, Restricted picklist **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.

Field	Details
	If the organization has multicurrency enabled, and a Pricebook2 is specified on the parent opportunity (that is, the `Pricebook2Id` field is not blank on the opportunity referenced by this object's `OpportunityId`), then the value of this field must match the currency of the `CurrencyIsoCode` field on the PricebookEntry records that are associated with this object.
`Description`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Text description of the opportunity line item. Limit: 255 characters.
`Discount`	**Type** percent **Properties** Filter, Nillable, Sort **Description** Discount for the product as a percentage. When updating these records: • If you specify `Discount` without specifying `TotalPrice`, the `TotalPrice` is adjusted to accommodate the new `Discount` value, and the `UnitPrice` is held constant. • If you specify both `Discount` and `Quantity`, you must also specify either `TotalPrice` or `UnitPrice` so the system knows which one to automatically adjust.
`HasQuantitySchedule`	**Type** boolean **Properties** Defaulted on create, Filter, Group **Description** Read-only. Indicates whether a quantity schedule has been created for this object (`true`) or not (`false`).
`HasRevenueSchedule`	**Type** boolean **Properties** Defaulted on create, Filter, Sort

Field	Details
	Description Indicates whether a revenue schedule has been created for this object (`true`) or not (`false`). If this object has a revenue schedule, the `Quantity` and `TotalPrice` fields can't be updated. In addition, the `Quantity` field can't be updated if this object has a quantity schedule. Update requests aren't rejected but the updated values are ignored.
`HasSchedule`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** If either `HasQuantitySchedule` or `HasRevenueSchedule` is `true`, this field is also `true`.
`ListPrice`	**Type** currency **Properties** Filter, Nillable, Sort **Description** Corresponds to the `UnitPrice` on the PricebookEntry that is associated with this line item, which can be in the standard pricebook or a custom pricebook. A client application can use this information to show whether the unit price (or sales price) of the line item differs from the pricebook entry list price.
`Name`	**Type** string **Properties** Filter, Nillable, Sort **Description** The opportunity line item name (known as "Opportunity Product" in the user interface). This read-only field is available in API version 30.0 and later.
`OpportunityId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the associated Opportunity.

Field	Details
`PricebookEntryId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** Required. ID of the associated PricebookEntry. Exists only for those organizations that have Products enabled as a feature. You can specify values for either this field or `ProductId`, but not both. For this reason, both fields are declared nillable.
`ProductId`	**Type** reference **Properties** Create, Filter, Nillable **Description** ID of the related Product record. This field is unavailable as of version 3.0 and is only provided for backward compatibility. The Product object is unavailable beginning with version 8.0. Use the `PricebookEntryId` field instead, specifying the ID of the PricebookEntry record.
`Product2Id`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the related Product2 record. This is a read-only field available in API version 30.0 and later. Use the `PricebookEntryId` field instead, specifying the ID of the PricebookEntry record.
`ProductCode`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** This read-only field is available in API version 30.0 and later. It references the value in the ProductCode field of the related Product2 record.
`Quantity`	**Type** double

Field	Details
	Properties Create, Filter, Sort, Update **Description** Read-only if this record has a quantity schedule, a revenue schedule, or both a quantity and a revenue schedule. When updating these records: • If you specify `Quantity` without specifying the `UnitPrice`, the `UnitPrice` value will be adjusted to accommodate the new `Quantity` value, and the `TotalPrice` will be held constant. • If you specify both `Discount` and Quantity, you must also specify either `TotalPrice` or `UnitPrice` so the system can determine which one to automatically adjust.
`RecalculateTotalPrice`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Changes behavior of OpportunityLineItem calculations when a line item has child schedule rows for the `Quantity` value. When enabled, if the rollup quantity changes, then the quantity rollup value is multiplied against the sales price to change the total price. Product2 flag must be set to true.
`ServiceDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Date when the product revenue will be recognized and the product quantity will be shipped. Whether this value is used by customizable forecasting depends upon the Forecast Date setting for the organization: • Opportunity Close Date—`ServiceDate` is ignored. • Product Date—`ServiceDate` is used if not `null`. • Schedule Date—`ServiceDate` is used if not `null` and there are no revenue schedules present for this line item, that is, there are no OpportunityLineItemSchedule records with a field `Type` value of Revenue that are children of this record.
`SortOrder`	**Type** int

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** Number indicating the sort order selected by the user. Client applications can use this to match the sort order in Salesforce.
`Subtotal`	**Type** currency **Properties** Filter, Nillable **Description** Difference between standard and discounted pricing. Converted currency amounts when the opportunity's currency is different from the user's currency.
`TotalPrice`	**Type** currency **Properties** Create, Defaulted on create, Filter, Nillable, Sort, Update **Description** This field is available only for backward compatibility. It represents the total price of the OpportunityLineItem. If you do not specify `UnitPrice`, this field is required. If you specify `Discount` and `Quantity`, this field or `UnitPrice` is required. When updating these records, you can change either this value or the `UnitPrice`, but not both at the same time. This field is nillable, but you can't set both `TotalPrice` and `UnitPrice` to null in the same update request. To insert the `TotalPrice` via the API (given only a unit price and the quantity), calculate this field as the unit price multiplied by the quantity. This field is read-only if the opportunity line item has a revenue schedule. If the opportunity line item does not have a schedule or only has quantity schedule, this field can be updated.
`UnitPrice`	**Type** currency **Properties** Create, Defaulted on create, Filter, Nillable, Sort, Update **Description** The unit price for the opportunity line item. In the Salesforce user interface, this field's value is calculated by dividing the total price of the opportunity line item by the quantity listed for that line item. Label is **Sales Price**. This field or `TotalPrice` is required. You can't specify both. If you specify `Discount` and `Quantity`, this field or `TotalPrice` is required.

Usage

An Opportunity can have associated OpportunityLineItem records only if the Opportunity has a Pricebook2. An OpportunityLineItem must correspond to a Product2 that is listed in the opportunity's Pricebook2. For information about inserting OpportunityLineItem for an opportunity that does not have an associated Pricebook2 or any existing line items, see Effects on Opportunities.

This object is defined only for those organizations that have products enabled as a feature. If the organization does not have the products feature, this object does note appear in the `describeGlobal()` call, and you can't use `describeSObjects()` or query the OpportunityLineItem object.

For a visual diagram of the relationships between OpportunityLineItem and other objects, see Product and Schedule Objects.

Note: If multicurrency option has been enabled, the `CurrencyIsoCode` field is present. It can't be modified, and is always set to the value of the `CurrencyIsoCode` of the parent Opportunity.

Effects on Opportunities

Opportunities that have associated OpportunityLineItem records are affected in the following ways:

- Creating an OpportunityLineItem increments the Opportunity `Amount` value by the `TotalPrice` of the OpportunityLineItem. Additionally, inserting an OpportunityLineItem increments the `ExpectedRevenue` on the opportunity by the `TotalPrice` times the opportunity `Probability`.
- The Opportunity `Amount` becomes a read-only field when the opportunity has line items. The API ignores any attempt to update this field on an opportunity with line items. Update requests are not rejected, but the updated value is ignored.
- You can't update the `PricebookId` field or the `CurrencyIsoCode` field on the opportunity if line items exist. The API rejects any attempt to update these fields on an opportunity with line items.
- When you create or update an OpportunityLineItem, the API verifies that the line item corresponds to a PricebookEntry in the Pricebook2 that is associated with the opportunity. If the opportunity does not have an associated Pricebook2, the API automatically sets the pricebook on the opportunity if the line item corresponds to a PricebookEntry in an active Pricebook2, and if the PricebookEntry has a `CurrencyIsoCode` field that matches the `CurrencyIsoCode` field of the opportunity. If the Pricebook2 is not active or the `CurrencyIsoCode` fields do not match, an error is returned.
- The Opportunity `HasOpportunityLineItem` field is set to `true` when an OpportunityLineItem is inserted for that Opportunity.

See Also:

OpportunityLineItemSchedule

OpportunityLineItemSchedule

Represents information about the quantity, revenue distribution, and delivery dates for a particular OpportunityLineItem.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`CurrencyIsoCode`	**Type** picklist

Field	Details
	Properties Create, Defaulted on create, Filter, Nillable, Restricted picklist, Update **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization. This field is available in version 10.0 and later.
`Description`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Text description of the opportunity line item schedule. Limit: 255 characters. Label is **Comments**.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`OpportunityLineItemId`	**Type** reference **Properties** Create, Filter **Description** Required. ID of the associated OpportunityLineItem.
`Quantity`	**Type** double **Properties** Create, Filter, Nillable, Update **Description** Required. The total number of units to be scheduled in a quantity schedule.
`Revenue`	**Type** currency

Field	Details
	Properties Create, Filter, Nillable, Update **Description** The revenue that should be recognized, or the quantity that should be shipped, or both - depending upon the value of `Type`.
`ScheduleDate`	**Type** date **Properties** Create, Filter, Update **Description** Required. The date the associated OpportunityLineItem is to be scheduled for an event: delivery, shipping, or any other date you wish to track.
`Type`	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** The type of the schedule. Required when inserting an OpportunityLineItemSchedule. Valid values include `Quantity`, `Revenue`, or `Both`.

Allowed Type Field Values

The allowed `Type` values for an OpportunityLineItemSchedule depend on the product-level schedule preferences and whether the line item has any existing schedules. The following criteria must be met:

- The Product2 on which the OpportunityLineItem is based must have the appropriate `CanUseRevenueSchedule` or `CanUseQuantitySchedule` fields (or both) set to `true`.
- When you create a schedule for a line item that does not have any existing schedules, you can specify any valid value.
- If you create a schedule for a line item that already has existing schedules, the new schedule must be consistent with the existing schedules. The following matrix outlines the allowable values:

Value of HasRevenueSchedule on line item	Value of HasQuantitySchedule on line item	Allowable Type Values
false	false	`Revenue`, `Quantity`, both
false	true	`Quantity`
true	false	`Revenue`
true	true	both

Allowed Quantity and Revenue Field Values

The allowable `Quantity` and `Revenue` field values depend on the value of the `Type` field:

Type Value	Allowable Quantity Value	Allowable Revenue Value
`Revenue`	Null	Non-null
`Quantity`	Non-null	Null
both	Non-null	Non-null

The `Quantity` and `Revenue` fields have the following restrictions when this object is updated:

- For a schedule of `Type Quantity`, you can't update a null `Revenue` value to non-null. Likewise for a schedule of `Type Revenue`, you can't update a null `Quantity` value to non-null.
- You can't null out the `Quantity` field for a schedule of `Type Quantity`. Likewise you can't null out the `Revenue` field for a schedule of `Type Revenue`.
- You can't null out either the `Revenue` or `Quantity` fields for a schedule of type `Both`.

Usage

OpportunityLineItemSchedule supports two types of schedules:

- `Quantity` schedules
- `Revenue` schedules

The user must have edit access rights on the Opportunity in order to create or update line item schedules on that opportunity.

Products and Schedules Must Be Enabled

The OpportunityLineItemSchedule object is defined only for those organizations that have the products and schedules features enabled. If the organization does not have the products and schedules features, the OpportunityLineItemSchedule object is not returned in a describe, and you can't describe or query OpportunityLineItemSchedule records.

Effects on Opportunities and Opportunity Line Items

OpportunityLineItemSchedule records affect opportunities and opportunity line items in the following ways:

- Inserting an OpportunityLineItemSchedule of `Type` "Revenue" or "Quantity" increments the `TotalPrice` field on the OpportunityLineItem by the OpportunityLineItemSchedule `Revenue` amount. Inserting an OpportunityLineItemSchedule of `Type Quantity` or `Both` increments the `Quantity` field on the OpportunityLineItem by the OpportunityLineItemSchedule `Quantity` amount.
- Creating an OpportunityLineItemSchedule record affects the original opportunity:
  1. The Opportunity `Amount` is incremented the by OpportunityLineItemSchedule revenue amount
  2. The Opportunity `ExpectedRevenue` is incremented by the line item schedule amount multiplied by the Opportunity `Probability`
- Deleting an OpportunityLineItemSchedule has a similar effect on the related OpportunityLineItem and Opportunity. Deleting an OpportunityLineItemSchedule decrements the OpportunityLineItem `TotalPrice` by the deleted OpportunityLineItemSchedule `Quantity` or `Revenue` amount. The Opportunity `Amount` is also decremented by the OpportunityLineItemSchedule `Quantity` or `Revenue` amount, and the Opportunity `ExpectedRevenue` is reduced by OpportunityLineItemSchedule `Quantity` or `Revenue` amount multiplied by the Opportunity `Probability`.

Deleting an Opportunity Line Item Schedule

Deleting the last remaining schedule will set the corresponding `HasQuantitySchedule` or `HasRevenueSchedule` flags (or both) to `false` on the parent line item.

See Also:

OpportunityLineItem
Product2
delete()

OpportunityOverride

A forecast override of an Opportunity. This read-only object is specific to customizable forecasting. It has a parent-child relationship with LineitemOverride

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Requires the "View All Data" permission.

Fields

Field	Details
`AmountInherited`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the overridden amount rolls up through the forecast hierarchy (`true`), or was overridden by the owner of the OpportunityOverride (`false`).
`ForecastCategoryInherited`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the overridden forecast category rolls up through the forecast hierarchy (`true`) or was overridden by the owner of the OpportunityOverride (`false`).
`IsDeleted`	**Type** boolean

Field	Details
	Properties Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`OpportunityId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the associated Opportunity.
`OutOfDate`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** `true` if a subordinate user's OpportunityOverride, or the opportunity itself, has been updated since this override was last updated, such that an overridden value on this override may be obsolete. For example, a subordinate user has more recently overridden the same period, category, or amount field.
`OverrideAmount`	**Type** currency **Properties** Create, Filter, Sort, Update **Description** The total monetary amount of the opportunity, possibly overridden.
`OverrideComment`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The comment entered on the opportunity forecast edit page. Limit: 255 characters.

Field	Details
OverrideForecastCategory	**Type** picklist **Properties** Create, Filter, Group, Sort, Update **Description** The forecast category of the opportunity, possibly overridden.
OverridePeriodId	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the associated fiscal Period, possibly overridden. If you are using custom fiscal years and the period falls after your custom fiscal year has ended, then the override is not returned.
OverrideQuantity	**Type** double **Properties** Create, Filter, Sort, Update **Description** The quantity of the opportunity, possibly overridden.
OwnerId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the OpportunityOverride owner.
PeriodInherited	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the overridden period rolls up through the forecast hierarchy (`true`) or was overridden by the OpportunityOverride owner (`false`).

Field	Details
QuantityInherited	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the overridden quantity rolls up through the forecast hierarchy (`true`) or was overridden by the OpportunityOverride owner (`false`).

See Also:

LineitemOverride

OpportunityOwnerSharingRule

Represents the rules for sharing an opportunity with users other than the owner.

Note: Contact salesforce.com customer support to enable access to this object for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
Description	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
DeveloperName	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field	Details
	Description The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Rule Name** in the user interface. This field is available in API version 24.0 and later. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`GroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the source group. Opportunities owned by users in the source group trigger the rule to give access.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to **Label** on the user interface.
`OpportunityAccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of sharing being allowed. The possible values are: • `Read` • `Edit`

Field	Details
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the target user or group. The target user or group is being given access.

Usage

Use this object to manage the sharing rules for opportunities. General sharing and Territory-related sharing use this object.

See Also:

Case

OpportunityPartner

This read-only object represents a partner relationship between an Account and an Opportunity. This object is automatically created when a Partner object is created for a partner relationship between an account and an opportunity.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`AccountToId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the partner Account in the partner relationship.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.

Field	Details
IsPrimary	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the account is the opportunity's primary partner (`true`) or not (`false`). Label is **Primary**.
OpportunityId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Opportunity that is in the partner relationship.
Role	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** The UserRole that the Account has on the Opportunity. For example, `Reseller` or `Manufacturer`.

Creating an Account-Opportunity Partner Relationship

When you create a partner relationship between an account and an opportunity (when you create a Partner object and specify the `OpportunityId` field), the API automatically creates an OpportunityPartner with the corresponding values:

- The value of the Partner field `AccountToId` maps to the value of the OpportunityPartner field `AccountToId`.
- The values of the `OpportunityId`, `Role`, and `IsPrimary` fields in both objects are the same.
- If you set the `IsPrimary` value to 1 (`true`) upon insert of a new OpportunityPartner, any other existing primary partners for that opportunity will automatically have the `IsPrimary` value set to 0 (`false`).

This mapping allows the API to manage the objects and their relationship efficiently.

See Also:

Partner
AccountPartner

OpportunityShare

Represents a sharing entry on an Opportunity.

Supported Calls

`describeSObjects()`, `create()`, `delete()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`OpportunityAccessLevel`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Level of access that the user or group has to the opportunity. The possible values are: • `Read` • `Edit` • `All`This value is not valid when creating, updating, or deleting records. This field must be set to an access level that is higher than the organization's default access level for opportunities.
`OpportunityId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Opportunity associated with this sharing entry. This field can't be updated.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Valid values include:

Field	Details
	• `Owner`—User is the owner of the opportunity or is in a UserRole above the opportunity owner in the role hierarchy. • `Manual`—User or group has access because a user with “All” access manually shared the opportunity with them. • `Rule`—User or group has access via an opportunity sharing rule. • `ImplicitChild`—User or group has access to the opportunity on the Account associated with this Opportunity. • `Sales Team`—User has access to the opportunity because she or he is on the opportunity sales team for the opportunity. The OpportunityTeamMember object for this opportunity sets the access level. See OpportunityTeamMember for more information.
`UserOrGroupId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the User or Group that has been given access to the opportunity. This field can’t be updated.

Usage

This object allows you to determine which users and groups can view or edit opportunities owned by other users.

If you attempt to create a record that matches an existing record, any modified fields are updated, the system returns the existing record.

See Also:

Standard and Custom Object Basics

OpportunitySplit

OpportunitySplit credits one or more opportunity team members with a portion of the opportunity amount. This object is available in API version 16.0 and later for pilot customers, and version 28.0 and later for others.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`OpportunityId`	**Type** reference

Field	Details
	Properties Create, FilterGroup, Sort **Description** ID of the opportunity for which the split is being created. Label is `Opportunity ID`.
`Split`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, Sort **Description** Read-only. Automatically-generated number identifying the split within the opportunity.
`SplitAmount`	**Type** currency **Properties** Filter, Nillable, Sort **Description** Monetary amount of the split. Label is `Split Amount`.
`SplitNote`	**Type** string **Properties** Create, Filter, Group, Nillable,Sort, Update **Description** Enter any notes or comments about the split. Character limit is 255. Label is `Split Note`.
`SplitOwnerId`	**Type** reference **Properties** Create, Filter,Group, Sort **Description** The opportunity owner. Label is `User ID`.

Field	Details
`SplitPercentage`	**Type** percent **Properties** Create, Filter, Sort, Update **Description** Split percentage that this team member will receive. If the split type is validated to a 100% total, this number can range from 0 to 100. If the total isn't validated, this number can range from 0 to 1,000. Label is `Split (%)`.
`SplitTypeId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Auto-generated, numeric ID for the split type defined by the OpportunitySplitType object. This field is available in API version 28 and later. If this field is blank, the system automatically specifies the default split type for the opportunity amount, which is validated to 100%.

Usage

Use the OpportunitySplit object to manage splits for an opportunity.

If you change the opportunity owner using the API, the old owner remains on the opportunity team with either Read-only access, or the level of access specified in your organization-wide defaults.

OpportunitySplitType

OpportunitySplitType provides unique labels and behavior for each split type. This object is available in API version 28.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`, `update()`

Fields

Field Name	Details
`Description`	**Type** textarea

Field Name	Details
	Properties Create, Filter, Group, Sort, Update **Description** Describes the purpose of the split type, providing context to future developers.
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. The unique name of the object in the API. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Enables or disables the split type.
`IsTotalValidated`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** If `true`, the split must total 100%. If `false`, the split can total any percentage.
`Language`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update

Field Name	Details
	Description Indicates language of split labels in the user interface.
`MasterLabel`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The user-interface label for the split type.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects. This field can't be accessed unless the logged-in user has the "Customize Application" permission.

Usage

Use OpportunitySplitType if your organization uses different naming conventions for the default Revenue and Overlay splits.

OpportunityStage

Represents the stage of an Opportunity in the sales pipeline, such as New Lead, Negotiating, Pending, Closed, and so on.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`DefaultProbability`	**Type** percent **Properties** Filter, Nillable, Sort, **Description** The default percentage estimate of the confidence in closing a specific opportunity for this opportunity stage value. Label is **Probability (%)**.
`Description`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Description of this opportunity stage value. Limit: 255 characters.
`ForecastCategory`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The default forecast category for this opportunity stage value. The forecast category automatically determines how opportunities are tracked and totaled in a forecast.
`ForecastCategoryName`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Available in API version 12.0 and later. The default forecast category value for this opportunity stage value.
`IsActive`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates whether this opportunity stage value is active (`true`) or not (`false`). Inactive opportunity stage values are not available in the picklist and are retained for historical purposes only.
`IsClosed`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this opportunity stage value represents a closed opportunity (`true`) or not (`false`). Multiple opportunity stage values can represent a closed opportunity. Label is **Closed**.
`IsWon`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this opportunity stage value represents a won opportunity (`true`) or not (`false`). Multiple opportunity stage values can represent a won opportunity. Label is **Won**.
`MasterLabel`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Master label for this opportunity stage value. This display value is the internal label that does not get translated. Limit: 255 characters.
`SortOrder`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Number used to sort this value in the opportunity stage picklist. These numbers are not guaranteed to be sequential, as some previous opportunity stage values might have been deleted.

Usage

This object represents a value in the opportunity stage picklist, which provides additional information about the stage of a Opportunity, such as its probability or forecast category. Query this object to retrieve the set of values in the opportunity stage picklist, and then use that information while processing Opportunity records to determine more information about a given opportunity. For example, the application could test whether a given opportunity is won or not based on its `StageName` value and the value of the `IsWon` property in the associated OpportunityStage object.

This object is read-only via the API.

See Also:

Standard and Custom Object Basics

OpportunityTag

Associates a word or short phrase with an Opportunity.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter

Field Name	Details
	Description ID of the parent TagDefinition object that owns the tag.
Type	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

OpportunityTag stores the relationship between its parent TagDefinition and the Opportunity being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

OpportunityTeamMember

Represents a User on the opportunity team of an Opportunity.

See also UserTeamMember, which represents a User who is on the default opportunity team of another user.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
CurrencyIsoCode	**Type** picklist **Properties** Filter, Restricted picklist **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.

Field	Details
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**. **Note:** Any OpportunityTeamMember record that is deleted is not moved to the Recycle Bin, and cannot be undeleted, unless the record was cascade-deleted when deleting a related Opportunity. For directly deleted OpportunityTeamMember records, don't use the isDeleted field to detect deleted records in SOQL queries or `queryAll()` calls. Instead, use `getDeleted()`.
`Name`	**Type** string **Properties** Filter, Nillable, Sort **Description** The team member name. This read-only field is available in API version 30.0 and later.
`OpportunityAccessLevel`	**Type** picklist **Properties** Filter, Nillable, Restricted picklist **Description** Opportunity access level for this team member. Valid values: • `None` • `Read` • `Edit` • `All`
`OpportunityId`	**Type** reference **Properties** Create, Filter

Field	Details
	Description Required. ID of the Opportunity associated with this opportunity team. This field can't be updated.
`TeamMemberRole`	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** Role that the team member has on the opportunity. The valid values are set by the organization's administrator in the Opportunity Team Roles picklist. Label is **Team Role**.
`UserId`	**Type** reference **Properties** Create, Filter **Description** Required. ID of the User who is a member of the opportunity team. This field can't be updated.

Usage

If you attempt to create a record for this object and it matches an existing record, the system updates any modified fields and returns the existing record.

In the user interface, users can set up an opportunity team for the opportunities they own. The opportunity team includes other users that are working on the opportunity with them. This object is available only in organizations that have enabled team selling.

Note: The behavior for changing ownership of opportunities is different using the user interface when the previous owner is on an opportunity team. For example, when you change the owner of an opportunity using the API, the previous owner's access becomes Read Only or the access specified in your organization-wide default for opportunities, whichever is greater. However, performing this same action in the user interface allows you to select the access level for the previous owner when he or she is on an opportunity team.

See Also:

UserTeamMember

Order

Represents an order associated with a contract or an account.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
`AccountId`	**Type** reference **Properties** Create, Filter, Nillable, Group, Sort **Description** Required. ID of the Account associated with this order.
`ActivatedById`	**Type** reference **Properties** Filter, Group, Nillable, Sort,Update **Description** ID of the User who activated this order.
`ActivatedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort, Update **Description** Date and time when the order was activated.
`BillingCity`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** City for the billing address for this order. Maximum size is 40 characters.
`BillingCountry`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description Country for the billing address for this order. Maximum size is 80 characters.
`BillingCountryCode (beta)`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ISO country code for the billing address for this order.
`BillingLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `BillingLongitude` to specify the precise geolocation of a billing address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`BillingLongitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `BillingLatitude` to specify the precise geolocation of a billing address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`BillingPostalCode`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Postal code for the billing address for this order. Maximum size is 20 characters.
`BillingState`	**Type** string

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** State for the billing address for this order. Maximum size is 80 characters.
`BillingStateCode (beta)`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ISO state code for the order's billing address.
`BillingStreet`	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Street address for the billing address.
`BillToContactId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the contact that the order will be billed to.
`CompanyAuthorizedById`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the user who authorized the account associated with the order.
`CompanyAuthorizedDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description Date on which the order was authorized by your organization.
`ContractId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** ID of the contract associated with the order.
`CustomerAuthorizedById`	**Type** **Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the contact who authorized the order.
`CustomerAuthorizedDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Date on which the contact authorized the order.
`Description`	**Type** textarea **Properties** Create, Nillable, Update **Description** Description of the order.
`EffectiveDate`	**Type** date **Properties** Create, Filter, Group, Sort, Update **Description** Date at which the order becomes effective. Label is **Order Start Date**.

Field Name	Details
`End Date`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Date at which the order ends. Label is **Order End Date**.
`IsReductionOrder`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** Read-only. Determines whether an order is a reduction order. Label is **Reduction Order**.
`LastReferencedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`Name`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Name for this order.

Field Name	Details
`OpportunityId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID for the opportunity that's associated with this order.
`OrderNumber`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** Order number assigned to this order (not the unique, system-generated ID assigned during creation). Maximum size is 30 characters.
`OrderReferenceNumber`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Order reference number assigned to this order. Maximum size is 80 characters.
`OriginalOrderId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Nillable, Sort **Description** Optional. ID of the original order that a reduction order is reducing, if the reduction order is reducing a single order. Label is **Original Order**. Editable only if `isReductionOrder` is `true`. If the reduction order is reducing more than one order, leave blank.
`PoDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description Date of the purchase order.
PoNumber	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Number identifying the purchase order. Maximum is 80.
Pricebook2Id	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Nillable, Sort, Update **Description** Required. ID of the price book associated with this order.
QuoteId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the quote that's associated with this order. If you set `QuoteId` to null, `QuoteLineItemId` on all of the order's child order products is set to null.
RecordTypeId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the record type assigned to this order.
ShippingCity	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description City of the shipping address. Maximum size is 40 characters.
`ShippingCountry`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Country of the shipping address. Maximum size is 80 characters.
`ShippingCountryCode (beta)`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ISO country code for the order's shipping address.
`ShippingLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `ShippingLongitude` to specify the precise geolocation of a shipping address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`ShippingLongitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `ShippingLatitude` to specify the precise geolocation of an address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`ShippingPostalCode`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description Postal code of the shipping address. Maximum size is 20 characters.
`ShippingState`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** State of the shipping address. Maximum size is 80 characters.
`ShippingStateCode (beta)`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ISO state code for the order's shipping address.
`ShippingStreet`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Street address of the shipping address. Maximum of 255 characters.
`ShipToContactId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the contact the order will be shipped to.
`Status`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Picklist of values that indicate order status. Each value is within one of two status categories defined in `StatusCode`. For example, the status

Field Name	Details
	picklist might contain Draft, Ready for Review, and Ready for Activation values with a `StatusCode` of Draft.
`StatusCode`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update **Description** The status category for the order. An order can be either Draft or Activated. Label is **Status Category**.
`TotalAmount`	**Type** currency **Properties** Filter, Sort **Description** The total amount for the order products associated with this order.
`Type`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The type of order.

Usage

The `Status` field specifies the current state of an order. Status strings represent its current state (Draft or Activated).

When a client application creates an order, the `Status Code` must be Draft and the `Status` must be any value that corresponds to a `Status Code` of Draft. The application can subsequently activate an order by updating it and setting the value in its `Status` field to an Activated state; however, the `Status` field is the only field you can update when activating the order.

After an order is activated, your client application can change the `Status` back to the Draft state—but only if the order doesn't have any child reduction order products. Your client application can delete orders when the `Status` is Draft but not when its `Status` is Activated.

Client applications can use the API to create, update, delete, and query any Attachment associated with an order.

See Also:

OrderFeed
OrderHistory
OrderItem

OrderFeed

Represents a single feed item in the feed displayed on an order.

Supported Calls

`delete()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field Name	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of the OrderFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ContentData`	**Type** base64 **Properties** Nillable

Field Name	Details
	Description The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Create, Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`InsertedById`	**Type** reference

Field Name	Details
	Properties Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`.

Field Name	Details
	• For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the order record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as a FeedItem object of `Type` `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of FeedItem: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging).

Field Name	Details
	For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated.

Field Name	Details
	• `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case. **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for an order record.

See Also:

Order

OrderHistory

Represents historical information about changes that have been made to the standard fields of the associated order, or to any custom fields with history tracking enabled.

Supported Calls

`getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Name of the order field that was modified, or a special value to indicate some other modification to the order.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** New value of the modified order field. Maximum of 255 characters.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** Previous value of the modified order field. Maximum of 255 characters.
`OrderId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the order associated with this record.

Usage

Order history entries are automatically created each time an order is modified.

Two rows are added to this record when foreign key fields change. One row contains the foreign key object names that display in the online application. For example, `Jane Doe` is recorded as the name of a Contact. The other row contains the actual foreign key ID that is only returned to and visible from the API.

This object respects field-level security on the parent object.

See Also:

Order

OrderItem

Represents an order product that your organization sells.

Supported Calls

`create()`, `delete()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`

Fields

Field Name	Details
`AvailableQuantity`	**Type** double **Properties** Filter, Nillable, Sort **Description** Amount of an order product that is available to be reduced. Value must be greater than or equal to 0. An order product is reducible only if `AvailableQuantity` is greater than 0. Value is always 0 if the order product's parent order is a reduction order.
`Description`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Text description of this object.
`EndDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Optional. Last day the order product is available.
`ListPrice`	**Type** currency

Field Name	Details
	Properties Filter, Nillable, Sort **Description** List price for the order product. Value is inherited from the associated `PriceBookEntry` upon order product creation.
`OrderId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the order that this order product is a child of.
`OrderItemNumber`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** Automatically-generated number that identifies the order product.
`OriginalOrderItemId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** Required if `isReductionOrder` on the parent order is `true`. ID of the original order product being reduced.
`PricebookEntryId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the associated PricebookEntry. This field must be specified when creating OrderItem records. It can’t be changed in an update.
`Quantity`	**Type** double

Field Name	Details
	Properties Create, Filter, Sort, Update **Description** Number of units of this order product. If you specify `Quantity` without specifying `UnitPrice`, `UnitPrice` is adjusted to accommodate the new `Quantity` value while keeping `TotalPrice` constant.
`QuoteLineItemId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the associated quote line item. If this field is specified, the quote line item's QuoteId must match the QuoteId for the order product's parent order.
`ServiceDate`	**Type** date **Properties** Create, Filter, Nillable, Sort, Update **Description** Start date for the order product. Label is **Start Date**.
`UnitPrice`	**Type** currency **Properties** Create, Filter, Nillable, Sort, Update **Description** Unit price for the order product. In the Salesforce user interface, this field's value is calculated by dividing the total price of the order product by the quantity listed for that order product.

Usage

An order can have associated order product records only if the order has a price book associated with it. An order product must correspond to a product that is listed in the order's price book.

See Also:

- OrderItemFeed
- OrderItemHistory
- Order

OrderItemFeed

Represents a single feed item in the feed displayed on the detail page for an order product record. An order product feed shows changes to an order product for fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to order products in Salesforce.

Supported Calls

`delete()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of the FeedItem. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ContentData`	**Type** base64 **Properties** Nillable

Field Name	Details
	Description The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in ContentData.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if Type is ContentPost. The name of the file uploaded to the feed. Setting ContentFileName automatically sets Type to ContentPost.
`ContentSize`	**Type** string **Properties** Group, Nillable, Sort **Description** Read-only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** int **Properties** Group, Nillable, Sort **Description** Read-only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`InsertedById`	**Type** reference

Field Name	Details
	Properties Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then `InsertedById` is set to the ID of the logged-in user.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`.

Field Name	Details
	• You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** string **Properties** Filter, Group, Sort **Description** ID of the account record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as a FeedItem object of `TypeContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the FeedItem. When the Type is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of FeedItem: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging).

Field Name	Details
	For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated.

Field Name	Details
	• `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case. **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for an order product record.

See Also:

OrderItem

OrderItemHistory

Represents the history of changes to the values in the fields of an order product.

Supported Calls

`getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.
`OrderItemId`	**Type** reference **Properties** Group, Sort **Description** ID of the order product. Label is **Order Product ID**.

Usage

Use this object to identify changes to an order product.

This object respects field-level security on the parent object.

See Also:

OrderItem

Organization

This object represents key configuration information for an organization. You must have the “View All Data” permission to access this object.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`

Executing a SOQL SELECT query returns the value of fields in this object, but no value is visible for some of the fields.

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`Address` (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the address. Read-only. See Address Compound Fields for details on compound address fields.
`AllowsSelfServiceLogin`	**Type** boolean **Properties** Defaulted on create, Filter, Update **Description** Indicates whether the organization allows Self-Service login (`true`) or not (`false`).
`City`	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** Name of the city for the organization's address.
`ComplianceBccEmail`	**Type** email

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** Email address for compliance blind carbon copies. Limit: 80 characters.
`Country`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Name of the country for the organization's address. Limit: 80 characters.
`CountryCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO country code for the organization’s address.
`DailyWebToCaseCount`	**Type** int **Properties** Filter, Nillable **Description** The number of web form submissions that have been converted to cases for the day.
`DailyWebToCaseLimit`	**Type** int **Properties** Filter, Nillable **Description** The maximum number of web form submissions that can be converted to cases per day.
`DailyWebToLeadCount`	**Type** int

Field	Details
	Properties Filter, Nillable **Description** The number of web form submission that have been converted to leads for the day
`DailyWebToLeadLimit`	**Type** int **Properties** Filter, Nillable **Description** The maximum number of web form submissions that can be converted to leads per day.
`DefaultAccountAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** In API version 10.0 and later, represents the default access level for accounts, contracts, and assets. The possible values are: • `None` • `Read` • `Edit` In versions before 10.0, `DefaultAccountAndContactAccess` represented this value.
`DefaultAccountAndContactAccess`	**Type** picklist **Properties** Filter, Nillable, Restricted picklist **Description** Default access level for accounts, contacts, contracts, and assets. This field is supported for backward compatibility only and is not available in API version 10.0 or later. In version 10.0 and later, use either `DefaultAccountAccess` or `DefaultContactAccess`.
`DefaultCalendarAccess`	**Type** picklist

Field	Details
	Properties Filter, Group, Nillable, Restricted picklist, Sort **Description** Default access level for calendars. The possible values are listed, followed by the user interface labels in parentheses: • `HideDetails` (Hide Details) • `HideDetailsInsert` (Hide Details and Add Events) • `ShowDetails` (Show Details) • `ShowDetailsInsert` (Show Details and Add Events) • `AllowEdits` (Full Access)
`DefaultCampaignAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Default access level for campaigns. The possible values are: • `None` • `Read` • `Edit` • `All`
`DefaultCaseAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Default access level for cases. The possible values are: • `None` • `Read` • `Edit` • `ReadEditTransfer`
`DefaultContactAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Default access level for contacts. The possible values are:

Field	Details
	• `None` • `Read` • `Edit` • `ControlledByParent` In versions before 10.0, `DefaultAccountAndContactAccess` represented this value. **Note:** When `DefaultContactAccess` is set to “Controlled by Parent,” you can’t update the `ContactAccessLevel` field.
`DefaultLeadAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Default access level for leads. The possible values are: • `NoneRead` • `Edit` • `ReadEditTransfer`
`DefaultLocaleSidKey`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update **Description** Default locale SID key.
`DefaultOpportunityAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Default access level for opportunities. The possible values are: • `None` • `Read` • `Edit`

Field	Details
`DefaultPricebookAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Default access level for price books. The possible values are listed, followed by the user interface labels in parentheses: • `None` (No access) • `Read` (Read only) • `ReadSelect` (Use)
`DefaultTerritoryAccountAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Default access level for accounts in territories. The possible values are: • `Read` • `Edit` • `All`
`DefaultTerritoryCaseAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Default access level for cases associated with accounts in territories. The possible values are: • `None` • `Read` • `Edit`
`DefaultTerritoryContactAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Default access level for contacts associated with accounts in territories. The possible values are: • `NoneRead` • `Edit` **Note:** When `DefaultContactAccess` is set to “Controlled by Parent” you can’t update this field.
`DefaultTerritoryOppAccess`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Default access level for opportunities in territories. Valid values: • `NoneRead` • `Edit`
`Division`	**Type** string **Properties** Filter, Nillable, Update **Description** The name of the division for this organization. This field is not related to the Division object.
`Fax`	**Type** phone **Properties** Filter, Group, Nillable, Sort, Update **Description** Fax number. Limit: 40 characters.
`FiscalYearStartMonth`	**Type** int **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Number that corresponds to the month that this organization's fiscal year starts.
`HomepageHtml`	**Type** textarea **Properties** Nillable, Update **Description** The Home tab custom links and company message for this organization.
`InstanceName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Read-only. The name of the instance. Available in API version 31.0 or later.
`IsSandbox`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Read-only. Indicates whether the current organization is a sandbox (`true`) or production (`false`) instance. Available in API version 31.0 or later.
`LanguageLocaleKey`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update **Description** The same as `Language`, the two-to-five character code which represents the language and locale ISO code. This controls the language for labels displayed in an application.
`LastWebToCaseDate`	**Type** dateTime

Field	Details
	Properties Filter, Nillable **Description** The last date that a web form submission was converted to a case.
`LastWebToLeadDate`	**Type** dateTime **Properties** Filter, Nillable **Description** The last date that a web form submission was converted to a lead.
`Latitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `Longitude` to specify the precise geolocation of an address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`Longitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `Latitude` to specify the precise geolocation of an address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`MaxActionsPerRule`	**Type** int **Properties** Filter, Nillable **Description** Maximum number of actions per workflow, assignment, escalation, and auto-response rules. This field is unavailable in version 15.0 and later.

Field	Details
MaxRulesPerEntity	**Type** int **Properties** Filter, Nillable **Description** Maximum number of rules per object, inclusive of workflow, assignment, escalation, and auto-response rules. This field is unavailable in version 15.0 and later.
MonthlyPageViewsEntitlement	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of page views allowed for the current calendar month for the sites in your organization. To access this field, Force.com Sites must be enabled for your organization. This field is generally available in API versions 18.0 and later.
MonthlyPageViewsUsed	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of page views used in the current calendar month for the sites in your organization. To access this field, Force.com Sites must be enabled for your organization. This field is generally available in API versions 18.0 and later.
Name	**Type** string **Properties** Filter, Group, Sort, Update **Description** The name of the organization.
OrganizationType	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field	Details
	Description Edition of the organization, for example Enterprise Edition or Unlimited Edition.
`Phone`	**Type** phone **Properties** Filter, Group, Nillable, Sort, Update **Description** Phone number for the organization.
`PostalCode`	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** Postal code for the address of the organization. Limit: 20 characters.
`PreferencesEventScheduler`	**Type** boolean **Properties** Update **Description** Indicates whether opportunities require products (`true`) or not (`false`).
`PreferencesRequireOpportunityProducts`	**Type** boolean **Properties** Filter, Update **Description** Indicates whether opportunities require products (`true`) or not (`false`).
`PreferencesS1BrowserEnabled`	**Type** boolean **Properties** Filter, Update

Field	Details
	Description Indicates whether the Salesforce1 mobile browser app is enabled for all users in your organization (`true`) or is disabled for all users (`false`). This field is available in API version 29.0 or later.
`PrimaryContact`	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** Name of the primary contact for the organization. Limit: 80 characters.
`ReceivesAdminInfoEmails`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the organization receives administrator emails (`true`) or not (`false`).
`ReceivesInfoEmails`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the organization receives informational email from Salesforce (`true`) or not (`false`).
`SelfServiceCasePlural`	**Type** string **Properties** Filter, Nillable, Update **Description** The plural version of the term used to represent the Case object in the Self-Service portal.
`SelfServiceCaseSingle`	**Type** string

Field	Details
	Properties Filter, Nillable, Update **Description** The singular version of the term used to represent the Case object in the Self-Service portal.
`SelfServiceCaseSubmitRecordTypeId`	**Type** reference **Properties** Filter, Nillable, Update **Description** The ID of the record type associated with a case submitted via the Self-Service portal.
`SelfServicDefaultCaseOrigin`	**Type** string **Properties** Filter, Nillable, Update **Description** The default origin of a case submitted via the Self-Service portal.
`SelfServiceEmailSenderAddress`	**Type** email **Properties** Filter, Nillable, Update **Description** The Self-Service email address from which new Self-Service user and password email messages are sent, such as `support@acme.com`.
`SelfServiceEmailSenderName`	**Type** string **Properties** Filter, Nillable, Update **Description** The name associated with the email address in the `SelfServiceEmailSenderAddress` field, such as `Acme Customer Support`.

Field	Details
SelfServiceEmailUserOnCaseCreationTemplateId	**Type** reference **Properties** Filter, Nillable, Update **Description** The ID of the email template used when email is sent to a Self-Service user when he or she creates a case.
SelfServiceEnabledForResponseRules	**Type** boolean **Properties** Filter, Nillable, Update **Description** Indicates whether the Self-Service portal is enabled for auto-response rules (`true`) or not (`false`).
SelfServiceFeatureConfig	**Type** int **Properties** Filter, Nillable, Update **Description** An integer representing the active Self-Service feature configuration for this organization.
SelfServiceLogoutUrl	**Type** url **Properties** Filter, Nillable, Update **Description** The Web page that displays when a Self-Service user logs out of the Self-Service portal.
SelfServiceMaxNumSuggestions	**Type** int **Properties** Filter, Nillable, Update **Description** The maximum number of suggested solutions allowed for a Self-Service case.

Field	Details
SelfServiceNewCommentCheckedByDefault	**Type** boolean **Properties** Defaulted on create, Filter, Update **Description** If `true`, When a customer notification is automatically sent when a new comment is added to a case.
SelfServiceNewCommentTemplateId	**Type** reference **Properties** Filter, Nillable, Update **Description** The ID of the email template used to send a notification to Self-Service users when a public comment is added to one of their cases.
SelfServiceNewPassTemplateId	**Type** reference **Properties** Filter, Nillable, Update **Description** The ID of the email template used when new passwords are generated for Self-Service users.
SelfServiceNewUserTemplateId	**Type** reference **Properties** Filter, Nillable, Update **Description** The ID of the email template used when new Self-Service users are enabled.
SelfServicePageHeight	**Type** int **Properties** Filter, Nillable, Update **Description** The maximum height in pixels of Self-Service pages.

Field	Details
SelfServicePageWidth	**Type** int **Properties** Filter, Nillable, Update **Description** The maximum width in pixels of Self-Service pages.
SelfServiceSelfClosedCaseStatus	**Type** picklist **Properties** Filter, Nillable, Update **Description** The default status for cases closed by Self-Service users.
SelfServiceSolutionCategoryAvailable	**Type** boolean **Properties** Defaulted on create, Filter, Update **Description** Indicates whether solution categories are available in the Self-Service portal (`true`) or not (`false`).
SelfServiceSolutionCategoryStartNodeId	**Type** reference **Properties** Filter, Nillable, Update **Description** The ID of the top-level category in the Self-Service portal.
SelfServiceSolutionPlural	**Type** string **Properties** Filter, Nillable, Update **Description** The plural version of the term used to represent the Solution object in the Self-Service portal.
SelfServiceSolutionSingle	**Type** string

Field	Details
	Properties Filter, Nillable, Update **Description** The singular version of the term used to represent the Solution object in the Self-Service portal.
SelfServiceStyleSheetUrl	**Type** url **Properties** Filter, Nillable, Update **Description** The public URL of your organization's Self-Service portal stylesheet.
SelfServiceWelcomePageConfig	**Type** int **Properties** Filter, Nillable, Update **Description** Integer that represents the welcome page configuration for the Self-Service portal.
SelfServiceWelcomeText	**Type** string **Properties** Filter, Nillable, Update **Description** The custom welcome message displayed at the top of the Self-Service home page when Self-Service users log in. Limit: 32,000 characters.
State	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** State of the address of the organization. Limit: 80 characters.
StateCode	**Type** picklist

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The ISO state code for the organization's address.
`Street`	**Type** textarea **Properties** Filter, Group, Nillable, Sort, Update **Description** Street address for the organization. Limit: 255 characters.
`TrialExpirationDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date that this organization's trial license expires.
`UiSkin`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The user interface theme selected for the organization.
`UsesStartDateAsFiscalYearName`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the calendar year when the fiscal year begins is referred to as the year of the company's fiscal year (`true`) or not (`false`). For example, if the fiscal year begins in February 2006, a `true` value means the fiscal year is FY2006, and a `false` value means the fiscal year is FY2007.
`UsesWebToCase`	**Type** boolean

Field	Details
	Properties Filter, Nillable, Update **Description** Indicates whether this organization can use Web-to-Case (`true`) or not (`false`).
`UsesWebToLead`	**Type** boolean **Properties** Filter, Nillable, Update **Description** Indicates whether this organization can use Web-to-Lead (`true`) or not (`false`).
`WebToCaseAssignedEmailTemplateId`	**Type** reference **Properties** Filter, Nillable, Update **Description** The ID of the email template used when a new case is assigned to a user via Web-to-Case.
`WebToCaseCreatedEmailTemplateId`	**Type** reference **Properties** Filter, Nillable, Update **Description** The ID of the email template used when a new case is created via Web-to-Case.
`WebToCaseDefaultCreatorId`	**Type** reference **Properties** Filter, Nillable, Update **Description** The ID of the user specified as the default creator of cases created via Web-to-Case.
`WebToCaseDefaultOrigin`	**Type** string

Field	Details
	Properties Filter, Group, Nillable, Sort, Update **Description** The default value for the **Case Origin** field on cases submitted via Web-to-Case. Limit: 40 characters.

Usage

Query this object to obtain information about an organization's settings. Only one organization object exists per organization.

See Also:

Standard and Custom Object Basics

OrgWideEmailAddress

Represents an organization-wide email address for user profiles.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`Address`	**Type** email **Properties** Create, Filter, Sort, Update **Description** The organization-wide email address.
`IsAllowAllProfiles`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** If `true`, any user profile in your organization can use this object. If `false`, only specified user profiles can use this object when sending email. If you do not have the appropriate user profile, you can't use this object.

Field	Details
DisplayName	**Type** string **Properties** Create, Filter, Sort, Update **Description** The name that is used to identify the sender of the email.

Usage

This object represents an email alias for user profiles. You can pass in the ID to an OrgWideEmailAddress record when calling `sendEmail()` for a SingleEmailMessage.

OwnedContentDocument

Represents a file owned by a user. This object is available in versions 30.0 and later.

Supported Calls

`describeSObjects()`

Fields

Field Name	Details
ContentDocumentId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the document.
ContentSize	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Size of the document in bytes.
ContentUrl	**Type** url

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** URL for links and Google Docs. This field is set only for links and Google Docs, and is one of the fields that determine the `FileType`. This field is available in API version 31.0 and later.
`FileExtension`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** File extension of the document. This field is available in API version 31.0 and later.
`FileType`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Type of document, determined by the file extension.
`OwnerId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the owner of the document.
`Title`	**Type** string **Properties** Filter, Group, idLookup, Sort **Description** Title of the document.

PackageLicense

Represents a license for an installed managed package. This object is available in API version 31.0 and later.

Supported Calls

`query()`, `retrieve()`

Fields

Field Name	Details
`AllowedLicenses`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of users allowed to use the package.
`ExpirationDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date and time when the package license expires.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Sort **Description** The namespace prefix associated with the package.
`Status`	**Type** picklist **Properties** Filter, Group, Sort **Description** The status of the license. Possible values are: `Active`, `Expired`, `Free`, and `Trial`.
`UsedLicenses`	**Type** int

Field Name	Details
	Properties Create, Filter, Nillable, Sort **Description** The number of users who have a license to the package.

Usage

Use this object to determine the number of licenses allowed and in use for a managed package installed in your organization.

The following example demonstrates the use of the API to manage licenses for a package. The example defines an Apex class that does the following.

- Retrieves the PackageLicense record for the specified package (identified by its namespace prefix).
- Defines a function that returns a list of all users with the specified profile.
- Creates a UserPackageLicense record for each user with that profile, which has the effect of assigning a license for the package to all users with that profile.
- Returns an error message if the number of users exceeds the number of available licenses.

```
public class AssignPackageLicense {

    static String PACKAGE_NAMESPACE_PREFIX = 'acme_101';
    static String PROFILE_ID = '00exx000000jz1SAAQ';
    public static String exceptionText {get; set;}

    public AssignPackageLicense() {
         exceptionText = 'Initialized';
    }

   static List<User> getUsersWithProfile(){
      String userQuery = 'SELECT Id FROM User WHERE ProfileId = :PROFILE_ID';
      List<User> matchingUsers = new List<User>();
      matchingUsers = [SELECT Id FROM User WHERE ProfileId = :PROFILE_ID];
      return matchingUsers;
    }

    public static void assignLicenseByProfile() {
        //find the PackageLicense Id
        PackageLicense pl = [SELECT Id, NamespacePrefix, AllowedLicenses, UsedLicenses,
              ExpirationDate,Status FROM PackageLicense WHERE
              NamespacePrefix = :PACKAGE_NAMESPACE_PREFIX];
        System.assert(pl != null, 'PackageLicense cannot be null.');
        List<User> usersToAssignLicenses = getUsersWithProfile();
        List<UserPackageLicense> firstUPLs = new List<UserPackageLicense>();

        //create a new UserPackageLicense record for each user with the specified profile
        for (Integer i = 0; i< usersToAssignLicenses.size(); i++){
              UserPackageLicense upl = new UserPackageLicense();
              upl.PackageLicenseId = pl.Id;
              upl.UserId = usersToAssignLicenses[i].Id;
              firstUPLs.add(upl);
        }

        try {
          //bulk insert
          insert(firstUPLs);
          } catch(DmlException e) {
             for (Integer i = 0; i < e.getNumDml(); i++) {
             // process exception here
               System.debug(e.getDmlMessage(i));
               String status = e.getDmlStatusCode(i);
```

```
                System.debug(status + ' ' + e.getDmlMessage(i));
                if(status.equals('LICENSE_LIMIT_EXCEEDED')){
                 exceptionText = 'You tried to assign more licenses than available. '
                 +' You tried to create '+ firstUPLs.size()+' licenses but only have '
                 + (pl.AllowedLicenses - pl.UsedLicenses) + ' licenses free.';
                 System.debug(exceptionText);
                }
              }
          }
     }
}
```

Partner

Represents a partner relationship between two Account records or between an Opportunity and an Account.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

- You must have the "View All Data" permission to access this object via the API. All of the Partner fields are accessible in the `describeSObjects()` and `query()` calls. You cannot `update()` or `upsert()` partners via the API.
- Customer Portal users can't access this object.

Fields

Field	Details
`AccountFromId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** Required if `OpportunityId` is null. ID of the main Account in a partner relationship between two accounts. Specifying this field when creating a Partner record creates two AccountPartner objects, one for each direction of the relationship. If you specify the `OpportunityId` field, you can't specify this field as well.
`AccountToId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Partner Account related to either an opportunity or an account. You must specify this field when creating an opportunity Partner or an Account Partner.

Field	Details
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsPrimary`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** Valid for Opportunity Partners only. Indicates that the account is the primary partner for the opportunity. Only one account can be marked as primary for an opportunity. If you set this field to 1 (`true`) upon insert of a new opportunity partner, any other primary partners for that opportunity will automatically have this field set to 0 (`false`). Label is **Primary**.
`OpportunityId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** Required if `AccountFromId` is null. ID of the Opportunity in a partner relationship between an Account and an Opportunity. Specifying this field when creating a record creates an OpportunityPartner. If you specify the `AccountFromId` field, you can't specify this field as well.
`Role`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, **Description** UserRole that the account has towards the related opportunity or account, such as consultant or distributor.

Roles

In the Salesforce user interface, system administrators can set up the valid role values and their corresponding reverse role values in the PartnerRole object. Each account in the relationship is assigned a `Role` (such as `Consultant` or `Distributor`) designating that account's role towards the related account or opportunity.

Creating an Account-Opportunity Partner Relationship

When you create a partner relationship between an account and an opportunity (when you create a Partner object and specify the `OpportunityId` field), the API automatically creates an OpportunityPartner with the corresponding values:

- The value of the Partner field `AccountToId` maps to the value of the OpportunityPartner field `AccountToId`.
- The values of the `OpportunityId`, `Role`, and `IsPrimary` fields in both objects are the same.
- If you set the `IsPrimary` value to 1 (`true`) upon insert of a new OpportunityPartner, any other existing primary partners for that opportunity will automatically have the `IsPrimary` value set to 0 (`false`).

This mapping allows the API to manage the objects and their relationship efficiently.

Creating an Account-Account Partner Relationship

When you create a partner relationship between two accounts (when you create a Partner object and specify the `AccountFromId`), the API automatically creates two AccountPartner objects, one for the forward relationship and one for the reverse. For example, if you create a Partner object with "Acme, Inc." as the `AccountFromId` and "Acme Consulting" as the `AccountToId`, the API automatically creates two AccountPartner objects:

- The forward relationship AccountPartner with "Acme, Inc." as the `AccountFromId` and "Acme Consulting" as the `AccountToId`.
- The reverse relationship AccountPartner with "Acme Consulting" as the `AccountFromId` and "Acme, Inc." as the `AccountToId`.
- The value of the `Role` field in the reverse relationship AccountPartner is set to the PartnerRole object `ReverseRole` value associated with the value of the `Role` field in the forward relationship AccountPartner.

This mapping allows the API to manage the objects and their relationship efficiently.

See Also:

AccountPartner
OpportunityPartner
UserRole
PartnerRole

PartnerNetworkConnection

Represents a Salesforce to Salesforce connection between Salesforce organizations.

Supported Calls

`create()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`AccountId`	**Type** reference

Field	Details
	Properties Filter, Nillable **Description** ID of the Account associated with this connection.
`ConnectionName`	**Type** string **Properties** Filter, idLookup **Description** A descriptive name for the connection. Limit: 225 characters.
`ConnectionStatus`	**Type** picklist **Properties** Filter **Description** Picklist of values. The picklist includes the following values: • `Sent` • `Received` • `Pending` • `Accepted` • `Rejected` • `Inactive`
`ContactId`	**Type** reference **Properties** Filter, Nillable **Description** Contact associated with this connection.
`CreatedDate`	**Type** dateTime **Properties** Filter **Description** The date and time that the connection was created.

Field	Details
PrimaryContactId	**Type** reference **Properties** Filter **Description** ID of the User associated with this connection.
ResponseDate	**Type** dateTime **Properties** Filter **Description** The date and time that the connection was accepted or rejected.

Usage

Represents Salesforce to Salesforce connections. This object is referenced by lead, opportunity, account, contact, task, product, opportunity product, and custom objects that have been shared with other organizations, enabling you to determine which connections share the record with you. If the organization does not have Salesforce to Salesforce enabled, the PartnerNetworkConnection object is not available, and you can't access it via the API.

See Also:

PartnerNetworkRecordConnection

PartnerNetworkRecordConnection

Represents a record shared between Salesforce organizations using Salesforce to Salesforce.

Supported Calls

`create()`, `delete()`, `query()`, `retrieve()`

Fields

Field	Details
ConnectionId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** Required. ID of the connection a record is shared with.

Field	Details
EndDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Date that sharing of the record was stopped.
LocalRecordId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the shared record.
ParentRecordId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** ID of the parent record of the shared record.
PartnerRecordId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the shared record in the connection's organization.
RelatedRecords	**Type** string **Properties** Create, Filter, Nillable, Sort **Description** A comma-separated list of API names for child records to be shared with a parent record.
SendClosedTasks	**Type** boolean

Field	Details
	Properties Create, Defaulted on create, Filter, Group, Sort **Description** Forwards closed tasks related to the shared record.
SendEmails	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** Sends an email notifying the connection's representative that you have forwarded the record to them. Only new recipients of a record will receive a notification email.
SendOpenTasks	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** Forwards open tasks related to the shared record.
StartDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Date that the shared record was accepted.
Status	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The status of the shared record. One of the following values: • Active (received) • Active (sent) • Connected • Inactive • Inactive (converted)

Field	Details
	• `Inactive (deleted)` • `Pending (sent)`

Usage

When you create a PartnerNetworkRecordConnection, you forward a record to a connection.

Note: Attempting to forward a record from an object to which the connection is not subscribed results in an `Invalid Partner Network Status` error.

Don't forward a record to the connection that originally shared it. Doing so causes errors when Apex triggers run.

When you delete a PartnerNetworkRecordConnection, you stop sharing a record with a connection.

- To share a record, use the following fields: `LocalRecordID` and `ConnectionId`
- To share a child of a parent record, use the following fields: `LocalRecordID`, `ConnectionId`, and `ParentRecordID`
- To share a child of a parent record and its child records, use the following fields: `LocalRecordID`, `ConnectionId`, `ParentRecordID`, and `RelatedRecords`

If the organization does not have Salesforce to Salesforce enabled, the PartnerNetworkRecordConnection object is not available, and you can't access it using the API.

Sample Code—Apex

The following example shows how to forward a record.

```
List<PartnerNetworkConnection)> connMap = new List<PartnerNetworkConnection>(
    [select Id, ConnectionStatus, ConnectionName from PartnerNetworkConnection
        where ConnectionStatus = 'Accepted']
);
for(PartnerNetworkConnection network : connMap) {
     PartnerNetworkRecordConnection newrecord = new PartnerNetworkRecordConnection();

    newrecord.ConnectionId = network.Id;
    newrecord.LocalRecordId = accountId;
    newrecord.RelatedRecords = 'Contact,Opportunity,Orders__c';
    newrecord.SendClosedTasks = true;
    newrecord.SendOpenTasks = true;
    newrecord.SendEmails = true;

    insert newrecord;
}
```

The following example shows how to stop sharing a record.

```
List<PartnerNetworkRecordConnection> recordConns = new List<PartnerNetworkRecordConnection>(

    [select Id, Status, ConnectionId, LocalRecordId from PartnerNetworkRecordConnection
        where LocalRecordId in :accounts]
);

   for(PartnerNetworkRecordConnection recordConn : recordConns) {
        if(recordConn.Status.equalsignorecase('Sent')){ //account is connected - outbound
            delete nets;
```

```
        }
    }
```

See Also:

PartnerNetworkConnection

PartnerRole

Represents a role for an account Partner, such as consultant, supplier, and so on.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`MasterLabel`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Master label for this partner role value. This display value is the internal label that does not get translated. Limit: 255 characters.
`ReverseRole`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Name of the reverse role that corresponds to this partner role. For example, if the role is “subcontractor,” then the reverse role might be “general contractor.” In the user interface, assigning a partner role to an account creates a reverse partner relationship so that both accounts list the other as a partner.
`SortOrder`	**Type** int **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Number used to sort this value in the partner role picklist. These numbers are not guaranteed to be sequential, as some previous partner role values might have been deleted.

Usage

This object represents a value in the partner role picklist. In the user interface, the partner role picklist provides additional information about the role of a Partner, such as their corresponding reverse role. Query this object to retrieve the set of values in the partner role picklist, and then use that information while processing PartnerRole records to determine more information about a given partner role. For example, the application could determine the reverse role of a given Partner `Role` value and the value of the `ReverseRole` property in the associated PartnerRole object.

See Also:

Standard and Custom Object Basics

Period

Represents a fiscal period defined in FiscalYearSettings.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`EndDate`	**Type** date **Properties** Filter, Group, Sort **Description** The last date of the fiscal period.
`FiscalYearSettingsId`	**Type** reference **Properties** Filter, Nillable, Group, Sort **Description** The parent record for this period.

Field	Details
IsForecastPeriod	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the period is associated with customizable forecasts (`true`) or not (`false`).
Number	**Type** int **Properties** Filter, Nillable, Group, Sort **Description** If the labeling scheme of your fiscal year's quarters or months is numbered, this field indicates the relative number of the row.
PeriodLabel	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** If the months in your fiscal year use custom names, then this field contains the appropriate name for rows of type Month.
QuarterLabel	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** If the quarters in your fiscal year use custom names, then this field contains the appropriate name for rows of type Quarter.
StartDate	**Type** date **Properties** Filter, Group, Sort **Description** The first date of the fiscal period.

Field	Details
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Indicates whether the period is of type Month, Quarter, Week, or Year. Label is the field value.

See Also:

FiscalYearSettings

PermissionSet

Represents a set of permissions that's used to grant additional access to one or more users without changing their profile or reassigning profiles. You can use permission sets to grant access, but not to deny access. This object is available in API version 22.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `search()`, `update()`, `upsert()`

Fields

Field Name	Details
`Description`	**Type** string **Properties** Create, Filter, Nillable, Group, Sort, Update **Description** A description of the permission set. Limit: 255 characters.
`IsOwnedByProfile`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** If `true`, the permission set is owned by a profile. Available in API version 25.0 and later.

Field Name	Details
`Label`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The permission set label, which corresponds to **Label** in the user interface. Limit: 80 characters.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. Label corresponds to **API Name** in the user interface. Limit: 80 characters.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix for a permission set that's been installed as part of a managed package. If the permission set isn't packaged or is part of an unmanaged package, this value is empty. Available in API version 23.0 and later.
`Permissions`*`PermissionName`*	**Type** boolean **Properties** Create, Filter, Update **Description** One field for each permission. If `true`, users assigned to this permission set have the named permission. The number of fields varies depending on the permissions for the organization and license type. **Tip:** To get a list of available permissions in the SOAP API, use `describeSObjects()`.

Field Name	Details
ProfileId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** If the permission set is owned by a profile, this field returns the ID of the Profile. If the permission set isn't owned by a profile, this field returns a null value. Available in API version 25.0 and later.
UserLicenseId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** ID of the UserLicense associated with this permission set. This field is nillable in API version 26.0 and later.

Usage

Use the PermissionSet object to query existing permission sets.

For example, to search for all permission sets that contain the "Modify All Data" permission:

```
SELECT Name, PermissionsModifyAllData
FROM PermissionSet
WHERE PermissionsModifyAllData=true
```

When combined with the PermissionSetAssignment object, you can create a nested query that returns all users assigned to a particular permission like "Modify All Data":

```
SELECT Name, (SELECT AssigneeId FROM Assignments)
FROM PermissionSet
WHERE PermissionsModifyAllData=true
```

You can also create a permission set, or delete a permission set if it isn't assigned to a user.

User Licenses

The user license controls the permissions that are available in a permission set.

Every permission set may be associated with a user license. If you plan to assign a permission set to multiple users with different licenses, leave `UserLicenseId` empty. If only users with one type of license will use this permission set, set the `UserLicenseId` to that single user license. To get the `UserLicenseId`, run this query:

```
SELECT Id, Name
FROM UserLicense
```

Alternatively, to query a user or profile for the `UserLicenseId`.

```
SELECT Id, Profile.UserLicenseId
FROM User
```

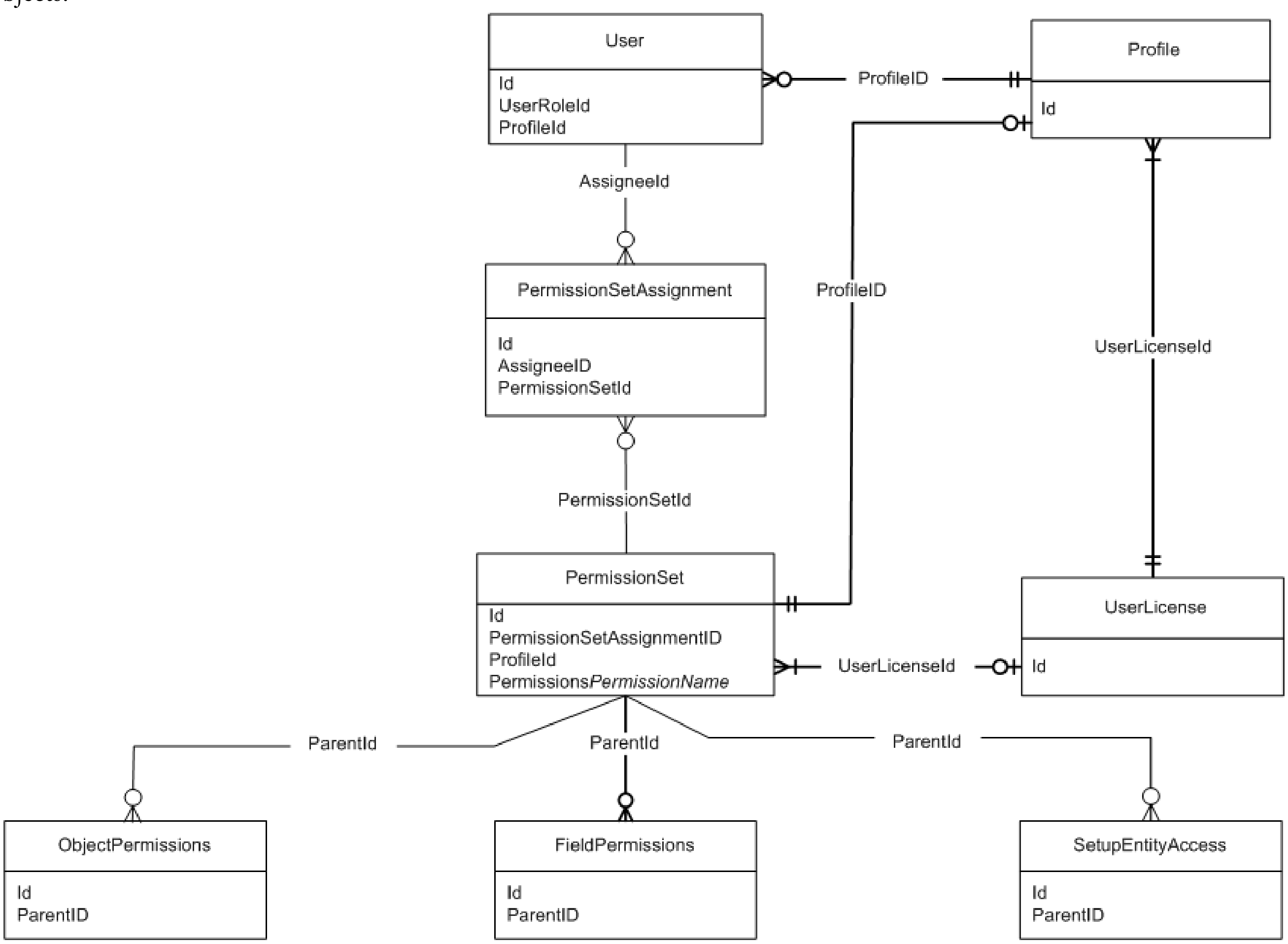
Child Objects

When using the API, think of each permission set or related set of access controls as starting as an empty container that you fill with permission records.

In the API, a permission set can contain user, object, and field permissions, as well as setup entity access settings for other settings, such as Apex classes.

- ObjectPermissions and FieldPermissions objects are available in API version 24.0 and later.
- The SetupEntityAccess object is available in API version 25.0 and later.

Only user permissions are managed in the PermissionSet API object; all other permission types are managed in child API objects.



In these child objects, access is stored in a record, while the absence of a record indicates no access. In order to return a record in a SOQL query, a minimum permission or setting is required for each child object.

Because permissions are stored in related objects, it's important to understand what questions to ask when using SOQL. For example, you may want to know which permission sets have "Delete" on an object or have the right to approve a return merchandise authorization (where the approval checkbox is controlled with field permissions). Asking the right questions when using SOQL with permission sets will ensure that you get the information you need to make an informed decision, such as whether to migrate permissions or assign a permission set to a user.

For example, the following returns all permission sets where the “Read” permission is enabled for the Merchandise__c object.

```
SELECT SobjectType, ParentId, PermissionsRead
FROM ObjectPermissions
WHERE PermissionsRead = True AND SobjectType = 'Merchandise__c'
```

You can query for all permission sets that have “Read” on an object. However, you can’t query for permission sets that have no access on an object, because no records exist for that object. For example, the following returns no records because the object must have at least “Read” to return any records.

```
SELECT SobjectType, ParentId, PermissionsRead
FROM ObjectPermissions
WHERE PermissionsRead = False AND SobjectType = 'Merchandise__c'
```

If you have at least the “Read” permission on an object, you can create a conditional query on other permissions in the same object. For example, the following returns any records where the object has at least the “Read” permission but not the “Edit” permission.

```
SELECT ParentId, PermissionsRead, PermissionsEdit
FROM ObjectPermissions
WHERE PermissionsEdit = False AND SobjectType = 'Merchandise__c'
```

To set an object or field permission to no access, just delete the record that contains the permission. For example, to disable all object permissions in the Merchandise__c object for a particular permission set, first query to retrieve the ID of the object permission record.

```
SELECT Id
FROM ObjectPermissions
WHERE SobjectType = 'Merchandise__c'
```

Then delete the IDs returned from the query.

Note: If you try to update the object or field permissions by setting all permissions to false, the permission record is automatically deleted. As a result, any subsequent queries for the record ID won’t return any results and you must add a new permission record to grant access.

Viewing a Permission Set with Nested Queries

You can build on the PermissionSet object using child relationships that show all of the permissions in a single permission set. For example, the following returns all permission sets and displays the “Transfer Leads” permission as well as any “Read” permissions on any objects and fields.

```
SELECT Label, PermissionsTransferAnyLead,
(SELECT SobjectType, PermissionsRead FROM ObjectPerms),
(SELECT SobjectType, Field, PermissionsRead FROM FieldPerms)
FROM PermissionSet
```

Associated Profiles

In API version 25.0 and later, every profile is associated with a permission set that stores the profile’s user, object, and field permissions, as well as setup entity access settings. Permission sets that are owned by profiles can be queried but not modified.

The following example returns all permission sets, including those owned by a profile.

```
SELECT Id, Label, ProfileId, Profile.Name
FROM PermissionSet
```

The following returns all permission sets except those owned by profiles.

```
SELECT Id, Label, ProfileId, Profile.Name, IsOwnedByProfile
FROM PermissionSet
WHERE IsOwnedByProfile = FALSE
```

Because permission sets have child objects in the API, you can query their values on permission sets owned by a profile. For example, the following returns all enabled object permission records for profiles only.

```
SELECT Id,ParentId, PermissionsRead, SobjectType, Parent.ProfileId
FROM ObjectPermissions
WHERE Parent.IsOwnedByProfile = TRUE
```

Once you have the IDs for permission sets that are owned by profiles as well as those not owned by profiles, you can use the PermissionSetAssignment object to determine whether users can access an object or field via a permission in their profile or any of their permission sets. For example, the following SOQL query returns all users who have the "Read" permission on the Merchandise__c object and specifies whether the permission is granted through a profile or permission set.

```
SELECT Assignee.Name, PermissionSet.Id, PermissionSet.isOwnedByProfile
FROM PermissionSetAssignment
WHERE PermissionSetId
IN (SELECT ParentId
FROM ObjectPermissions
WHERE SObjectType = 'Merchandise__c' AND PermissionsRead = true)
```

Note: For permission sets that are owned by profiles, don't use any Name and Label values that are returned in a query, as they can change at any time.

See Also:

ObjectPermissions
FieldPermissions
SetupEntityAccess
PermissionSetAssignment
Profile

PermissionSetAssignment

Represents the association between a User and a PermissionSet. This object is available in API version 22.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
AssigneeId	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Sort **Description** ID of the User to assign the permission set specified in `PermissionSetId`.
`PermissionSetId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the PermissionSet to assign to the user specified in `AssigneeId`.

Usage

Finding Permission Set Assignments

Use the PermissionSetAssignment object to query permission set assignments to find out which permission sets are assigned to which users. Because each user may be assigned to many permission sets and each permission set may be assigned to many users, each PermissionSetAssignment ID represents the association of a single user and single permission set.

For example, to search for all of the permission sets assigned to a particular user:

```
SELECT Id, PermissionSetId
FROM PermissionSetAssignment
WHERE AssigneeId = '005600000017cKt'
```

To search for all users assigned to a particular permission set:

```
SELECT Id, AssigneeId
FROM PermissionSetAssignment
WHERE PermissionSetId = '0PS30000000000e'
```

You can also create a new permission set assignment, or use delete to remove a permission set that's assigned to a user. To update an assignment, delete an existing assignment and insert a new one.

User Licenses

When assigning a permission set, if the PermissionSet has a `UserLicenseId`, its `UserLicenseId` and the Profile `UserLicenseId` must match. To determine a user's license assignment, query the user's profile and then query the profile's license.

For example, to find a user's profile ID:

```
SELECT Id, ProfileId
FROM User
WHERE Id = '005D0000001GMAT'
```

To find a permission set's `UserLicenseId`:

```
SELECT Id, UserLicenseId
FROM PermissionSet
WHERE Id = '0PS30000000000e'
```

If the IDs match, the assignment succeeds.

To find all the permission sets with no license that are assigned to any user:

```
SELECT Id, Assignee.Name, PermissionSet.Name
FROM PermissionSetAssignment
WHERE PermissionSet.UserLicenseId = null
```

See Also:

PermissionSet

PermissionSetLicense

Represents a license that's used to enable one or more users to receive a specified permission without changing their profile or reassigning profiles. You can use permission set licenses to grant access, but not to deny access. This object is available in API version 29.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`DeveloperName`	**Type** string **Properties** Filter, Group, Sort **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.

Field Name	Details
ExpirationDate	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** The date at which the permission set license expires.
Language	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The language of the permission set license.
MasterLabel	**Type** string **Properties** Filter, Group, Sort **Description** The label of the permission set license. Label is **Permission Set License Label**.
MaximumPermissions*PermissionName*	**Type** boolean **Properties** Filter, **Description** One field for each permission. For example, MaximumPermissionsIdentityConnect corresponds to the "Use Identity Connect" permission. If true, this PermissionSetLicense grants the specified permission. The number of fields varies depending on the permissions available for the organization.
PermissionSetLicenseKey	**Type** string **Properties** Filter, Group, idLookup, Sort

Field Name	Details
	Description A string that uniquely identifies a particular permission set license.
`Status`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The status of a permission set license. If `Active`, the permission set license is available. If `Disabled`, the permission set license has expired.
`TotalLicenses`	**Type** int **Properties** Filter, Group, Sort **Description** The total number of this permission set license that are available to your organization.
`UsedLicenses`	**Type** int **Properties** Filter, Group, Sort **Description** The number of this permission set license that are currently assigned to users.

Usage

Users with the “View Setup and Configuration” permission can use the PermissionSetLicense object to view the set of currently defined permission set licenses in your organization.

Use the PermissionSetLicense object to query existing permission licenses.

For example, to return a list of all active permission set licenses:

```
SELECT MasterLabel
FROM PermissionSetLicense
WHERE Status = 'Active'
```

When combined with the PermissionSetLicenseAssign object, you can create a nested query that returns all users assigned to a particular permission set license like “Identity Connect”:

```
SELECT MasterLabel, (SELECT AssigneeId FROM PermissionSetLicenseAssignments)
FROM PermissionSetLicense
WHERE MaximumPermissionsIdentityConnect=true
```

See Also:

PermissionSetLicenseAssign

PermissionSetLicenseAssign

Represents the association between a User and a PermissionSetLicense. This object is available in API version 29.0 and later.

Note: The relationship name for PermissionSetLicenseAssign is `PermissionSetLicenseAssignments`.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`AssigneeId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the User to assign the permission set license specified in `PermissionSetLicenseId`.
`PermissionSetLicenseId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the permission set license the user is assigned to.

Usage

Use the PermissionSetLicenseAssign object for querying permission set license assignments to find out which permission set licenses are assigned to which users. Because each user can be assigned to many permission set licenses, each PermissionSetLicenseAssign ID represents the association of a single user and single permission set license.

For example, to search for all of the permission sets assigned to a particular user:

```
SELECT Id, PermissionSetLicenseId
FROM PermissionSetLicenseAssign
WHERE AssigneeId = '005D0000001RFek'
```

To search for all users assigned to a particular permission set license:

```
SELECT AssigneeId
FROM PermissionSetLicenseAssign
WHERE PermissionSetLicenseId = '0PLD000000003mwOAA'
```

You can also create a new permission set license assignment, or use delete to remove a permission set license that's been assigned to a user. To update an assignment, delete an existing assignment and insert a new one.

See Also:

PermissionSetLicense

Pricebook2

Represents a price book that contains the list of products that your organization sells.

Note: In this release, price books are represented by Pricebook2 objects. The Pricebook object is no longer available for use.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`Description`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Text description of this object.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field	Details
	Description Indicates whether this object is active (`true`) or not (`false`). Inactive objects are hidden in many areas in the user interface. You can change this field's value as often as necessary. Label is **Active**.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsStandard`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this object is the standard price book for the organization (`true`) or not (`false`). Every organization has one standard price book—all other price books are custom price books.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of this object. This field is read-only for the standard price book. Label is **Price Book Name**.

Usage

A price book is a list of products that your organization sells:

- Each organization has one standard price book that defines the standard or generic list price for each product or service that it sells.
- An organization can have multiple custom price books that can be used for specialized purposes, such as a discount price book, price books for different channels or markets, price books for select accounts or opportunities, and so on. While your client application can create, delete, and update and custom price books, your client application can only update the standard price book.
- For some organizations, the standard price book might be the only price needed, but if you need to set up further price books, you can reference the standard price book when setting up list prices in custom price books.

Use this object to query standard and custom price books that have been configured for your organization. A common use of this object is to allow your client application to obtain valid Pricebook2 object IDs for use when configuring PricebookEntry records via the API.

Your client application can perform the following tasks on PricebookEntry objects:

- Query
- Create for the standard pricebook or custom pricebooks.
- Update
- Delete
- Change the `IsActive` field when creating or updating records

PriceBook2, Product2, and PricebookEntry Relationships

In the API:

- Price books are represented by Pricebook2 objects (the Pricebook object is no longer available).
- Products are represented by Product2 objects (the Product object is not available as of version 8.0.).
- Each price book contains zero or more entries (represented by PricebookEntry records) that specify the products that are associated with the price book. A price book entry defines the price for which you sell a product at a particular currency.

These objects are defined only for those organizations that have products enabled as a feature. If the organization does not have the products feature enabled, the Pricebook2 object does not appear in the `describeGlobal()` call, and you can't access it via the API.

If you delete a Pricebook2, while a line item references PricebookEntry in the price book, the line item is unaffected, but the Pricebook2 will be archived and unavailable from the API.

For a visual diagram of the relationships between Pricebook2 and other objects, see Product and Schedule Objects.

Price Book Setup

The process of setting up a price book via the API usually means:

1. Initially loading product data from your organization into Product2 objects (creating a Product2 record for each product that you want to add).
2. For each Product2 object, creating a PricebookEntry that links the Product2 object to the standard Pricebook2. You need to define a standard price for a product at a given currency (if you have multicurrency enabled), before defining a price for that product in the same currency in a custom price book.
3. Creating a custom Pricebook2.
4. Querying the Pricebook2 object to obtain their IDs.
5. For each Pricebook2 object, creating a PricebookEntry for every Product2 that you want to add, specifying unique properties for each PricebookEntry (such as the `UnitPrice` and `CurrencyIsoCode`) as needed.

Code Sample—Java

```
public void pricebookSample() {
  try {
    //Create a custom pricebook
    Pricebook2 pb = new Pricebook2();
    pb.setName("Custom Pricebok");
    pb.setIsActive(true);
    SaveResult[] saveResults = connection.create(new SObject[]{pb});
    pb.setId(saveResults[0].getId());

    // Create a new product
    Product2 product = new Product2();
    product.setIsActive(true);
    product.setName("Product");
    saveResults = connection.create(new SObject[]{product});
```

```
    product.setId(saveResults[0].getId());

    // Add product to standard pricebook
    QueryResult result = connection.query(
        "select Id from Pricebook2 where isStandard=true"
    );
    SObject[] records = result.getRecords();
    String stdPbId = records[0].getId();

    // Create a pricebook entry for standard pricebook
    PricebookEntry pbe = new PricebookEntry();
    pbe.setPricebook2Id(stdPbId);
    pbe.setProduct2Id(product.getId());
    pbe.setIsActive(true);
    pbe.setUnitPrice(100.0);
    saveResults = connection.create(new SObject[]{pbe});

    // Create a pricebook entry for custom pricebook
    pbe = new PricebookEntry();
    pbe.setPricebook2Id(pb.getId());
    pbe.setProduct2Id(product.getId());
    pbe.setIsActive(true);
    pbe.setUnitPrice(100.0);
    saveResults = connection.create(new SObject[]{pbe});
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

See Also:

Product and Schedule Objects

Pricebook2History

Represents historical information about changes that have been made to the standard fields of the associated Pricebook2, or to any custom fields with history tracking enabled. This object is available in API version 31.0 and later.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

This object is always read-only.

Fields

Field	Details
`Pricebook2Id`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Pricebook2 associated with this record.

Field	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Name of the price book field that was modified, or a special value to indicate some other modification to the price book.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field. Label is **Deleted**.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** New value of the modified price book field. Maximum of 255 characters.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** Previous value of the modified price book field. Maximum of 255 characters.

Usage

Price book history entries are indirectly created each time a price book is modified.

Two rows are added to this record when foreign key fields change. One row contains the foreign key object names that display in the online application. For example, `Jane Doe` is recorded as the name of a Contact. The other row contains the actual foreign key ID that is only returned to and visible from the API.

This object respects field level security on the parent object.

See Also:

Pricebook2

PricebookEntry

Represents a product entry (an association between a Pricebook2 and Product2) in a price book.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`CurrencyIsoCode`	**Type** picklist **Properties** Filter, Restricted picklist **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this object is active (`true`) or not (`false`). Although you can never delete PricebookEntry records, your client application can set this flag to `false`. Inactive PricebookEntry objects are hidden in many areas in the user interface. You can change this flag on a PricebookEntry record as often as necessary.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`Name`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Name of this PricebookEntry object. This read-only field references the value in the Name field of the Product2 object. Label is **Product Name**.
`Pricebook2Id`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Pricebook2 object with which this object is associated. This field must be specified when creating Pricebook2 records. It can't be changed in an update.
`Product2Id`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the Product2 object with which this object is associated. Required field. This field must be specified when creating Product2 records. It can't be changed in an update.
`ProductCode`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Product code for this object. This read-only field references the value in the **ProductCode** field of the associated Product2 record.
`UnitPrice`	**Type** currency **Properties** Create, Filter, Sort, Update **Description** Required. Unit price for this object. You can specify a value only if `UseStandardPrice` is set to `false`. Label is **List Price**.
`UseStandardPrice`	**Type** boolean

Field	Details
	Properties Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this object uses the standard price defined in the standard Pricebook2 object (`true`) or not (`false`). If set to `true`, then the `UnitPrice` field is read-only, and the value will be the same as the `UnitPrice` value in the corresponding PricebookEntry in the standard price book (that is, the PricebookEntry object whose `Pricebook2Id` refers to the standard price book and whose `Product2Id` and `CurrencyIsoCode` are the same as this object). For PricebookEntry objects associated with the standard Pricebook2 object, this field must be set to `true`.

Usage

Use this object to define the association between your organization's products (Product2) and your organization's standard price book or to other, custom-defined price books (Pricebook2). Create one PricebookEntry record for each standard or custom price and currency combination for a product in a Pricebook2.

When creating these records, you must specify the IDs of the associated Pricebook2 object and Product2 object. Once created, your client application can't update these IDs.

This object is defined only for those organizations that have products enabled as a feature. If the organization does not have the products feature enabled, then the PricebookEntry object does not appear in the `describeGlobal()` call, and you can't access it.

If you delete a PricebookEntry while a line item references it, the line item is unaffected, but the PricebookEntry will be archived and unavailable from the API.

You must load the standard price for a product before you are permitted to load its custom price(s).

See Also:

Standard and Custom Object Basics

ProcessDefinition

Represents the definition of a single approval process.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Portal and Communities users can't access this object.

Fields

Field	Details
`Description`	**Type** string

Field	Details
	Properties Filter, Nillable, Sort **Description** A description of this process, with a maximum of 3,000 characters.
DeveloperName	**Type** string **Properties** Filter, Group, Sort **Description** The external name of the process; the name seen by users.
LockType	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The type of lock applied to the record being approved. When a record is in the approval process, it's always locked, and only an administrator can edit it. However, the currently assigned approver can also be allowed to edit the record. • Total • Admin • Owner • Workitem • Node • none
Name	**Type** string **Properties** Filter, Group, Sort **Description** The unique process name, used internally.
State	**Type** picklist **Properties** Filter, Group, Sort

Field	Details
	Description The current state of this process. • Active • Inactive • Obsolete
`TableEnumOrId`	**Type** picklist **Properties** Filter, Group, Sort **Description** Reserved for future use.
`Type`	**Type** picklist **Properties** Filter, Group, Sort **Description** The type of this process. • Approval Process—Used to control the action taken for a record. • State-based Process—Used internally to track various control processes, such as for developing Salesforce Knowledge articles.

Usage

Use this object to read the description of an approval process. The definition is read-only.

ProcessInstance

Represents an instance of a single, end-to-end approval process. Use this and the node, step, and workitem process instance objects to create approval history reports.

Note: Exceptions apply to approval history data retrieved with this object and available only via the SOAP API. For each approval process instance that was pending when Summer '14 became available for your organization, some field values are never populated or are populated only after the approval process instance is next acted upon—such as when a user approves, rejects, or reassigns an approval request—after the Summer '14 rollout.

For approval process instances that were completed before the Summer '14 rollout, all Process Instance fields are automatically populated, with one exception: `CompletedDate` is never populated for approval process instances that were completed before January 1, 2013. For approval process instances that were pending during the Summer '14 rollout, all ProcessInstance fields are automatically populated, with two exceptions: `CompletedDate` and `LastActorId` are populated only after the approval process instance is complete.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Portal and Communities users can't access this object.

Fields

Field	Details
`CompletedDate`	**Type** dateTime **Properties** Filter, Sort **Description** The completion date and time of the approval process. The `ElapsedTimeDay`, `ElapsedTimeHours`, and `ElapsedTimeMinutes` field values are calculated using `CompletedDate`.
`ElapsedTimeInDays`	**Type** double **Properties** Filter, Sort **Description** The total elapsed time in days between when the approval process instance was started and now.
`ElapsedTimeInHours`	**Type** double **Properties** Filter, Sort **Description** The total elapsed time in hours between when the approval process instance was started and now.
`ElapsedTimeInMinutes`	**Type** double **Properties** Filter, Sort **Description** The total elapsed time in minutes between when the approval process instance was started and now.

Field	Details
LastActorId	**Type** reference **Properties** Group, Filter, Sort **Description** The last actor that approved, rejected, or recalled the process.
ProcessDefinitionId	**Type** reference **Properties** Defaulted on create, Group, Filter, Sort **Description** The ID of this approval process instance.
Status	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The status of this approval process instance, for example Started, Pending, or Approved.
TargetObjectId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the object affected by this approval process instance.

Usage

Use this object to query or retrieve an approval process.

The following SOQL query returns details for all the ProcessInstanceStep records related to individual ProcessInstance records. The nested query references `Steps`, which is the child `relationshipName` for ProcessInstanceStep in the ProcessInstance object.

```
SELECT Id, (SELECT Id, StepStatus, Comments FROM Steps)
FROM ProcessInstance
```

The following SOQL query returns details for all the ProcessInstanceWorkitem records related to individual ProcessInstance records. The nested query references `Workitems`, which is the child `relationshipName` for ProcessInstanceWorkitem in the ProcessInstance object.

```
SELECT Id, (SELECT Id, ActorId, ProcessInstanceId FROM Workitems)
FROM ProcessInstance
```

ProcessInstanceHistory can help provide a unified read-only view of the ProcessInstanceStep and ProcessInstanceWorkitem objects.

See Also:

ProcessInstanceHistory
ProcessInstanceStep
ProcessInstanceWorkitem

ProcessInstanceHistory

This read-only object shows all steps and pending approval requests associated with an approval process (ProcessInstance).

Supported Calls

`describeSObjects()`

Special Access Rules

Portal and Communities users can’t access this object.

Fields

Field	Details
`ActorId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the user who is currently assigned to this ProcessInstance.
`Comments`	**Type** string **Properties** Filter, Nillable, Sort **Description** Comments for a ProcessInstanceStep. This field doesn't apply to ProcessInstanceWorkitem records.

Field	Details
`ElapsedTimeInDays`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total time in days between when the approval process instance was started and when it was completed.
`ElapsedTimeInHours`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total time in hours between when the approval process instance was started and when it was completed.
`ElapsedTimeInMinutes`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total time in minutes between when the approval process instance was started and when it was completed.
`IsPending`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the ProcessInstance is pending (`true`) or not (`false`).
`OriginalActorId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the user who was originally assigned this ProcessInstance.

Field	Details
ProcessInstanceId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the ProcessInstance.
RemindersSent	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Number of reminders that have been sent. Default is 0 (zero).
StepStatus	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Indicates the current status of the ProcessInstanceStep.
TargetObjectId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the object being approved.

Usage

This object helps you replicate the related list functionality of the Salesforce.com user interface for approval processes. Use ProcessInstanceHistory for a unified read-only view of the ProcessInstanceStep and ProcessInstanceWorkitem objects. You can't queryProcessInstanceHistory. Instead, you can query ProcessInstanceHistory by including it in a nested query on the parent ProcessInstance object. For example, the following SOQL query returns all the ProcessInstanceHistory records related to individual ProcessInstance records. The nested query references `StepsAndWorkitems`, which is the child `relationshipName` for ProcessInstanceHistory in the ProcessInstance object.

```
SELECT Id, (SELECT Id, StepStatus, Comments FROM StepsAndWorkitems)
FROM ProcessInstance
```

This object respects field-level security on the parent object.

See Also:

ProcessInstance
ProcessInstanceStep
ProcessInstanceWorkitem

ProcessInstanceStep

Represents one work item in an approval process (ProcessInstance).

Note: Exceptions apply to approval history data retrieved with this object and available only via the SOAP API. For each approval process instance that was pending when Summer '14 became available for your organization, some field values are never populated or are populated only after the approval process instance is next acted upon—such as when a user approves, rejects, or reassigns an approval request—after the Summer '14 rollout.

ProcessInstanceStep fields are never populated for approval process instances that were completed before the Summer '14 rollout. For approval process instances that were pending during the Summer '14 rollout, all ProcessInstanceStep fields are populated only after the approval process instance is next acted upon after the Summer '14 rollout.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Portal and Communities users can't access this object.

Fields

Field	Details
`ActorId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the user who is currently assigned to this approval step.
`Comments`	**Type** string **Properties** Filter, Nillable, Sort **Description** Limit: 4,000 bytes.

Field	Details
`ElapsedTimeInDays`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total time in days since this step was started.
`ElapsedTimeInHours`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total time in hours since this step was started.
`ElapsedTimeInMinutes`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total time in minutes since this step was started.
`OriginalActorId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the user who was originally assigned to this approval step.
`ProcessInstanceId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the ProcessInstance that this approval step belongs to.
`StepNodeId`	**Type** reference

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** ID of the node currently assigned to this approval step.
`StepStatus`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The current status of this approval step. • Approved • Fault • Held • NoResponse • Pending • Reassigned • Rejected • Removed • Started If the approval step requires unanimous approval and one approver rejects the request, the value of this field for the other approvers changes to NoResponse. Likewise, if approval is based on the first response and an approver responds, the value of this field for the other approvers changes to NoResponse.

Usage

Query or retrieve a new step in an approval process (ProcessInstance).

See Also:

ProcessInstance
ProcessInstanceHistory
ProcessInstanceWorkitem

ProcessInstanceNode

Represents a step in an instance of an approval process. Compare to ProcessNode, which describes the step in a process definition. Use this object to retrieve approval history.

Note: Exceptions apply to approval history data retrieved with this object and available only via the SOAP API. For each approval process instance that was pending when Summer '14 became available for your organization, some field values are never populated or are populated only after the approval process instance is next acted upon—such as when a user approves, rejects, or reassigns an approval request—after the Summer '14 rollout.

ProcessInstanceNode fields are never populated for approval process instances that were completed before the Summer '14 rollout. For approval process instances that were pending during the Summer '14 rollout, all ProcessInstanceNode fields are populated only after the approval process instance is next acted upon after the Summer '14 rollout.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Portal and Communities users can't access this object.

Fields

Field	Details
`CompletedDate`	**Type** dateTime **Properties** Filter, Sort **Description** The completion date and time of this step in the approval process. The `ElapsedTimeDay, ElapsedTimeHours`, and `ElapsedTimeMinutes` field values are calculated using `CompletedDate`.
`ElapsedTimeInDays`	**Type** double **Properties** Filter, Sort **Description** The total time in days since this step was started.
`ElapsedTimeInHours`	**Type** double **Properties** Filter, Sort **Description** The total time in hours since this step was started.
`ElapsedTimeInMinutes`	**Type** double **Properties** Filter, Sort

Field	Details
	Description The total time in minutes since this step was started.
`LastActorId`	**Type** reference **Properties** Defaulted on create, Group, Filter, Sort **Description** The last actor that approved or rejected this step.
`NodeStatus`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The status of this approval instance, for example Started, Pending, or Approved.
`ProcessInstanceId`	**Type** reference **Properties** Filter, Group, Sort **Description** The approval process this step is part of.
`ProcessNodeId`	**Type** reference **Properties** Filter, Group, Sort **Description** The identifier for this step.
`ProcessNodeName`	**Type** string **Properties** Filter, Group, Sort **Description** The name of this step.

ProcessInstanceWorkitem

Represents a user's pending approval request.

Note: Exceptions apply to approval history data retrieved with this object and available only via the SOAP API. For each approval process instance that was pending when Summer '14 became available for your organization, some field values are never populated or are populated only after the approval process instance is next acted upon—such as when a user approves, rejects, or reassigns an approval request—after the Summer '14 rollout.

ProcessInstanceWorkitem fields are never populated for approval process instances that were completed before the Summer '14 rollout. For approval process instances that were pending during the Summer '14 rollout, all ProcessInstanceWorkitem fields are populated after the approval process instance is next acted upon after the Summer '14 rollout, with three exceptions: `ElapsedTimeInDays`, `ElapsedTimeInDays`, and `ElapsedTimeInMinutes` fields are never populated in ProcessInstanceWorkitem records for which equivalent ProcessInstanceStep records were created before the Summer '14 rollout.

For all other ProcessInstanceWorkitem records, these three fields are populated after the approval process instance is next acted upon after the Summer '14 rollout.

Note: Because ProcessInstanceHistory combines fields from ProcessInstanceStep and ProcessInstanceWorkitem, you may notice incorrect elapsed times of 0 in ProcessInstanceHistory records because the elapsed time fields were never populated in the related ProcessInstanceWorkitem record.

Supported Calls

`delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`

Special Access Rules

Portal and Communities users can't access this object.

Fields

Field	Details
`ActorId`	**Type** reference **Properties** Filter, Group, Sort, Update **Description** ID of the user who is currently responsible for approving an approval request.
`ElapsedTimeInDays`	**Type** double **Properties** Filter, Nillable **Description** The total time in days since this approval request was started.

Field	Details
`ElapsedTimeInHours`	**Type** double **Properties** Filter, Nillable **Description** The total time in hours since this approval request was started.
`ElapsedTimeInMinutes`	**Type** double **Properties** Filter, Nillable **Description** The total time in minutes since this approval request was started.
`OriginalActorId`	**Type** reference **Properties** Filter, Group, Sort, Update **Description** ID of the user who was originally assigned this approval request.
`ProcessInstanceId`	**Type** reference **Properties** Filter, Group, Sort, Update **Description** ID of the ProcessInstance associated with this approval request.

Usage

Use this object to manage a pending approval request for a user.

See Also:

ProcessInstance
ProcessInstanceHistory
ProcessInstanceStep

ProcessNode

Describes a step in a process definition. Compare to ProcessInstanceNode, which describes the step in a running process.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Portal and Communities users can't access this object.

Fields

Field	Details
`Description`	**Type** string **Properties** Filter, Nillable, Sort **Description** A description of this node, no longer than 3,000 bytes.
`DeveloperName`	**Type** string **Properties** Filter, Group, Sort **Description** The external name of the node; the name seen by users.
`Name`	**Type** string **Properties** Filter, Group, Sort **Description** The unique node name.
`ProcessDefinition`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the object affected by this approval instance.

Usage

Use this object to get the description of a process definition.

The following SOQL query returns details for all the ProcessInstanceStep records related to individual ProcessInstance records. The nested query references `Steps`, which is the child `relationshipName` for ProcessInstanceStep in the ProcessInstance object.

```
SELECT Id, (SELECT Id, StepStatus, Comments FROM Steps)
FROM ProcessInstance
```

The following SOQL query returns details for all the ProcessInstanceWorkitem records related to individual ProcessInstance records. The nested query references `Workitems`, which is the child `relationshipName` for ProcessInstanceWorkitem in the ProcessInstance object.

```
SELECT Id, (SELECT Id, ActorId, ProcessInstanceId FROM Workitems)
FROM ProcessInstance
```

ProcessInstanceHistory can help provide a unified read-only view of the ProcessInstanceStep and ProcessInstanceWorkitem objects.

Product2

Represents a product that your organization sells.

This object has several fields that are used only for quantity and revenue schedules (for example, annuities). Schedules are available only for organizations that have enabled the products and schedules features. If these features aren't enabled, the schedule fields don't appear in the DescribeSObjectResult, and you can't query, create, or update the fields.

Note: Products are represented by Product2 objects. The Product object is no longer available.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`CanUseQuantitySchedule`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the product can have a quantity schedule (`true`) or not (`false`). Label is **Quantity Scheduling Enabled**.
`CanUseRevenueSchedule`	**Type** boolean

Field	Details
	Properties Defaulted on create, Filter, Group, Sort **Description** Indicates whether the product can have a revenue schedule (`true`) or not (`false`). Label is **Revenue Scheduling Enabled**.
`ConnectionReceivedId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that shared this record with your organization. This field is only available if you have enabled Salesforce to Salesforce.
`ConnectionSentId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that you shared this record with. This field is only available if you have enabled Salesforce to Salesforce. Beginning with API version 15.0, the `ConnectionSentId` field is no longer supported. The `ConnectionSentId` field is still visible, but the value is null. You can use the new PartnerNetworkRecordConnection object to forward records to connections.
`CurrencyIsoCode`	**Type** picklist **Properties** Filter, Restricted picklist **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.
`DefaultPrice`	**Type** currency **Properties** Create, Filter, Update **Description** The default price for this record.

Field	Details
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A text description of this record. Label is **Product Description**.
`Family`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Name of the product family associated with this record. Product families are configured as picklists in the user interface. To obtain a list of valid values, call `describeSObjects()` and process the DescribeSObjectResult for the values associated with the `Family` field. Label is **Product Family**.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this record is active (`true`) or not (`false`). Inactive Product2 records are hidden in many areas in the user interface. You can change the `IsActive` flag on a Product2 object as often as necessary. Label is **Active**.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update

Field	Details
	Description Required. Default name of this record. Label is **Product Name**.
NumberOfQuantityInstallments	**Type** int **Properties** Filter, Group, Sort **Description** If the product has a quantity schedule, the number of installments.
NumberOfRevenueInstallments	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** If the product has a revenue schedule, the number of installments.
ProductCode	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Default product code for this record. The product code naming pattern is defined by your organization.
QuantityInstallmentPeriod	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** If the product has a quantity schedule, the amount of time covered by the schedule.
QuantityScheduleType	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of the quantity schedule, if the product has one.

Field	Details
RecalculateTotalPrice	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Changes behavior of OpportunityLineItem calculations when a line item has child schedule rows for the `Quantity` value. When enabled, if the rollup quantity changes, then the quantity rollup value is multiplied against the sales price to change the total price.
RevenueInstallmentPeriod	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** If the product has a revenue schedule, the period of time covered by the schedule.
RevenueScheduleType	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of the revenue schedule, if the product has one.

Schedule Enabled Flags

When enabling the schedules feature, organizations can decide whether to enable quantity schedules, revenue schedules, or both. In addition, you can use the API to control quantity and revenue scheduling at the product level via the `CanUseQuantitySchedule` and `CanUseRevenueSchedule` flags. A value of `true` for either flag indicates that the product and any OpportunityLineItems can have a schedule of that type. These flags can be set when creating or updating Product2 records.

Default Schedule Fields

The remaining schedule fields for this object define default schedules. Default schedule values are used to create an OpportunityLineItemSchedule when an OpportunityLineItem is created for the Product.

The default schedule fields support the following valid values (all fields are also nillable).

Field	Valid Values
RevenueScheduleType	Divide, Repeat
RevenueInstallmentPeriod	Daily, Weekly, Monthly, Quarterly, Yearly
NumberOfRevenueInstallments	Integer between 1 to 150, inclusive.

Field	Valid Values
`QuantityScheduleType`	Divide, Repeat
`QuantityInstallmentPeriod`	Daily, Weekly, Monthly, Quarterly, Yearly
`NumberOfQuantityInstallments`	Integer between 1 to 150, inclusive

When you attempt to set the schedule fields when creating or updating, the API applies cross-field integrity checks. The integrity requirements are:

- If the schedule type is nil, the installment period and number of installments must be nil.
- If the schedule type is set to any value, then the installment period and number of installments must be non-nil.

Any create or update that fails these integrity checks is rejected with an error.

These default schedule fields, as well as `CanUseQuantitySchedule` and `CanUseRevenueSchedule`, are restricted picklist fields and are available only if the organization has the schedules feature enabled.

Usage

Use this object to define the default product information for your organization. This object is associated by reference with Pricebook2 objects via PricebookEntry objects. The same product can be represented in different price books as price book entries. In fact, the same product can be represented multiple times (as separate PricebookEntry records) in the same price book with different prices or currencies. A product can only have one price for a given currency within the same price book. To be used in custom price books, all standard prices must be added as price book entries to the standard price book.

You can query the products that have been configured for your organization. For example, you can allow your client application to obtain valid product IDs for use when configuring PricebookEntry records via the API. Your client application can perform the following tasks on PricebookEntry objects:

- Query
- Create for the standard pricebook or custom pricebooks.
- Update
- Delete
- Change the `IsActive` field when creating or updating records

This object is defined only for those organizations that have products enabled as a feature. If the organization does not have the products feature, this object does not appear in the `describeGlobal()` call, and you can't describe or query this object.

If you try to delete a product via the API but there is an opportunity that uses that product, the delete fails. The workaround is to delete the product in the user interface, which gives you an option to archive the product.

Note: On opportunities and opportunity products, the workflow rules, validation rules, and Apex triggers fire when an update to a child opportunity product or schedule causes an update to the parent record. This means your custom application logic is enforced when there are updates to the parent record, ensuring higher data quality and compliance with your organization's business policies.

See Also:

Standard and Custom Object Basics

Product2Feed

Represents a single feed item in the feed displayed on the detail page for a product record.

A product feed shows recent changes to a product record for any fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to products in Salesforce. This object is available in API version 18.0 and later.

Note: Products are represented by Product2 objects. The Product object is no longer available.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the Product2 object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of Product2Feed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string

Field	Details
	Properties Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization.

Field	Details
	`NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the product record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the feed item. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field	Details
	Description The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.

Field	Details
Visibility	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a product record.

See Also:

Product2
EntitySubscription
NewsFeed
UserProfileFeed

ProductEntitlementTemplate

Represents predefined terms of customer support (Entitlement) that users can add to products (Product2).

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
EntitlementTemplateId	**Type** reference

Field	Details
	Properties Create, Filter, Group, Sort **Description** Required. ID of the entitlement template. Must be a valid ID.
`Product2Id`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the Product2 associated with the entitlement template. Must be a valid ID.

Usage

Use to query and manage entitlement templates.

See Also:

Entitlement

Profile

Represents a profile, which defines a set of permissions to perform different operations, such as querying, adding, updating, or deleting information.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `update()`

Special Access Rules

- Customer Portal users can't access this object.
- Partner portal users can't access this object.

Fields

Field	Details
`Description`	**Type** string **Properties** Filter, Group, Nillable, Sort, Update

Field	Details
	Description Description of the profile.
`IsSsoEnabled`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** If `true`, users assigned to this profile can delegate username and password authentication to a corporate database instead of the user database.
`LastReferencedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The timestamp for when the current user last viewed a record related to this profile. Available in API version 29.0 and later.
`LastViewedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The timestamp for when the current user last viewed this profile. Available in API version 29.0 and later.
`Name`	**Type** string **Properties** Filter, Group, Sort, Update **Description** The name of the profile.
`Permissions`*`PermissionName`*	**Type** boolean **Properties** Filter, Update

Field	Details
	Description One field for each permission. If `true`, users assigned to this profile have the named permission. The number of fields varies depending on the permissions for the organization and license type. **Tip:** To get a list of available permissions in the SOAP API, use `describeSObjects()`.
`UserLicenseId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the UserLicense associated with this profile.
`UserType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The category of user license. Each `UserType` is associated with one or more UserLicense records. Each UserLicense is associated with one or more profiles. In API version 10.0 and later, valid values include: • Standard: user license. This user type also includes Salesforce Platform and Salesforce Platform One user licenses. Label is **Standard**. • PowerPartner: User whose access is limited because he or she is a partner and typically accesses the application through a partner portal or community. Label is **Partner**. • CSPLitePortal: user whose access is limited because he or she is an organization's customer and accesses the application through aCustomer Portal or community. Label is **High Volume Portal**. • CustomerSuccess: user whose access is limited because he or she is an organization's customer and accesses the application through a Customer Portal. Label is **Customer Portal User**. • PowerCustomerSuccess: user whose access is limited because he or she is an organization's customer and accesses the application through a Customer Portal. Label is **Customer Portal Manager**. Users with this license type can view and edit data they directly own or data owned by or shared with users below them in the Customer Portal role hierarchy. • CsnOnly: user whose access to the application is limited to Chatter. This user type includes Chatter Free and Chatter moderator users. Label is **Chatter Free**.

Field	Details
	`UserType` replaces `LicenseType`, which is unavailable as of API version 10.0. In API versions 8.0 and 9.0 `LicenseType` is still available with the following valid values: • AUL: Force.com user license. Label is **Apex Platform**. • AUL1: Force.com user license with only one user. Label is **Apex Platform One**. • Salesforce: Salesforce user license. Label is **Salesforce**. • PackageManager: user who can create and work with managed packages for Force.com AppExchange. Label is **Package Manager**. • PRM: user whose access is limited because he or she is a partner and typically accesses the application through a partner portal. Label is **Partner**. • CustomerUser: user whose access is limited because he or she is an organization's customer and accesses the application through a Customer Portal. Label is **Customer Portal User**. • CustomerManager: user whose access is limited because he or she is an organization's customer and accesses the application through a Customer Portal. Label is **Customer Portal Manager**. Users with this license type can view and edit data they directly own or data owned by or shared with users below them in the Customer Portal role hierarchy.

Usage

Use the Profile object to query the set of currently configured user profiles in your organization. Your client application can use Profile objects to obtain valid profile IDs for use when querying or modifying users through the API.

In the user interface, profiles can be used to assign user licenses from specific pools (Force.com Platform user license or Salesforce user license, for example). If a user is assigned to a profile with a different license type, the number of available licenses in the old license type pool increases, one per user changed, and decreases by the same amount in the new license type pool.

See Also:

Standard and Custom Object Basics
PermissionSet

ProfileSkill

Represents a profile skill, which describes a user's professional knowledge. This is a global record for the organization, and users are associated through the ProfileSkillUser object.

Note: For information about Live Agent skills, see the Skill topic.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
`Description`	**Type** textarea **Properties** Create, Nillable, Update **Description** Description of the profile skill.
`LastReferencedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The timestamp indicating when the current user last viewed a record related to this profile skill. Available in API version 29.0 and later.
`LastViewedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The timestamp indicating when the current user last viewed this profile skill. Available in API version 29.0 and later.
`Name`	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update **Description** The name of the profile skill.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The owner of the profile skill.

Field Name	Details
UserCount	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of users with the profile skill.

Usage

Use the ProfileSkill object to look up the attributes of a skill that can be assigned to a user. This is a global object and is not owned by any specific user.

ProfileSkillEndorsement

Represents a detail relationship of ProfileSkillUser. An endorsement of a profile skill shows approval and support of another user's publicly declared skill.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
Name	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** The name of the profile skill being endorsed.
ProfileSkillUserId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the ProfileSkillUser record that is being endorsed.
UserId	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The user ID of the person giving the endorsement.

Usage

Use the ProfileSkillEndorsement object to query about a single endorsement given to a user about a specific skill. Users can't endorse themselves, they can only be endorsed by others unless they are administrators with the "Modify All Data" permission.

ProfileSkillEndorsementHistory

Represents the history of changes to the fields of a ProfileSkillEndorsement.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The profile skill endorsement changed field.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the profile skill endorsement field.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The old value of the profile skill endorsement field.

Field Name	Details
ProfileSkillEndorsementId	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the profile skill endorsement.

Usage

Use this read-only object to identify changes to a ProfileSkillEndorsement.

ProfileSkillFeed

Represents a single feed item in the feed displayed on the detail page for a profile skill.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
Body	**Type** textarea **Properties** Nillable, Sort **Description** The content of the ProfileSkillFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
CommentCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
ContentData	**Type** base64

Field Name	Details
	Properties Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.

Field Name	Details
InsertedById	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
LikeCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the object type to which the FeedItem object is related. For example, set this field to a `UserId` to post to someone's profile feed, or an `AccountId` to post to a specific account.
RelatedRecordId	**Type** reference **Properties** Group, Nillable, Sort

Field Name	Details
	Description ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as a FeedItem object of `Type` `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the FeedItem. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of ProfileSkillFeed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold.

Field Name	Details
	• DashboardComponentSnapshot—created when a user posts a dashboard snapshot on a feed. • LinkPost—a post with an attached URL. • PollPost—a poll posted on a feed. • ProfileSkillPost—generated when a skill is added to a user's Chatter profile. • QuestionPost—generated when a user posts a question. • ReplyPost—generated when Chatter Answers posts a reply. • RypplePost—generated when a user creates a Thanks badge in Work.com. • TextPost—a direct text entry on a feed. • TrackedChange—a change or group of changes to a tracked field. • UserStatus—automatically generated when a user adds a post. Deprecated. The following values appear in the Type picklist for all feed objects but apply only to CaseFeed: • AttachArticleEvent—generated event when a user attaches an article to a case. • CallLogPost—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • CaseCommentPost—generated event when a user adds a case comment for a case object. • ChangeStatusPost—generated event when a user changes the status of a case. • ChatTranscriptPost—generated event when Live Agent transcript is saved to a case. • EmailMessageEvent—generated event when an email related to a case object is sent or received. • FacebookPost—generated when a Facebook post is created from a case. Deprecated. • MilestoneEvent—generated when a case milestone is completed or reaches violation status. • SocialPost—generated when a social post is created from a case. **Note:** If you set Type to ContentPost, you must specify ContentData and ContentFileName.

Usage

A ProfileSkill feed shows recent changes to a ProfileSkill record for any fields that are tracked in feeds, as well as comments and posts about the record It is a useful way to stay up-to-date with changes to solutions. Use this object to track changes for a ProfileSkill record.

ProfileSkillHistory

Represents the history of changes to the fields of a ProfileSkill.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The value of the field before it was changed.
`ProfileSkillId`	**Type** reference **Properties** Filter, Group, Sort **Description** Required. ID of the ProfileSkill.

Usage

Use this read-only object to identify changes to a ProfileSkill.

ProfileSkillShare

Represents a sharing entry on a ProfileSkill.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Level of access that the User or Group has to the ProfileSkill. The possible values are: • `Read` • `Edit` • `All` (This value is not valid for `create()` or `update()` calls.) This value must be set to an access level that is higher than the organization's default access level for ProfileSkill objects.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the parent object, if any.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Values may include: • `Manual`—The User or Group has access because a user with "All" access manually shared the ProfileSkill with them. • `Owner`—The User is the owner of the ProfileSkill or is in a role above the ProfileSkill owner in the role hierarchy.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort

Field Name	Details
	Description ID of the User or Group that has been given access to the ProfileSkill.

Usage

This object is read only. It is visible because of constraints to the ProfileSkill object, but it is ignored and does not control which users and groups can view and edit ProfileSkill records owned by other users.

ProfileSkillUser

Represents a detail relationship of User. The object connects profile skills with users.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
`EndorsementCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of endorsements.
`Name`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** The name of the skill user.
`ProfileSkillId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the ProfileSkill.

Field Name	Details
UserId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the user. This field can't be changed once it is created.

Usage

Use this object to assign specific skills to specific users. ProfileSkillUser appears on the Overview tab on the Chatter profile page. Users can only create a skill mapping for themselves, they can't create skill mappings for others unless they are administrators with the "Modify All Data" permission. Additionally, users can only edit this object if they are the context user and are not editing the `UserId` field.

ProfileSkillUserHistory

Represents the history of changes to the fields of a ProfileSkillUser.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
Field	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
NewValue	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
OldValue	**Type** anyType

Field Name	Details
	Properties Nillable, Sort **Description** The latest value of the field before it was changed.
`ProfileSkillUserId`	**Type** reference **Properties** Filter, Group, Sort **Description** Required. ID of the ProfileSkillUser.

Usage

Use this read-only object to identify changes to a ProfileSkillUser.

PushTopic

Represents a query that is the basis for notifying listeners of changes to records in an organization. This is available from API version 21.0 or later.

Supported Calls

REST: DELETE, GET, PATCH, POST (query requests are specified in the URI)

SOAP: `create()`, `delete()`, `describe()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`

Special Access Rules

- This object is only available if Streaming API is enabled for your organization.
- Only users with “Create” permission can create this record. Users with “View All Data” can view PushTopic records and see streaming messages.

Fields

Field	Field Type	Description
`ApiVersion`	double	Required. API version to use for executing the query specified in `Query`. It must be an API version greater than 20.0. If your query applies to a custom object from a package, this value must match the package's `ApiVersion`. Example value: 31.0 **Field Properties:** Create, Filter, Sort, Update
`Description`	string	Description of the PushTopic. Limit: 400 characters **Field Properties:** Create, Filter, Sort, Update

Field	Field Type	Description
ID	ID	System field: Globally unique string that identifies a record. **Field Properties:** Default on create, Filter, Group, idLookup, Sort
isActive	boolean	Indicates whether the record currently counts towards the organization's limit. **Field Properties:** Create, Default on create, Filter, Group, Sort, Update
IsDeleted	boolean	System field: Indicates whether the record has been moved to the Recycle Bin (true) or not (false). **Field Properties:** Default on create, Filter, Group, Sort
Name	string	Required. Descriptive name of the PushTopic, such as MyNewCases or TeamUpdatedContacts. Limit: 25 characters. This value identifies the channel. **Field Properties:** Create, Filter, Group, Sort, Update
NotifyForFields	picklist	Specifies which fields are evaluated to generate a notification. Valid values: • All • Referenced (default) • Select • Where **Field Properties:** Create, Filter, Sort, Update
NotifyForOperations	picklist	Specifies which record events may generate a notification. Valid values: • All (default) • Create • Extended • Update **Field Properties for API version 28.0 and earlier:** Create, Filter, Sort, Update **Field Properties for API version 29.0 and later:** Filter, Sort In API version 29.0 and later, this field is read-only, and will not contain information about delete and undelete events. Use NotifyForOperationCreate, NotifyForOperationDelete, NotifyForOperationUndelete and NotifyForOperationUpdate to specify which record events should generate a notification. A value of Extended means that neither create or update operations are set to generate events.
NotifyForOperationCreate	boolean	true if a create operation should generate a notification, otherwise, false. Defaults to true. This field is available in API version 29.0 and later.
NotifyForOperationDelete	boolean	true if a delete operation should generate a notification, otherwise, false. Defaults to true. Clients must connect using the cometd/29.0 (or later) Streaming API endpoint to receive delete and undelete event notifications. This field is available in API version 29.0 and later.
NotifyForOperationUndelete	boolean	true if an undelete operation should generate a notification, otherwise, false. Defaults to true. Clients must connect using the cometd/29.0 (or later)

Field	Field Type	Description
		Streaming API endpoint to receive delete and undelete event notifications. This field is available in API version 29.0 and later.
`NotifyForOperationUpdate`	boolean	`true` if an update operation should generate a notification, otherwise, `false`. Defaults to `true`. This field is available in API version 29.0 and later.
`Query`	string	Required. The SOQL query statement that determines which record changes trigger events to be sent to the channel. Limit: 1300 characters **Field Properties:** Create, Filter, Sort, Update

`PushTopic` and Notifications

The PushTopic defines when notifications are generated in the channel. This is specified by configuring the following PushTopic fields:

- PushTopic Queries
- Events
- Notifications

QuantityForecast

Represents a quantity-based forecast.

The API also provides revenue-based forecasts using RevenueForecast.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- Requires the “View All Data” permission.
- Customer Portal users can’t access this object.

Fields

Field	Details
`Closed`	**Type** double **Properties** Filter, Nillable, Sort **Description** Read-only. A rollup of opportunities or opportunity line items that have closed in this period.

Field	Details
Commit	**Type** double **Properties** Filter, Nillable, Sort **Description** The owner's Commit total.
CommitComment	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Read-only. The comment entered when the owner edited his or her **Commit** total from the **Adjusted Total** link on the forecast edit page.
CommitOverride	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Read-only. The owner's override of their own **My Commit** total.
DefaultRollupCommit	**Type** double **Properties** Filter, Nillable, Sort **Description** Read-only. The owner's standard **Commit** rollup, including their own opportunities and forecast-level overrides from subordinate users in the role hierarchy.
DefaultRollupUpside	**Type** double **Properties** Filter, Nillable, Sort **Description** Read-only. The owner's standard **Best Case** rollup, including their own opportunities and forecast-level overrides from subordinate users in the role hierarchy.

Field	Details
InvalidationDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Read-only. If not blank, indicates that the rollup numbers for **Forecast Override** fields that represent calculated (summarized) amounts may not be up to date.
ManagerChoiceCommit	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Read-only. The manager's choice regarding the commit amount: • DefaultRollup: Use the manager's default commit rollup for owner's forecast, which reflects the manager's opportunity forecast overrides. • AcceptForecast (default selection): Accept the forecast owner's **Adjusted Total** commit amount, which may or may not be an override. • ManagerManualOverride: Use the manager's manual override. • OpportunityOnlyRollup: Use the opportunity rollup, including opportunity forecast overrides, but excluding any forecast-level (**Adjusted Total**) overrides.
ManagerChoiceUpside	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Read-only. The manager's choice regarding the best case amount: • DefaultRollup: Use the manager's default best case rollup for owner's forecast, which reflects the manager's opportunity forecast overrides. • AcceptForecast (default selection): Accept the forecast owner's **Adjusted Total** best case amount, which may or may not be an override. • ManagerManualOverride: Use the manager's manual override. • OpportunityOnlyRollup: Use the opportunity rollup, including opportunity forecast overrides, but excluding any forecast-level (**Adjusted Total**) overrides.
ManagerClosed	**Type** double

Field	Details
	Properties Filter, Nillable, Sort **Description** Read-only. The manager's closed total for the owner's forecast, including any opportunity or opportunity product overrides made by the manager.
ManagerCommit	**Type** double **Properties** Filter, Nillable, Sort **Description** The manager's Commit total.
ManagerCommitOverride	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Read-only. The manager's manual override of the forecast owner's **Commit** total. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.
ManagerDefaultRollupCommit	**Type** double **Properties** Filter, Nillable, Sort **Description** Read-only. The manager's standard **Commit** rollup for the forecast owner. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.
ManagerDefaultRollupUpside	**Type** double **Properties** Filter, Nillable, Sort **Description** Read-only. The manager's standard **Best Case** rollup for the forecast owner. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.

Field	Details
ManagerId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Read-only. ID of the direct manager of the user who owns this forecast.
ManagerOpportunityRollupCommit	**Type** double **Properties** Filter, Nillable, Sort **Description** Read-only. The manager's view of the forecast owner's opportunity-level **Commit** rollup, ignoring all forecast overrides. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.
ManagerOpportunityRollupUpside	**Type** double **Properties** Filter, Nillable, Sort **Description** Read-only. The manager's view of the forecast owner's opportunity-level **Best Case** rollup, ignoring all forecast overrides. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.
ManagerPipeline	**Type** double **Properties** Filter, Nillable, Sort **Description** The manager's pipeline total for the owner's forecast, including any opportunity or opportunity product overrides made by the manager.
ManagerUpside	**Type** double **Properties** Filter, Nillable, Sort

Field	Details
	Description The manager's Best Case total.
`ManagerUpsideOverride`	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** The manager's manual override of the forecast owner's **Best Case** total. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.
`OpportunityRollupClosed`	**Type** double **Properties** Filter, Nillable, Sort **Description** The owner's **Closed** total for his or her opportunities only.
`OpportunityRollupCommit`	**Type** double **Properties** Filter, Nillable, Sort **Description** The owner's **Commit** total for his or her opportunities only.
`OpportunityRollupPipeline`	**Type** double **Properties** Filter, Nillable, Sort **Description** The owner's **Pipeline** total for his or her opportunities only.
`OpportunityRollupUpside`	**Type** double **Properties** Filter, Nillable, Sort **Description** The owner's **Best Case** total for his or her opportunities only.

Field	Details
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** ID of the User who owns this forecast. Required on create.
`PeriodId`	**Type** reference **Properties** Filter, Nillable, Group, Sort **Description** The ID of the Period that contains the `StartDate`.
`Pipeline`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total pipeline rollup from subordinates in the role hierarchy, including the owner's opportunities.
`ProductFamily`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** The value chosen in the Product Family picklist, which can be configured from Setup at **Customize** > **Products** > **Fields**. This field is relevant if you have chosen "Use Product Families" as the Forecast Type from Setup at **Customize** > **Forecast** > **Settings**. If you are not forecasting by product family or if the forecast represents opportunities that are not associated with a product family, then this field is blank. Required on create.
`Quota`	**Type** double **Properties** Create, Filter, Nillable, Sort, Update

Field	Details
	Description The quota amount for the period. You can create, update, upsert, and delete. Requires the “Modify All Data” and “Manage Users” permission. Required on create.
StartDate	**Type** date **Properties** Create, Filter, Group, Nillable, Sort **Description** The start date of this forecast. The ID of the Period that contains this date is written to the PeriodId field if it changes. A new Period is created if none exists. Required on create.
Upside	**Type** currency **Properties** Filter, Nillable **Description** The owner's **Best Case** total.
UpsideComment	**Type** string **Properties** Filter, Nillable, Update **Description** The comment entered when the owner edited his or her **Best Case** total.
UpsideOverride	**Type** double **Properties** Filter, Nillable, Update **Description** Read only. The owner's override of their own My **Best Case** total.

Usage

Query this object to support customizable forecasts based on quantities.

You can update one field, `Quota`, which means that you can mass update sales users' quotas instead of updating them one by one in the user interface. Editing `Quota`, which means that you can mass update sales users' quotas instead of updating them one by one in the user interface. Editing `Quota` requires the “Modify All Data” and “Manage Users” permissions.

The rollup fields always reflect opportunity and opportunity product overrides by the forecast owner or one of the forecast owner's subordinates in the role hierarchy. In addition, the manager rollup fields include overrides by the forecast owner's direct manager in the role hierarchy.

Some of the rollup fields ignore forecast-level (**Adjusted Total**) overrides, but they never ignore opportunity forecast overrides that are visible to the owner or manager.

See Also:

RevenueForecast
QuantityForecastHistory

QuantityForecastHistory

Represents historical information about quantity-based forecasts that have been submitted (saved) in the user interface.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

- Requires the "View All Data" permission.
- Customer Portal users can't access this object.

Fields

Field	Details
`Closed`	**Type** double **Properties** Filter, Nillable, Sort **Description** The closed amount of the forecast.
`Commit`	**Type** double **Properties** Filter, Nillable, Sort **Description** The commit amount of the forecast.
`CommitComments`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Comments about the commit value.
CommitOverridden	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the commit value was overridden (`true`) or not (`false`).
CurrencyIsoCode	**Type** picklist **Properties** Filter, Restricted picklist **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.
ForecastOverrideId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the related forecast override.
Pipeline	**Type** double **Properties** Filter, Nillable, Sort **Description** The pipeline amount of the forecast.
Quota	**Type** currency **Properties** Filter, Nillable **Description** The quota amount of the forecast.

Field	Details
Upside	**Type** currency **Properties** Filter, Nillable **Description** The best case amount of the forecast.
UpsideComments	**Type** string **Properties** Filter, Nillable **Description** Comments about the upside value.
UpsideOverridden	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the upside value was overridden (`true`) or not (`false`).

Usage

This is a read-only object specific to customizable forecasting.

When a user submits a revenue-based forecast in the user interface, a new record is created. If the same forecast is ever resubmitted, additional records are added. The `CreatedDate` of a record reflects the day on which the forecast was submitted. This object respects field level security on the parent object.

See Also:

QuantityForecast
RevenueForecastHistory

Question

Represents a question in a community that users can view and reply to.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
BestReplyId	**Type** reference **Properties** Filter, Group, Nillable, Sort, Update **Description** The ID of the Reply that has been identified as the best answer to the question. You must use the user interface to identify the best answer for a question.
BestReplySelectedById	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the User who selected the best answer to the question. This field is available in API version 24.0 and later. In API version 24.0 through version 29.0, you must update this field using the UI. In API version31.0 and later, you can update this field using the API.
Body	**Type** textarea **Properties** Create, Nillable, Update **Description** Description of the question.
CommunityId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The zone ID associated with the question. Once you create a question, you can't change the zone ID associated with that question.
CreatorFullPhotoUrl	**Type** string **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description URL of the user's profile photo from the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.
`CreatorName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Name of the user who posted the question or reply. Only the first name of internal users (agents) appears to portal users in the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.
`CreatorSmallPhotoUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** URL of the user's thumbnail photo from the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.
`HasSingleFieldForContent`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the content of a Chatter Answers question is: • Included in only one field: `Title` if the content is unformatted and less than 255 characters; or `Body` if the content is formatted or more than 255 characters (`true`) • Included in two fields: `Title` and `Body` (`false`) This field also determines if content displays in one or two fields in Chatter Answers question feeds. This field is available in API version 25.0 and later.
`LastReferencedDate`	**Type** date

Field	Details
	Properties Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
LastReplyDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date and time the last reply (child Reply object) was posted.
LastReplyId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Read only. The ID of the last reply (child Reply object) posted to the question.
LastViewedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
MostReportAbusesOnReply	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The most number of user-reported abuses on a Reply associated with the question. This field is available in API version 24.0 and later.

Field	Details
`NumReplies`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The number of replies (child Reply object) that users have submitted for the question.
`NumReportAbuses`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Represents the number of user-reported abuses on the question. This field is available in API version 24.0 and later.
`NumSubscriptions`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Represents the number of users following the question. This field is available in API version 24.0 and later.
`Origin`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The source of the question, such as `Chatter Answers`. This field is available in API version 24.0 and later.
`Title`	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update

Field	Details
	Description The descriptive title of the question.
`UpVotes`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The total number of up votes for the question.
`VoteScore`	**Type** double **Properties** Filter , Nillable, Sort **Description** The internal score of the question, used to sort questions and articles on the Popular tab in the application user interface. The internal algorithm that determines the score gives older votes less weight than newer votes, simulating exponential decay. The score itself does not display in the application user interface. **Note:** Unlike other fields of type double, you can't use a SOQL aggregate function with this field.

Usage

Use this object to track questions in azone.

QuestionDataCategorySelection

A data category selection represents a data category that classifies a question. This object can be used to associate a question with a data category from a data category group or to query the categorization for a question.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

To create, read or update data category selection, you must have create, read or update permission on the categorized question. Users who can update question can also delete its category selection. Users who can create questions can only select categories visible to their role.

Fields

Field Name	Details
DataCategoryGroupName	**Type** DataCategoryGroupReference **Properties** Create **Description** Unique name of the data category group which has a category associated with the question.
DataCategoryName	**Type** DataCategoryGroupReference **Properties** Create **Description** Unique name of the data category associated with the question.
ParentId	**Type** reference **Properties** Create, Filter **Description** ID of the question associated with the data category selection.

Usage

Every question can be categorized in a data category. You can use the QuestionDataCategorySelection object to query and manage question categorization. Client applications can create categorization for a question. They can also delete, query, and retrieve question categorization.

Warning: Even though the API lets you select more than one category for QuestionDataCategorySelection, the Answers tab only supports one data category selection for questions. Selecting multiple categories through QuestionDataCategorySelection may result in unexpected behavior in the Answers tab, such as losing your multiple selections. You should only select one data category when using QuestionDataCategorySelection.

Sample Code—Java

In the following example, the `selectCategory` method adds a category to a question data category selection. The `retrieveCategorySelections` method returns all the categories from a question data category selection.

```
public void selectCategory(ID parentId, String categoryGroupName, String categoryName) {
    try {
        QuestionDataCategorySelection categorySelection = new
QuestionDataCategorySelection();

        categorySelection.setParentId(parentId);
```

```
        categorySelection.setDataCategoryGroupName(categoryGroupName);
        categorySelection.setDataCategoryName(categoryName);
        binding.create(new SObject[]{categorySelection});
    } catch (RemoteException e) {
        System.out.println("An unexpected error has occurred." + e.getMessage());
    }
}

public String[] retrieveCategorySelections(String parentId) {
    QueryResult qr = null;

    try {
        qr = binding.query("SELECT DataCategoryName FROM QuestionDataCategorySelection
WHERE Id = '" + parentId + "'");
    } catch (RemoteException e) {
        System.out.println("An unexpected error has occurred." + e.getMessage());
    }

    String[] categoryNames = new String[qr.getRecords().length];
    for (int index = 0; index < qr.getRecords().length; index++) {
        categoryNames[index] =
((QuestionDataCategorySelection)qr.getRecords()[index]).getDataCategoryName();
    }

    return categoryNames;
}
```

Salesforce Knowledge uses a similar object for article data category selection. See `Article Type`__DataCategorySelection for SOQL examples using this object.

See Also:

Article Type__DataCategorySelection
Data Categories

QuestionReportAbuse

Represents a user-reported abuse on a Question in a Chatter Answers community. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Name`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** The name of the Question from which the user reported abuse.

Field	Details
QuestionId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the Question from which the user reported abuse.
Reason	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** The reason the user reported abuse on the Question, such as `Spam`, `Hateful`, or `Inappropriate`.

Usage

Use this object to track user-reported abuse on questions created in a Chatter Answers community.

QuestionSubscription

Represents a subscription for a user following a Question. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
CommunityId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Required. ID of the zone associated with the Question the user is following. This field can’t be updated.
Name	**Type** string

Field	Details
	Properties Autonumber, Defaulted on create, Filter, Sort **Description** The name of the question subscription.
`QuestionCreatedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Required. Creation date of the Question which the user is following. This field can't be updated.
`QuestionId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** Required. ID of the Question which the user is following. This field can't be updated.
`SubscriberId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** Required. ID of the User who is following the Question. This field can't be updated.

Usage

Things to consider when following a Question:

- A user can only follow questions that they have permission to view.
- Administrators and users with the "Modify All Data" permission can configure other users to follow questions that the other user has read access to.
- Administrators and users with the "Modify All Data" permission can configure users to stop following questions.

Queries on QuestionSubscription:

- Users with the "Read" permission on Question can see which questions other users are following.
- A query must include a LIMIT clause and the limit can't exceed 1,000.
- A query using a `WHERE` clause can only filter by fields on Question.

QueueSobject

Represents the mapping between a queue Group and the sObject types associated with the queue, including custom objects.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`QueueId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of a queue.
`SobjectType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** A list of object types that can be associated with the queue specified by the `QueueId`.

Usage

Use this object to associate a queue with the sObject that can be associated with the queue, including custom objects.

A queue is a Group whose `Type` is `Queue`.

See Also:

Standard and Custom Object Basics

QuickText

This object stores a snippet of text that allows an agent to send a quick response to a customer in the Live Agent console. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
`Category`	**Type** picklist **Properties** Create, Filter, Group, Sort, Update **Description** A customizable picklist that can be used to group multiple related Quick Text records together
`Channel`	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** A multi-select picklist that can be used to specify where specific Quick Text messages should be available, such as in Live Agent or in the Email publisher in Case Feed.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.

Field Name	Details
Message	**Type** textarea **Properties** Create, Filter (unavailable in API version 25.0 and later), Sort (unavailable in API version 25.0 and later), Update **Description** The content of the Quick Text record
Name	**Type** string **Properties** Create, Filter (unavailable in API version 25.0 and later), Group, Sort (unavailable in API version 25.0 and later), Update **Description** A descriptive label for the Quick Text record
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the User who owns the Quick Text record

Usage

Use this object to create and manage the Quick Text messages available to agents in the Live Agent console. You can categorize multiple QuickText records into groups using the Category field. The Category field can also be a parent to multiple custom dependent Picklist fields to create a hierarchical structure of categories.

QuickTextHistory

Represents changes to field values on a QuickText object. This object is available in API version 24.0 and later.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
Field	**Type** picklist

Field Name	Details
	Properties Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The old value of the field that was changed
`QuickTextId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the QuickText that was changed

Usage

Use this object to identify changes to a QuickText.

QuickTextOwnerSharingRule

Represents a rule for sharing a QuickText object with users other than the owner.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
AccessLevel	**Type** picklist **Properties** Create, Group, Sort, Update **Description** A value that represents the type of sharing allowed. The possible values are: • `Read` • `Edit`
Description	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
DeveloperName	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
GroupID	**Type** reference **Properties** Create, Filter, Group, Sort

Field Name	Details
	Description The ID representing the source group. QuickText objects owned by users in the source group trigger the rule to give access.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to **Label** in the user interface.
`UserorGroupID`	**Type** reference **Properties** Create, Filter **Description** The ID representing the target user or group. Target users or groups have access to the QuickText.

Usage

Use this object to manage the sharing rules for a QuickText object.

QuickTextShare

Represents a sharing entry on a QuickText object. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Level of access that the User or Group has to the QuickText. The possible values are:

Field Name	Details
	• `Read` • `Edit` • `All` (This value is not valid for `create()` or `update()` calls.) This value must be set to an access level that is higher than the organization's default access level for QuickText objects.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the parent object, if any
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Values may include: • `Manual`—The User or Group has access because a user with "All" access manually shared the QuickText with them. • `Owner`—The User is the owner of the QuickText or is in a role above the QuickText owner in the role hierarchy.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the User or Group that has been given access to the QuickText

Usage

This object lets you determine which users and groups can view and edit QuickText records owned by other users.

If you attempt to create a new record that matches an existing record, the `create()` call updates any modified fields and returns the existing record.

Quote

The Quote object represents a quote, which is a record showing proposed prices for products and services. Quotes can be created from and synced with opportunities, and emailed as PDFs to customers

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`AccountId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the account that's associated with the quote.
`AdditionalAddress` (beta)	**Type** address **Properties** Filter, Nillable **Description** Compound form of the additional address. Read-only. See Address Compound Fields for details on compound address fields.
`AdditionalCity`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** City for the quote's additional address. Up to 40 characters allowed.
`AdditionalCountry`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Country for the quote's additional address. Up to 80 characters allowed.
`AdditionalCountryCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description ISO country code for the quote’s additional address.
`AdditionalLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `AdditionalLongitude` to specify the precise geolocation of an additional address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`AdditionalLongitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `AdditionalLatitude` to specify the precise geolocation of an additional address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`AdditionalName`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Name associated with the quote's additional address. Limited: 255 characters.
`AdditionalPostalCode`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Postal Code for the quote's additional address.
`AdditionalState`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description State for the quote's additional address. Up to 80 characters allowed.
`AdditionalStateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ISO state code for the quote's additional address.
`AdditionalStreet`	**Type** textarea **Properties** Create, Filter, Nillable, Group, Sort, Update **Description** Street name for the quote's additional address.
`BillingAddress` (beta)	**Type** address **Properties** Filter, Nillable **Description** Compound form of the billing address. Read-only. See Address Compound Fields for details on compound address fields.
`BillingCity`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** City for the quote's billing address. Up to 40 characters allowed.
`BillingCountry`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Country for the quote's billing address. Up to 80 characters allowed.

Field	Details
BillingCountryCode	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ISO country code for the quote’s billing address.
BillingLatitude (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with BillingLongitude to specify the precise geolocation of a billing address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
BillingLongitude (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with BillingLatitude to specify the precise geolocation of a billing address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
BillingName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Entity that the quote is billed to.
BillingPostalCode	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Postal Code for the quote's billing address.

Field	Details
BillingState	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** State for the quote's billing address. Up to 80 characters allowed.
BillingStateCode	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ISO state code for the quote’s billing address.
BillingStreet	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Street name for the quote's billing address.
ContactId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the contact that’s associated with the quote.
ContractId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the contract that’s associated with the quote.
CurrencyIsoCode	**Type** picklist

Field	Details
	Properties Filter, Restricted picklist **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization. If the organization has multicurrency and a `Pricebook2Id` specified on the quote, then the currency value of this field must match the currency of the PricebookEntry objects that are associated with any quote line items it has. This value is copied from the related Opportunity and can't be changed.
`Description`	**Type** textarea **Properties** Nillable **Description** Text description of the quote. Limit: 32,000 characters.
`Discount`	**Type** percent **Properties** Filter, Nillable, Sort **Description** Difference between the sum of the QuoteLineItem record's`Subtotal` and the sum of the QuoteLineItem record's `Discount` totals. Expressed as a percentage.
`Email`	**Type** email **Properties** Filter, Group, Nillable, Sort **Description** Email address of the contact associated with the quote.
`ExpirationDate`	**Type** date **Properties** Create, Filter, Nillable, Update

Field	Details
	Description Date this quote is no longer valid.
`Fax`	**Type** phone **Properties** Create, Filter, Nillable, Update **Description** Fax number for the contact associated with the quote.
`GrandTotal`	**Type** currency **Properties** Filter, Nillable **Description** Total price of the quote plus shipping and taxes.
`IsSyncing`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the quote is syncing with an opportunity.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update

Field	Details
	Description The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`LineItemCount`	**Type** int **Properties** Filter, Nillable **Description** Number of line items on the quote.
`Name`	**Type** string **Properties** Create, Filter, idLookup, Update **Description** Required. Name for the quote. Limit: 225 characters.
`OpportunityId`	**Type** reference **Properties** Create, Filter **Description** ID for the opportunity associated with the quote.
`Phone`	**Type** phone **Properties** Create, Filter, Nillable, Update **Description** Phone number of the contact associated with the quote.
`Pricebook2Id`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the price book associated with the quote.

Field	Details
`QuoteNumber`	**Type** string **Properties** Defaulted on create, Filter **Description** Automatically-generated number identifying the quote.
`QuoteToAddress` (beta)	**Type** address **Properties** Filter, Nillable **Description** Compound form of the quote to address. Read-only. See Address Compound Fields for details on compound address fields.
`QuoteToCity`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** City for the address to send the quote to for approval, such as a third party-agency representing a buyer. Up to 40 characters allowed.
`QuoteToCountry`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Country for the address to send the quote to for approval. Up to 80 characters allowed.
`QuoteToLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `QuoteToLongitude` to specify the precise geolocation of a quote to address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.

Field	Details
QuoteToLongitude (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with QuoteToLatitude to specify the precise geolocation of a quote to address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
QuoteToName	**Type** string **Properties** Create, Filter, Nillable, Update **Description** Name of the entity (such as a person or business) the quote is sent to for approval. Limit: 255 characters.
QuoteToPostalCode	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Postal code for the address to send the quote to for approval.
QuoteToState	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** State for the address to send the quote to for approval. Up to 80 characters allowed.
QuoteToStreet	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Street name for the address to send the quote to for approval.

Field	Details
RecordTypeID	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the record type assigned to the object.
ShippingAddress (beta)	**Type** address **Properties** Filter, Nillable **Description** Compound form of the shipping address. Read-only. See Address Compound Fields for details on compound address fields.
ShippingCity	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** City for the quote's shipping address. Up to 40 characters allowed.
ShippingCountry	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Country for the quote's shipping address. Up to 80 characters allowed.
ShippingCountryCode	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ISO country code for the quote’s shipping address.
ShippingHandling	**Type** currency

Field	Details
	Properties Create, Filter, Nillable, Update **Description** Total shipping and handling costs for the quote.
`ShippingLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `ShippingLongitude` to specify the precise geolocation of a shipping address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`ShippingLongitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `ShippingLatitude` to specify the precise geolocation of an address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`ShippingName`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** Name of the entity (such as a person or business) the quote is sent to for approval.
`ShippingPostalCode`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Postal code for the quote's shipping address.
`ShippingState`	**Type** string

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** State for the quote's shipping address. Up to 80 characters allowed.
`ShippingStateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ISO state code for the quote’s shipping address.
`ShippingStreet`	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Street name for the quote's shipping address.
`Status`	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** Current status of the quote. The standard options are: • —None— • Draft • Needs Review • In Review • Approved • Rejected • Presented • Accepted • Denied
`Subtotal`	**Type** currency **Properties** Filter, Nillable

Field	Details
	Description Line item's sales price multiplied by the quantity minus the discount.
`Tax`	**Type** currency **Properties** Create, Filter, Nillable, Update **Description** Total taxes for the quote.
`TotalPrice`	**Type** currency **Properties** Filter, Nillable **Description** Total of the quote line items after discounts and before taxes and shipping.

Usage

Use Quote to manage proposed product prices for customers. To update a Quote, your client application needs "Edit" permission.

- Client applications can create, update, delete, and query Attachment records associated with a quote via the API.
- You can sync a quote and its parent Opportunity.

See Also:

QuoteLineItem
QuoteDocument
Opportunity

QuoteDocument

The QuoteDocument object represents a quote in document format.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`CurrencyIsoCode`	**Type** picklist

Field	Details
	Properties Filter, Restricted picklist **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization. If the organization has multicurrency and a `Pricebook2Id` specified on the quote, then the currency value of this field must match the currency of the PricebookEntry objects that are associated with any quote line items it has.
`Discount`	**Type** percent **Properties** Create, Filter, Nillable **Description** The discount for the quote used in the document.
`Document`	**Type** base64 **Properties** Create, Nillable **Description** The binary data of the document stored in the QuoteDocument object.
`GrandTotal`	**Type** currency **Properties** Create, Filter, Nillable **Description** Grand total for the quote used in the document.
`Name`	**Type** string **Properties** Filter, idLookup **Description** Name of the quote document.

Field	Details
QuoteId	**Type** reference **Properties** Create, Filter **Description** ID for the quote used for the document.

Usage

Use the QuoteDocument object to store a document that can be used to present the quote information to the customer.

See Also:

Quote
QuoteLineItem

QuoteLineItem

The QuoteLineItem object represents a quote line item, which is a member of the list of Product2 products associated with a Quote, along with other information about those line items on that quote.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

The user must have "Edit" permissions on Quote records in order to create or update quote line items on a quote. The user must have "Edit" permissions on Quote records to delete a quote line item.

Fields

Field	Details
CurrencyIsoCode	**Type** picklist **Properties** Filter, Restricted picklist **Description** Available only for organizations enabled for multiples currencies. Contains the ISO code for any currency allowed by the organization. If the organization has multicurrency and a Pricebook2 is specified on the quote (the `Pricebook2Id` field is not blank), then the currency value of this field must match the currency of the PricebookEntry objects for any associated quote line items.

Field	Details
	This value is copied from the related Quote and can't be changed.
`Description`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Text description of the line item. Limit: 225 characters.
`Discount`	**Type** percent **Properties** Filter, Nillable, Sort **Description** Editable number from 0 to 100.
`HasQuantitySchedule`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Read-only. Indicates whether the opportunity line item that the quote line item is synced with has a quantity schedule.
`HasRevenueSchedule`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Read-only. Indicates whether the opportunity line item that the quote line item is synced with has a revenue schedule. If this object has a revenue schedule, the `GrandTotal` and `TotalPrice` fields can't be updated. In addition, the `Quantity` field can't be updated if this object has a quantity schedule. The system ignores any attempt to update this field. The update isn't rejected but the updated value is ignored.
`HasSchedule`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Read-only. Indicates whether the line item uses schedules.
`LineNumber`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, Sort **Description** Read-only. Automatically-generated number identifying the quote line item. In the form of `QL-XXXXX`.
`ListPrice`	**Type** currency **Properties** Filter, Nillable, Sort **Description** Read-only. Corresponds to the `UnitPrice` on the PricebookEntry that is associated with this line item, which can be in the standard price book or a custom price book. A client application can use this information to show whether the unit price (or sales price) of the line item differs from the price book entry list price.
`PricebookEntryId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the associated PricebookEntry. Exists only for those organizations that have Products enabled as a feature.
`Product2Id`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** ID of the Product2 associated with this QuoteLineItem.
`Quantity`	**Type** double **Properties** Create, Filter, Sort, Update

Field	Details
	Description The number of units for the line item.
`QuoteId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. ID of the associated Quote.
`ServiceDate`	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** Date when the product revenue will be recognized and the product quantity will be shipped.
`SortOrder`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The value of where the line item is in the sorted order, such as 1, 2, and so on. The SortOrder value determines the order in which a quote line item displays in the Quote Detail page. Client applications can use this to match the sort order in Salesforce. This field is only available in API versions 21.0 and greater.
`Subtotal`	**Type** currency **Properties** Filter, Nillable **Description** The line item's `Quantity` multiplied by the `UnitPrice`.
`TotalPrice`	**Type** currency **Properties** Filter, Nillable

Field	Details
	Description Read-only. Calculated by applying the `Discount` to the `Subtotal`. If you don't specify a `UnitPrice`, this field is required. This field is nillable, but you can't set both `TotalPrice` and `UnitPrice` to null in the same update. To insert the `TotalPrice` for a quote line item via the API (given only a unit price and the quantity), calculate this field as the unit price multiplied by the quantity. This field is read-only if the quote line item has a revenue schedule. If it doesn't have a revenue schedule or only has a quantity schedule, this field can be updated.
`UnitPrice`	**Type** currency **Properties** Create, Filter, Update **Description** The price per unit for the quote line item.

Usage

A Quote record can have QuoteLineItem records only if the Quote has a Pricebook2. A QuoteLineItem must correspond to a Product2 that is listed in the quote's Pricebook2.

Note: If the multicurrency option has been enabled, the `CurrencyIsoCode` field is present. It can't be modified, it is always set to the value of the `CurrencyIsoCode` of the parent Quote.

Effects on Quotes

Quotes that have associated QuoteLineItem objects are affected in the following ways:

- Creating a QuoteLineItem increments the Quote value by the `TotalPrice` of the QuoteLineItem.
- When you create or update a QuoteLineItem, the API verifies that the line item corresponds to a PricebookEntry in the Pricebook2 that is associated with the quote.

See Also:

Quote
QuoteDocument
Opportunity

RecentlyViewed

Represents records that the current user has recently viewed or referenced (by viewing a related record).

Supported Calls

`query()`, `update()`

Special Usage Rules

The RecentlyViewed object does not support the Report, KnowledgeArticle, and Article objects.

Fields

Field	Details
Alias	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The alias on the record.
Email	**Type** email **Properties** Filter, Group, Nillable, Sort **Description** The email address on the record.
FirstName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The first name on the record. If the recently viewed record is a user, this is the user's first name.
Id	**Type** ID **Properties** Defaulted on create, Filter, Group, Sort **Description** The ID of the recently viewed record.
IsActive	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the recently viewed record is an active user (`true`) or not (`false`). This field contains a value only if the recently viewed record is a user.

Field	Details
LastName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The last name on the record.
LastReferencedDate	**Type** dateTime **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
LastViewedDate	**Type** dateTime **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (LastReferencedDate) and not viewed.
Name	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The name on the recently viewed record. If the recently viewed record is a user, contact, or lead, the value is a concatenation of the firstname and lastname field values.
NetworkId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the community that this group is part of. This field is available only if Salesforce Communities is enabled in your organization.

Field	Details
	You can add a `NetworkId` only when creating a group. You can't change or add a `NetworkId` for an existing group. This field is available in API version 27.0 and later.
`Phone`	**Type** phone **Properties** Filter, Group, Nillable, Sort **Description** The phone number on the record.
`ProfileId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** If the recently viewed record is a user, this is the user's profile ID.
`Title`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** If the recently viewed record is a user, this is the title of the user; for example CFO or CEO.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The sObject type for this recently viewed record. Valid values include any standard or custom objects that RecentlyViewed supports.
`UserRoleId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description The ID of the user role associated with this object.

Usage

This object provides a heterogeneous list of different object types and consists of recently viewed records or records that were recently referenced (a related record was viewed). A record is considered viewed when the user sees the details associated with it, but not when the user sees it in a list with other records. Use this object to programmatically construct a list of recently viewed items specific to the current user, for example, on a custom user interface or for search auto-complete options. You can also retrieve a filtered list of records by object type (`Type`). The RecentlyViewed data is periodically truncated down to 200 records per object.

Use this query in your code to retrieve a list of all the records that were recently viewed. The results are ordered from most to least recent.

```
SELECT Id, Name
FROM RecentlyViewed
WHERE LastViewedDate !=null
ORDER BY LastViewedDate DESC
```

Use this query to retrieve data that was either viewed or referenced, but only for a limited set of objects.

```
SELECT Id, Name
FROM RecentlyViewed
WHERE Type IN ('Account', 'Contact', 'Plan__c')
ORDER BY LastViewedDate DESC
```

This query retrieves a list of all recently viewed contacts with contact-specific fields, such as the contact's account name, and the custom website field. Records are ordered from most to least recent.

```
SELECT Account.Name, Title, Email, Phone, Website__c
FROM Contact
WHERE LastViewedDate != NULL
ORDER BY LastViewedDate DESC
```

RecordType

Represents a record type.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`BusinessProcessId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Required for Opportunity and Lead record types in API version 17.0 and later. ID of an associated BusinessProcess.
`Description`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** A description of this record. Limit: 255 characters.
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Label is **Record Type Name**. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`IsActive`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether this record is active (`true`) or not (`false`). Only active record types can be applied to records. Label is **Active**.

Field	Details
IsPersonType	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether this record has been designated as a person account (`true`) or not (`false`). Visible only if the organization has the person account feature enabled.
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Label of the record type in the user interface. Limit: 80 characters. Label is **Record Type Label**.
NamespacePrefix	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***namespacePrefix__componentName*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects. This field can't be accessed unless the logged-in user has the "Customize Application" permission.
SobjectType	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort

Field	Details
	Description Object to which this record type applies, including custom objects.

Usage

Use this object to offer different BusinessProcess records and subsets of picklist values to different users based on their Profile. Your client application can describe or query RecordType records.

The following objects have a `RecordTypeId` field:

- Account
- Campaign
- CampaignMember
- Case
- Contact
- Contract
- Lead
- Opportunity
- QuickText
- Solution
- Custom objects

Client applications can create or update values in `RecordTypeId` on these objects, specifying a valid record type ID associated with these objects.

Note: You can't create or update the `RecordTypeId` field on the CampaignMember records. Set the CampaignMember record type using the `CampaignMemberRecordTypeId` field on Campaign.

A client application can retrieve the list of valid record type IDs for a given object by querying the RecordType.

See Also:

Record Type Objects

RecordTypeLocalization

When the Translation Workbench is enabled for your organization, the RecordTypeLocalization object provides the translation of the label for a record type.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- Your organization must be using Professional, Enterprise, Developer, Unlimited, or Performance Edition and be enabled for the Translation Workbench.
- To view this object, you must have the "View Setup and Configuration" permission.

Fields

Field	Details
`Language`	**Type** string **Properties** Create, Filter, Nillable, Restricted picklist **Description** The language for this translated label.
`NamespacePrefix`	**Type** string **Properties** Filter, Nillable **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.
`ParentId`	**Type** reference **Properties** Create, Filter, Nillable **Description** The ID of the RecordType associated with the label that is being translated.
`Value`	**Type** string **Properties** Create, Filter, Nillable, Update

Field	Details
	Description The actual translated label for the record type. Label is **Translation**.

Usage

Use this object to translate the labels of your record types into other supported languages.

Reply

Represents a reply that a user has submitted to a question in an answers community.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`Body`	**Type** textarea **Properties** Create, Update **Description** Body of this reply.
`CommunityId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The zone ID associated with the question and its reply. This field is available in API version 27.0 and later.
`CreatorFullPhotoUrl`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** URL of the user's profile photo from the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.

Field	Details
CreatorName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Name of the user who posted the question or reply. Only the first name of internal users (agents) appears to portal users in the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later
CreatorSmallPhotoUrl	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** URL of the user's thumbnail photo from the feed. Chatter Answers must be enabled to view this field. This field is available in API version 26.0 and later.
DownVotes	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The total number of down votes for a reply.
Name	**Type** string **Properties** Filter, Group, idLookup, Sort **Description** When creating a Reply, the `Name` field is automatically populated with a truncated, plain text version of the Reply `Body` field.
NumReportAbuses	**Type** int **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Represents the number of reported abuses on the reply by users. This field is available in API version 24.0 and later.
`QuestionId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the Question to which this reply was made.
`UpVotes`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The total number of up votes for a reply.
`VoteTotal`	**Type** double **Properties** Filter, Nillable, Sort **Description** The total number of all votes for a reply, including up and down votes.

Usage

Use this object to track replies to a Question.

ReplyReportAbuse

Represents a user-reported abuse on a Reply in a Chatter Answers community. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
Name	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** The name of the Reply from which the user reported abuse.
Reason	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** The reason the user reported abuse on the Reply, such as `Spam`, `Hateful`, or `Inappropriate`.
ReplyId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the Reply from which the user reported abuse.

Usage

Use this object to track user-reported abuse on replies created in a Chatter Answers community.

Report

Represents a report, a set of data that meets certain criteria, displayed in an organized way. Access is read-only. This object is available in API version 20.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`, `search()`

Fields

Field	Details
Description	**Type** string

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** The description of the report. Limit: 255 characters.
`DeveloperName`	**Type** string **Properties** Filter, Group, Sort **Description** Required. The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Label is **Report Unique Name**. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`Format`	**Type** picklist **Properties** Filter, Group, Sort **Description** Required. Indicates the format of the report. This field is available in API version 31.0 and later. Can have one of these values: • Tabular for reports in that format. The label is `Tabular`. • Summary for reports in that format. The label is `Summary`. • Matrix for reports in that format. The label is `Matrix`. • Multiblock for reports in joined format. The label is `Joined`.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.

Field	Details
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastRunDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** Returns the date the report was last run. Label is **Last Run**.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`Name`	**Type** string **Properties** Filter, Group, Sort **Description** Required. The report label used in the user interface.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation.

Field	Details
	The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects. This field can't be accessed unless the logged-in user has the "Customize Application" permission.
`OwnerId`	**Type** reference **Properties** Filter **Description** The ID of the user who currently owns this report.

Usage

Use the report object to get report metadata. Query, search, or retrieve specific metadata on reports. Report object fields are read-only.

Sample

In this example, a SOQL query returns reports that contain the name "sales" and are listed by their developer names, format, ID, and report name.

```
SELECT DeveloperName,Format,Id,Name FROM Report WHERE Name LIKE '%sales%'
```

See Also:

ReportFeed
ReportTag
Dashboard

ReportFeed

Represents a single feed item in the feed displayed on a report. This object is available in API version 20.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of ReportFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable

Field	Details
	Description The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime

Field	Details
	Properties Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort

Field	Details
	Description Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`.

Field	Details
	• For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the account record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the report record. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence.

Field	Details
	• AdvancedTextPost—created when a user posts a group announcement. • AnnouncementPost—Not used. • ApprovalPost—generated when a user submits an approval. • BasicTemplateFeedItem—Not used. • CanvasPost—a post made by a canvas app posts on a feed. • CollaborationGroupCreated—generated when a user creates a public group. • CollaborationGroupUnarchived—Not used. • ContentPost—a post with an attached file. • CreatedRecordEvent—generated when a user creates a record from the publisher. • DashboardComponentAlert—generated when a dashboard metric or gauge exceeds a user-defined threshold. • DashboardComponentSnapshot—created when a user posts a dashboard snapshot on a feed. • LinkPost—a post with an attached URL. • PollPost—a poll posted on a feed. • ProfileSkillPost—generated when a skill is added to a user's Chatter profile. • QuestionPost—generated when a user posts a question. • ReplyPost—generated when Chatter Answers posts a reply. • RypplePost—generated when a user creates a Thanks badge in Work.com. • TextPost—a direct text entry on a feed. • TrackedChange—a change or group of changes to a tracked field. • UserStatus—automatically generated when a user adds a post. Deprecated. The following values appear in the Type picklist for all feed objects but apply only to CaseFeed: • CaseCommentPost—generated event when a user adds a case comment for a case object • EmailMessageEvent—generated event when an email related to a case object is sent or received • CallLogPost—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • ChangeStatusPost—generated event when a user changes the status of a case • AttachArticleEvent—generated event when a user attaches an article to a case **Note:** If you set Type to ContentPost, you must specify ContentData and ContentFileName.
Visibility	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field	Details
	Description Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this to retrieve the current contents of the feed fields, such as type of feed or feed ID.

See Also:

Report

ReportTag

Associates a word or short phrase with a Report. This object is available in API version 20.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string

Field Name	Details
	Properties Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.
`Type`	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

ReportTag stores the relationship between its parent TagDefinition and the Report being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

See Also:

Report

RevenueForecast

Represents a revenue-based forecast.

The API also provides quantity-based forecasts with QuantityForecast.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`Closed`	**Type** currency **Properties** Create, Filter **Description** Read only. A rollup of opportunities or opportunity line items that have closed in this period.
`Commit`	**Type** currency **Properties** Filter, Nillable **Description** The owner's Commit total.
`CommitComment`	**Type** string **Properties** Filter, Nillable, Update **Description** Read only. The comment entered when the owner edited his or her Commit total from the Adjusted Total link on the forecast edit page.
`CommitOverride`	**Type** currency **Properties** Filter, Nillable, Update **Description** Read only. The owner's override of their own My Commit total.
`CurrencyIsoCode`	**Type** picklist

Field	Details
	Properties Filter, Restricted picklist, Update **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization. You can update or upsert this field and `Quota` only.
`DefaultRollupCommit`	**Type** currency **Properties** Filter, Nillable **Description** Read only. The owner's standard Commit rollup, including their own opportunities and forecast-level overrides from subordinate users in the role hierarchy.
`DefaultRollupUpside`	**Type** currency **Properties** Filter, Nillable **Description** Read only. The owner's standard Best Case rollup, including their own opportunities and forecast-level overrides from subordinate users in the role hierarchy.
`InvalidationDate`	**Type** dateTime **Properties** Filter, Nillable **Description** Read only. If not blank, indicates that the rollup numbers for Forecast Override fields that represent calculated (summarized) amounts may not be up to date.
`ManagerChoiceCommit`	**Type** picklist **Properties** Filter, Nillable, Restricted picklist, Update **Description** Read only. The manager's choice regarding the commit amount:

Field	Details
	• DefaultRollup: Use the manager's default commit rollup for owner's forecast, which reflects the manager's opportunity forecast overrides. • AcceptForecast (default selection): Accept the forecast owner's Adjusted Total commit amount, which may or may not be an override. • ManagerManualOverride: Use the manager's manual override. • OpportunityOnlyRollup: Use the opportunity rollup, including opportunity forecast overrides, but excluding any forecast-level (Adjusted Total) overrides.
`ManagerChoiceUpside`	**Type** picklist **Properties** Filter, Nillable, Restricted picklist, Update **Description** Read only. The manager's choice regarding the best case amount: • DefaultRollup: Use the manager's default best case rollup for owner's forecast, which reflects the manager's opportunity forecast overrides. • AcceptForecast (default selection): Accept the forecast owner's Adjusted Total best case amount, which may or may not be an override. • ManagerManualOverride: Use the manager's manual override. • OpportunityOnlyRollup: Use the opportunity rollup, including opportunity forecast overrides, but excluding any forecast-level (Adjusted Total) overrides.
`ManagerClosed`	**Type** currency **Properties** Filter, Nillable **Description** Read only. The manager's closed total for the owner's forecast, including any opportunity or opportunity product overrides made by the manager.
`ManagerCommit`	**Type** currency **Properties** Filter, Nillable **Description** The manager's Commit total.

Field	Details
ManagerCommitOverride	**Type** currency **Properties** Filter, Nillable, Update **Description** Read only. The manager's manual override of the forecast owner's Commit total. Represents an option in the override popup window, which allows managers to choose how to roll up the forecast numbers of a direct report.
ManagerDefaultRollupCommit	**Type** currency **Properties** Filter, Nillable **Description** Read only. The manager's standard Commit rollup for the forecast owner. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.
ManagerDefaultRollupUpside	**Type** currency **Properties** Filter, Nillable **Description** Read only. The manager's standard Best Case rollup for the forecast owner. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.
ManagerId	**Type** reference **Properties** Filter, Nillable **Description** Read only. ID of the direct manager of the user who owns this forecast.
ManagerOpportunityRollupCommit	**Type** currency

Field	Details
	Properties Filter, Nillable **Description** The manager's view of the forecast owner's opportunity-level Commit rollup, ignoring all forecast overrides. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.
`ManagerOpportunityRollupUpside`	**Type** currency **Properties** Filter, Nillable **Description** The manager's view of the forecast owner's opportunity-level Best Case rollup, ignoring all forecast overrides. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.
`ManagerPipeline`	**Type** currency **Properties** Filter, Nillable **Description** The manager's pipeline total for the owner's forecast, including any opportunity or opportunity product overrides made by the manager.
`ManagerTerritoryId`	**Type** reference **Properties** Filter, Nillable **Description** ID of the manager's UserRole or Territory.
`ManagerUpside`	**Type** currency **Properties** Filter, Nillable **Description** The manager's Best Case total.

Field	Details
ManagerUpsideOverride	**Type** currency **Properties** Filter, Nillable, Update **Description** The manager's manual override of the forecast owner's Best Case total. Represents an option in the override pop-up window, which allows managers to choose how to roll up the forecast numbers of a direct report.
OpportunityRollupClosed	**Type** currency **Properties** Filter, Nillable **Description** The owner's Closed total for his or her opportunities only.
OpportunityRollupCommit	**Type** currency **Properties** Filter, Nillable **Description** The owner's Commit total for his or her opportunities only.
OpportunityRollupPipeline	**Type** currency **Properties** Filter, Nillable **Description** The owner's Pipeline total for his or her opportunities only.
OpportunityRollupUpside	**Type** currency **Properties** Filter, Nillable **Description** Read only. The owner's Best Case total for his or her opportunities only.

Field	Details
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter **Description** ID of the User who owns this forecast. Required on create.
PeriodId	**Type** reference **Properties** Filter, Nillable **Description** The ID of the Period that contains the `StartDate`.
Pipeline	**Type** currency **Properties** Filter, Nillable **Description** The total pipeline rollup from subordinates in the role hierarchy, including the owner's opportunities.
ProductFamily	**Type** picklist **Properties** Create, Filter, Nillable **Description** The value chosen in the Product Family picklist, which can be configured from Setup at **Customize** > **Products** > **Fields**. This field is relevant if you have chosen “Use Product Families” as the Forecast Type from Setup at **Customize** > **Forecast** > **Settings**. If you are not forecasting by product family or if the forecast represents opportunities that are not associated with a product family, then this field is blank. Otherwise, this field is required on create.
Quota	**Type** currency **Properties** Create, Filter, Nillable, Update

Field	Details
	Description The quota amount for the period. You can update or upsert this field and `CurrencyIsoCode`, and for Quota only, create. Requires the "Modify All Data" and "Manage Users" permission. Required on create.
`StartDate`	**Type** date **Properties** Create, Filter, Nillable **Description** The start date of this forecast. The period ID of the period that contains this date is written to the `PeriodId` field if it changes. A new Period is created if none exists. Required on create.
`TerritoryId`	**Type** reference **Properties** Create, Filter, Nillable **Description** ID of the forecast owner's UserRole or Territory. Required on create if Territory Management is enabled (if this field is available).
`Upside`	**Type** currency **Properties** Filter, Nillable **Description** The owner's Best Case total.
`UpsideComment`	**Type** string **Properties** Filter, Nillable, Update **Description** Read only. The comment entered when the owner edited his or her Best Case total. Label is **Best Case Comment**.
`UpsideOverride`	**Type** currency

Field	Details
	Properties Filter, Nillable, Update **Description** Read only. The owner's override of their My Best Case total. Label is **Best Case Override**.

Usage

Query this object to support customizable forecasts based on revenue. Requires the “View All Data” permission.

You can also update `CurrencyIsoCode` and `Quota`, which means that you can mass update sales users' quotas instead of updating them one by one in the user interface. Editing `Quota` requires “Modify All Data” and “Manage Users” permissions.

The rollup fields always reflect opportunity and opportunity product overrides by the forecast owner or one of the forecast owner’s subordinates in the role hierarchy. In addition, the manager rollup fields include overrides by the forecast owner's direct manager in the role hierarchy.

Some of the rollup fields ignore forecast-level (**Adjusted Total**) overrides, but they never ignore opportunity forecast overrides that are visible to the owner or manager.

See Also:

QuantityForecast
RevenueForecastHistory
Standard and Custom Object Basics

RevenueForecastHistory

Represents historical information about revenue-based forecasts that have been submitted (saved) in the user interface.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

- Requires the “View All Data” permission.
- Customer Portal users can’t access this object.

Fields

Field	Details
`Closed`	**Type** currency **Properties** Filter Nillable

Field	Details
	Description The closed amount of the forecast.
`Commit`	**Type** currency **Properties** Filter, Nillable **Description** The commit amount of the forecast.
`CommitComments`	**Type** string **Properties** Filter, Nillable **Description** Comments about the commit value.
`CommitOverridden`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the commit value was overridden (`true`) or not (`false`).
`CurrencyIsoCode`	**Type** picklist **Properties** Filter, Restricted picklist **Description** Available only for organizations with the multicurrency feature enabled. Contains the ISO code for any currency allowed by the organization.
`ForecastOverrideId`	**Type** reference **Properties** Filter **Description** ID of the related forecast override.

Field	Details
Pipeline	**Type** currency **Properties** Filter, Nillable **Description** The pipeline amount of the forecast.
Quota	**Type** currency **Properties** Filter, Nillable **Description** The quota amount of the forecast.
Upside	**Type** currency **Properties** Filter, Nillable **Description** The best case amount of the forecast.
UpsideComments	**Type** string **Properties** Filter, Nillable **Description** Comments about the upside value.
UpsideOverridden	**Type** boolean **Properties** Defaulted on create Filter **Description** Indicates whether the upside value was overridden (`true`) or not (`false`).

Usage

This is a read-only object specific to customizable forecasting.

When a user submits a revenue-based forecast in the user interface, a new record is created. If the same forecast is ever resubmitted, additional records are added. The `CreatedDate` of a record reflects the day on which the forecast was submitted. This object respects field-level security on the parent object.

See Also:

- *QuantityForecastHistory*
- *RevenueForecast*
- *Standard and Custom Object Basics*

RuleTerritory2Association

Represents a record-assignment rule and its association to an object, such as Account. Available only if Territory Management 2.0 has been enabled for your organization.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

If a territory model is in `Active` state, any user can view that model, including its territories and assignment rules. For territories in an active model, any user can view assigned records and assigned users subject to your organization's sharing settings. Users cannot view territory models in other states (such as `Planning` or `Archived`).

Fields

Field Name	Details
`IsInherited`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Indicates whether the rule is an *inherited* rule (`true`) or a *local* rule (`false`). Rule inheritance flows from the parent territory where the rule is created to the rule's descendent territories (if any) in the territory model hierarchy. A local rule is created within a single territory and affects that territory only.
`RuleId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the rule.

Field Name	Details
`Territory2Id`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the territory where the rule was created.

Scontrol

Important: S-controls have been superseded by Visualforce pages. Organizations that haven't previously used s-controls can't create them. Existing s-controls are unaffected, and can still be edited. We recommend that you move your s-controls to Visualforce. We continue to support the Scontrol object.

Represents a custom s-control, which is custom content that is hosted by the system but executed by client applications. An s-control can contain any type of content that you can display or run in a Web browser.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`

Special Access Rules

- Your organization must be using Enterprise, Developer, or Unlimited Edition and be enabled for custom s-controls.
- Customer Portal users can't access this object.

Fields

Field	Details
`Binary`	**Type** base64 **Properties** Nillable, Update **Description** Binary content of this custom s-control, such as an ActiveX control or a Java archive. Can be specified when created, but not when updated. Limit: 5 MB.
`BodyLength`	**Type** int **Properties** Filter, Group, Sort

Field	Details
	Description The length of the custom s-control. Label is **Binary Length**.
`ContentSource`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Specify the source of the s-control content, either custom HTML, a snippet (s-controls that are designed to be included in other s-controls), or a URL.
`Description`	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** Description of the custom s-control.
`DeveloperName`	**Type** string **Properties** Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Label is **S-Control Name**. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`EncodingKey`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update

Field	Details
	Description Picklist of character set encodings, including ISO-08859-1, UTF-8, EUC, JIS, Shift-JIS, Korean (ks_c_5601-1987), Simplified Chinese (GB2312), and Traditional Chinese (Big5).
`Filename`	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** An uploaded object to display when the custom s-control is added to a custom link. Can be a Java applet, an ActiveX control, or any other type of desired content.
`HtmlWrapper`	**Type** textarea **Properties** Update **Description** Required. HTML page that will be delivered when the user views this custom s-control. This HTML page can be the entire content of the custom s-control, or it can reference the binary. Limit: 1,048,576 characters. Label is **HTML Body**.
`Name`	**Type** string **Properties** Filter, Group, Sort, Update **Description** Required. Name of this custom s-control. Label is **Label**.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values:

Field	Details
	• In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.
`SupportsCaching`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the s-control supports caching (`true`) or not (`false`).

Usage

Use custom s-controls to manage custom content that extends application functionality. All users can view custom s-controls, but the “Customize Application” permission is required to create or update custom s-controls.

See Also:

Standard and Custom Object Basics

ScontrolLocalization

Important: S-controls have been superseded by Visualforce pages. Organizations that haven’t previously used s-controls can’t create them. Existing s-controls are unaffected, and can still be edited.

When the Translation Workbench is enabled for your organization, provides the translation of the field label of an s-control.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- Your organization must be using Professional, Enterprise, Developer, or Unlimited Edition and be enabled for the Translation Workbench.
- To view this object, you must have the “View Setup and Configuration” permission.

Fields

Field	Details
LanguageLocaleKey	**Type** picklist **Properties** Create, Filter, Nillable, Restricted picklist **Description** This field is available in API version 16.0 and earlier. It is the same as the Language field.
Language	**Type** picklist **Properties** Create, Filter, Nillable, Restricted picklist **Description** This field is available in API version 17.0 and later. The combined language and locale ISO code, which controls the language for labels displayed in an application. This picklist contains the following fully-supported languages: • Chinese (Simplified): zh_CN • Chinese (Traditional): zh_TW • Danish: da • Dutch: nl_NL • English: en_US • Finnish: fi • French: fr • German: de • Italian: it • Japanese: ja • Korean: ko • Portuguese (Brazil): pt_BR • Russian: ru • Spanish: es • Spanish (Mexico): es_MX* * Spanish (Mexico) falls back to Spanish for customer-defined translations. • Swedish: sv • Thai: th*

Field	Details
	* Even though the Salesforce interface is fully translated to Thai, Help remains in English. The following end-user only languages are available. • Arabic: `ar` • Bulgarian: `bg` • Czech: `cs` • English (UK): `en_GB` • Greek: `el` • Hebrew: `iw` • Hungarian: `hu` • Indonesian: `in` • Norwegian: `no` • Polish: `pl` • Romanian: `ro` • Turkish: `tr` • Ukrainian: `uk` • Vietnamese: `vi` The following platform languages are available for organizations that use Salesforce exclusively as a platform. • Albanian: `sq` • Armenian: `hy` • Basque: `eu` • Bosnian: `bs` • Bengali: `bn` • Croatian: `hr` • English (Australia): `en_AU` • English (Canada): `en_CA` • English (India): `en_IN` • English (Malaysia): `en_MY` • English (Philippines): `en_PH` • Estonian: `et` • French (Canada): `fr_CA` • Georgian: `ka` • German (Austria): `de_AT` • German (Switzerland): `de_CH` • Hindi: `hi` • Icelandic: `is` • Irish: `ga` • Latvian: `lv` • Lithuanian: `lt` • Luxembourgish: `lb` • Macedonian: `mk` • Malay: `ms` • Maltese: `mt` • Romanian (Moldova): `ro_MD`

<table>
<tr><th>Field</th><th>Details</th></tr>
<tr><td></td><td>

Montenegrin: <code>sh_ME</code>
Portuguese (European): <code>pt_PT</code>
Romansh: <code>rm</code>
Serbian (Cyrillic): <code>sr</code>
Serbian (Latin): <code>sh</code>
Slovak: <code>sk</code>
Slovenian: <code>sl</code>
Tagalog: <code>tl</code>
Tamil: <code>ta</code>
Urdu: <code>ur</code>
Welsh: <code>cy</code>

The values in this field are not related to the default locale selection.
</td></tr>
<tr><td><code>NamespacePrefix</code></td><td>
Type
string

Properties
Filter, Nillable

Description
The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the <code>namespacePrefix__componentName</code> notation.

The namespace prefix can have one of the following values:

In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer.
In organizations that are not Developer Edition organizations, <code>NamespacePrefix</code> is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.

</td></tr>
<tr><td><code>ScontrolId</code></td><td>
Type
reference

Properties
Create, Filter, Nillable

Description
The ID of the Scontrol that is being translated.
</td></tr>
<tr><td><code>Value</code></td><td>
Type
string
</td></tr>
</table>

Field	Details
	Properties Create, Filter, Nillable, Update **Description** The actual translated field label of the s-control. Label is **Translation**.

Usage

Use this object to translate your s-controls into a supported language. Users with the Translation Workbench enabled can view s-control translations, but either the "Customize Application" or "Manage Translation" permission is required to create or update s-control translations.

See Also:

CategoryNodeLocalization
WebLinkLocalization

SelfServiceUser

Represents a Self-Service user.

Note: Starting with Spring '12, the Self-Service portal isn't available for new organizations. Existing organizations continue to have access to the Self-Service portal.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`ContactId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. All Self-Service users must be associated with a Contact. The contact's email should match the Self-Service user email. The contact must have a value in the `AccountId` field or an error occurs.
`Email`	**Type** email

Field	Details
	Properties Create, Filter, Group, Sort, Update **Description** Required. Make this the same as the email address for the Contact associated with this SelfServiceUser. Password resets and other system communication will be sent to this email address.
`FirstName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** First name of the Self-Service user.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the Self-Service user is allowed to log in to the Self-Service portal (`true`) or not (`false`). Note that there is no way to delete a Self-Service user. They can only be marked as inactive.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`LanguageLocaleKey`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. This is a restricted picklist field. It is the primary language for the user. All on-screen text in the Self-Service portal is displayed in this language.

Field	Details
LastLoginDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date and time when the Self-Service user last logged in.
LastName	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Last name of the Self-Service user.
LocaleSidKey	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. This is a restricted picklist field. The value of this field affects the formatting and parsing of values, especially numeric values, in the Self-Service portal. Values are two-letter codes that indicate language and sometimes language and country. The codes are based on ISO standards.
Name	**Type** string **Properties** Filter, Group, Sort **Description** Concatenation of `FirstName` and `LastName`. Limited to 121 characters.
SuperUser	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this Self-Service user is a super user with additional access on his or her company's Self-Service portal (`true`) or not (`false`).

Field	Details
TimeZoneSidKey	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. This is a restricted picklist field. The time zone of a affects the offset used when displaying or entering times in the Self-Service portal.
Username	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. This contains the name that a Self-Service user enters to log into the Self-Service portal. Value must be unique in your organization. If you try to create or update a user with a duplicate value, the operation is rejected and an error is returned.

Usage

For security reasons, you can't query Self-Service user passwords via the API or the user interface. However, the API allows you to set and reset Self-Service user passwords using the `setPassword()` and `resetPassword()` calls.

SelfServiceUser records created from the API don't cause a notification email to be sent. If you want to notify the user, you must send them an email after creating the user.

See Also:

Contact
User

ServiceContract

Represents a customer support contract (business agreement).

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
AccountId	**Type** reference

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** ID of the account associated with the service contract.
ActivationDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The initial day the service contract went into effect (whereas StartDate may include a renewal date).
ApprovalStatus	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** Approval status of the service contract.
BillingAddress (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the billing address. Read-only. See Address Compound Fields for details on compound address fields.
BillingCity	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address. Maximum size is 40 characters.
BillingCountry	**Type** string **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description Details for the billing address. Maximum size is 40 characters.
`BillingCountryCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO country code for the service contract's billing address.
`BillingLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `BillingLongitude` to specify the precise geolocation of a billing address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`BillingLongitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `BillingLatitude` to specify the precise geolocation of a billing address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`BillingPostalCode`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details for the billing address. Maximum size is 20 characters.
`BillingState`	**Type** string **Properties** Group, Sort, Filter, Nillable

Field	Details
	Description Details for the billing address. Maximum size is 20 characters.
`BillingStateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO state code for the service contract's billing address.
`BillingStreet`	**Type** textarea **Properties** Filter, Group, Nillable, Sort **Description** Street address for the billing address.
`ContactId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Required. ID of the Contact associated with the service contract. Must be a valid ID.
`ContractNumber`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, Sort **Description** Unique number automatically assigned to the service contract.
`Description`	**Type** textarea **Properties** Nillable **Description** Description of the service contract.

Field	Details
`Discount`	**Type** percent **Properties** Filter, Nillable, Sort **Description** The last day the service contract is in effect. Weighted average of all contract line item discounts on the service contract.
`EndDate`	**Type** date **Properties** Filter, Group, Nillable, Sort **Description** The last day the service contract is in effect.
`GrandTotal`	**Type** currency **Properties** Filter, Nillable, Sort **Description** Total price of the service contract plus shipping and taxes.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.

Field	Details
LastViewedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (LastReferencedDate) and not viewed.
LineItemCount	**Type** int **Properties** Filter, Nillable, Group, Sort **Description** Number of ContractLineItem records associated with the service contract.
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Name of the service contract.
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The ID of the user who currently owns the service contract.
Pricebook2Id	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the Pricebook2 associated with the service contract. Must be a valid ID.
ShippingAddress (beta)	**Type** address

Field	Details
	Properties Filter, Nillable **Description** The compound form of the shipping address. Read-only. See Address Compound Fields for details on compound address fields.
`ShippingCity`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details of the shipping address. Maximum size is 40 characters.
`ShippingCountry`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Details of the shipping address. Country maximum size is 40 characters.
`ShippingCountryCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO country code for the service contract's shipping address.
`ShippingLatitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `ShippingLongitude` to specify the precise geolocation of a shipping address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`ShippingLongitude` (beta)	**Type** double

Field	Details
	Properties Create, Filter, Nillable, Sort, Update **Description** Used with `ShippingLatitude` to specify the precise geolocation of an address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`ShippingPostalCode`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** Details of the shipping address. Postal code maximum size is 20 characters.
`ShippingState`	**Type** string **Properties** Create, Filter, Nillable, Update **Description** Details of the shipping address. State maximum size is 20 characters.
`ShippingStateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO state code for the service contract's shipping address.
`ShippingStreet`	**Type** textarea **Properties** Create, Filter, Nillable, Update **Description** The street address of the shipping address. Maximum of 255 characters.
`SpecialTerms`	**Type** textarea **Properties** Create, Nillable, Update

Field	Details
	Description Any terms specifically agreed to and tracked in the service contract.
`StartDate`	**Type** date **Properties** Create, Filter, Nillable, Update **Description** The first day the service contract is in effect.
`Status`	**Type** picklist **Properties** Filter, Nillable **Description** The status of the service contract, such as Inactive.
`Subtotal`	**Type** currency **Properties** calculated, Filter, Nillable **Description** Total of the service contract line items (products) before discounts, taxes, and shipping are applied.
`Tax`	**Type** currency **Properties** Create, Filter, Nillable, Update **Description** Total taxes for the service contract.
`Term`	**Type** int **Properties** Create, Filter, Nillable, Update **Description** Number of months that the service contract is valid.

Field	Details
TotalPrice	**Type** currency **Properties** calculated, Filter, Nillable **Description** Total of the contract line items (products) after discounts and before taxes and shipping.

Usage

Use this object to query and manage service contracts.

See Also:

ServiceContractHistory
ServiceContractShare
ServiceContractOwnerSharingRule

ServiceContractFeed

Represents a single feed item in the feed displayed on the detail page for a service contract record. A service contract feed shows recent changes to a service contract for any fields that are tracked in feeds, as well as comments and posts about the record. It's a useful way to stay up-to-date with changes to service contracts. This object is available in API version 23.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the ServiceContract object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field Name	Details
Body	**Type** textarea

Field Name	Details
	Properties Nillable, Sort **Description** The content for the ServiceContractFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort

Field Name	Details
	Description The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, `InsertedById` is set to the ID of the logged-in user.
`LastModifiedDate`	**Type** dateTime

Field Name	Details
	Properties Defaulted on create, Filter, Sort **Description** This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on the feed item, then `LastModifiedDate` becomes the `CreatedDate` for the FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities.

Field Name	Details
	Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the service contract record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as a ServiceContractFeed object of `Type ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** Title of the ServiceContractFeed. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field Name	Details
	Description The type of ServiceContractFeed: • ActivityEvent—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • AdvancedTextPost—created when a user posts a group announcement. • AnnouncementPost—Not used. • ApprovalPost—generated when a user submits an approval. • BasicTemplateFeedItem—Not used. • CanvasPost—a post made by a canvas app posts on a feed. • CollaborationGroupCreated—generated when a user creates a public group. • CollaborationGroupUnarchived—Not used. • ContentPost—a post with an attached file. • CreatedRecordEvent—generated when a user creates a record from the publisher. • DashboardComponentAlert—generated when a dashboard metric or gauge exceeds a user-defined threshold. • DashboardComponentSnapshot—created when a user posts a dashboard snapshot on a feed. • LinkPost—a post with an attached URL. • PollPost—a poll posted on a feed. • ProfileSkillPost—generated when a skill is added to a user's Chatter profile. • QuestionPost—generated when a user posts a question. • ReplyPost—generated when Chatter Answers posts a reply. • RypplePost—generated when a user creates a Thanks badge in Work.com. • TextPost—a direct text entry on a feed. • TrackedChange—a change or group of changes to a tracked field. • UserStatus—automatically generated when a user adds a post. Deprecated. The following values appear in the Type picklist for all feed objects but apply only to CaseFeed: • AttachArticleEvent—generated event when a user attaches an article to a case. • CallLogPost—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • CaseCommentPost—generated event when a user adds a case comment for a case object.

Field Name	Details
	• `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case.
`Visibility`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a service contract record.

See Also:

ServiceContract
ServiceContractHistory
ServiceContractOwnerSharingRule
FeedItem

ServiceContractHistory

Represents the history of changes to the values in the fields on a ServiceContract (customer support agreement).

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Field`	**Type** picklist **Properties** Filter, Restricted picklist **Description** The name of the field that was changed.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`NewValue`	**Type** anyType **Properties** Nillable **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable **Description** The latest value of the field before it was changed.

Field	Details
ServiceContractId	**Type** reference **Properties** Filter **Description** Required. ID of the ServiceContract.

Usage

Use this object to identify changes to a service contract.

This object respects field level security on the parent object.

See Also:

ServiceContract
ServiceContractShare
ServiceContractOwnerSharingRule

ServiceContractOwnerSharingRule

Represents the rules for sharing a ServiceContract (customer service agreement) with users other than the owner.

Note: Contact salesforce.com customer support to enable access to this object for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
AccessLevel	**Type** picklist **Properties** Create, Filter, Group, Sort, Update **Description** A value that represents the type of sharing allowed. The possible values are: • `Read` • `Edit`

Field Name	Details
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
`DeveloperName`	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Rule Name** in the user interface. This field is available in API version 24.0 and later. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`GroupId`	**Type** reference **Properties** Create, Filter, Group, Sort, **Description** The ID representing the source group. Service contracts owned by users in the source group trigger the rule to give access.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update

Field Name	Details
	Description Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to **Label** in the user interface.
`UserorGroupId`	**Type** reference **Properties** Create, Filter **Description** The ID representing the target user or group. Target users or groups are given access.

Usage

Use this object to manage the sharing rules for a service contract. General sharing and territory management-related sharing use this object.

See Also:

ServiceContract
ServiceContractShare
ServiceContractHistory

ServiceContractShare

Represents a sharing entry on a ServiceContract (customer support agreement).

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Sort, Update **Description** Level of access that the User or Group has to the ServiceContract. The possible values are: • `Read` • `Edit` • `All` (This value is not valid for create or update calls.)

Field	Details
	This field must be set to an access level that is higher than the organization's default access level for service contracts.
`ParentId`	**Type** reference **Properties** Create, Filter **Description** ID of the parent object, if any.
`RowCause`	**Type** picklist **Properties** Create, Filter, Defaulted on create, Restricted picklist **Description** Reason that this sharing entry exists. Values may include: • `Manual`—The User or Group has access because a user with "All" access manually shared the ServiceContract with them. • `Owner`—The User is the owner of the ServiceContract or is in a role above the service contract owner in the role hierarchy.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter **Description** ID of the User or Group that has been given access to the ServiceContract.

Usage

This object allows you to determine which users and groups can view and edit ServiceContract records owned by other users.

If you attempt to create a new record that matches an existing record, the create call updates any modified fields and returns the existing record.

See Also:

ServiceContract
ServiceContractHistory
ServiceContractOwnerSharingRule

SetupEntityAccess

Represents the enabled setup entity access settings (such as for Apex classes) for the parent PermissionSet. To grant users access to an entity, associate the appropriate SetupEntityAccess record with a PermissionSet that's assigned to a user. This object is available in API version 25.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the entity's parent PermissionSet.
`SetupEntityId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the entity for which access is enabled, such as an Apex class or Visualforce page.
`SetupEntityType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of setup entity for which access is enabled. Valid values are: • `ApexClass` for Apex classes • `ApexPage` for Visualforce pages • In API version 28.0 and later, `ConnectedApplication` for OAuth connected apps • In API version 31.0 and later, `CustomPermission` for custom permissions • In API version 28.0 and later, `ServiceProvider` for service providers • In API version 28.0 and later, `TabSet` for apps

Usage

Because SetupEntityAccess is a child of the PermissionSet object, the usage is similar to other PermissionSet child objects like FieldPermissions and ObjectPermissions.

For example, the following code returns all permission sets that grant access to any setup entities for which access is enabled:

```
SELECT Id, ParentId, Parent.Name, SetupEntityId
FROM SetupEntityAccess
```

The following code returns permission sets that grant access only to Apex classes:

```
SELECT Id, ParentId, Parent.Name, SetupEntityId
FROM SetupEntityAccess
WHERE SetupEntityType='ApexClass'
```

The following code returns permission sets that grant access to any setup entities, and are not owned by a profile:

```
SELECT Id, ParentId, Parent.Name, SetupEntityId
FROM SetupEntityAccess
WHERE ParentId
IN (SELECT Id
   FROM PermissionSet
   WHERE isOwnedByProfile = false)
```

You may want to return only those permission sets that have access to a specific setup entity. To do this, query the parent object. For example, this code returns all permission sets that grant access to the `helloWorld` Apex class:

```
SELECT Id, Name,
   (SELECT Id, Parent.Name, Parent.Profile.Name
   FROM SetupEntityAccessItems)
FROM ApexClass
WHERE Name = 'helloWorld'
```

While it's possible to return permission sets that have access to a `ConnectedApplication`, `ServiceProvider`, or `TabSet` by `SetupEntityId`, it's not possible to return permission sets that have access to these `SetupEntityType` fields by any other AppMenuItem attribute, such as `Name` or `Description`. For example, to find out if a user has access to the Recruiting app, you'd run two queries. First, query to get the AppMenuItem ID:

```
SELECT Id, Name, Label
FROM AppMenuItem
WHERE Name = 'Recruiting'
```

Let's say the previous query returned the AppMenuItem ID 02uD0000000GIiMIAW. Using this ID, you can now run a query to find out if a user has access to the Recruiting app:

```
SELECT Id, SetupEntityId, SetupEntityType
FROM SetupEntityAccess
WHERE ParentId
IN
   (SELECT PermissionSetId
   FROM PermissionSetAssignment
```

```
   WHERE AssigneeId = '005D0000001QOzF')
AND (SetupEntityId = '02uD0000000GIiMIAW')
```

See Also:

PermissionSet
FieldPermissions
ObjectPermissions
ApexClass
ApexPage

SignupRequest

Represents a request for a new Trialforce signup. This object is available in API version 27.0 and later.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`

Fields

Field Name	Details
`AuthCode`	**Type** string **Properties** Create, Filter, Group, Sort **Description** A one-time authorization code that can be exchanged for an OAuth access token and refresh token using standard Salesforce APIs. It's used in conjunction with `ConnectedAppCallbackUrl` and `ConnectedAppConsumerKey`, when the specified connected app hasn't been configured with an X.509 certificate. This is a read-only field provided by the system once the signup request has been processed. This field is available in API version 29.0 and later.
`Company`	**Type** string **Properties** Create, Filter, Group, Sort **Description** The name of the company requesting the trial signup.
`ConnectedAppCallbackUrl`	**Type** string

Field Name	Details
	Properties Create, Filter, Group, Sort **Description** When used in conjunction with `ConnectedAppConsumerKey`, specifies a connected app that should be approved automatically during the signup creation. This field is available in API version 28.0 and later.
`ConnectedAppConsumerKey`	**Type** string **Properties** Create, Filter, Group, Sort **Description** When used in conjunction with `ConnectedAppCallbackUrl`, specifies a connected app that should be approved automatically during the signup creation. This field is available in API version 28.0 and later.
`Country`	**Type** string **Properties** Create, Filter, Group, Sort **Description** The two-character, upper-case ISO-3166 country code. You can find a full list of these codes at a number of sites, such as: www.iso.ch/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/list-en1.html. The language of the trial organization is auto-determined based on the value of this field.
`CreatedOrgId`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The 15–character organization ID of the trial organization created. This is a read-only field provided by the system once the signup request has been processed.
`CreatedOrgInstance`	**Type** string **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description The server instance of the new trial organization, for example, “na8.” This field is available in API version 29.0 and later.
ErrorCode	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The error code if the signup request isn’t successful. This is a read-only field provided by the system to be used for support purposes.
FirstName	**Type** string **Properties** Create, Filter, Nillable, Sort **Description** The first name of the admin user for the trial signup.
LastName	**Type** string **Properties** Create, Filter, Group, Sort **Description** The last name of the admin user for the trial signup.
SignupEmail	**Type** email **Properties** Create, Filter, Group, Sort **Description** The email address of the admin user for the trial signup.
Status	**Type** picklist **Properties** Filter, Group, Sort, Update

Field Name	Details
	Description The status of the request. Possible values are `New`, `In Progress`, `Error`, or `Success`. The default value is `New`.
`Subdomain`	**Type** string **Properties** Create, Filter, Group, Sort **Description** The subdomain for the new trial organization when it uses a custom My Domain. The maximum length is 33 characters for Developer Edition (DE) and 40 characters for all other editions (because a suffix is appended to all DE organizations).
`SuppressSignupEmails`	**Type** boolean **Properties** Filter, Group, Nillable, Sort **Description** When set to `true`, no signup emails are sent when the trial organization is created. This field is used for the Proxy Signup feature, and is available in API version 29.0 and later.
`TemplateId`	**Type** string **Properties** Create, Filter, Group, Sort **Description** The 15–character ID of the approved Trialforce template that is the basis for the trial signup. The template is required and must be approved by salesforce.com.
`TrialDays`	**Type** anyType **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** The duration of the trial signup in days. Must be equal to or less than the trial days for the approved Trialforce template. If not provided, it defaults to the trial duration specified for the Trialforce template.
`TrialSourceOrgId`	**Type** string

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** The 15–character organization ID of the Trialforce Source Organization from which the Trialforce template was created.
`Username`	**Type** string **Properties** Create, Filter, Group, Sort **Description** The username of the admin user for the trial signup. It must follow the address convention specified in RFC822: www.w3.org/Protocols/rfc822/#z10

Usage

The Java class below uses the REST API to create a SignupRequest object. It authenticates to the Trialforce Management Organization and then posts a request to the SignupRequest object.

Here are the variables you need to specify in this example.

- SERVER — The name of the host server for the Trialforce Management Organization (TMO), for example, "na1.salesforce.com."
- USERNAME — The admin username for the TMO.
- PASSWORD — The concatenation of the admin password and the security token for the TMO. To get an email with the security token, from your personal settings in Salesforce select **Reset My Security Token** and click **Reset Security Token**.
- CLIENT_ID — From Setup in Salesforce, click **Create** > **Apps** and click **New** under Connected Apps. Enter values for the required fields (the Callback URL is required but can initially be set to any valid URL as it's not used), grant full access for the OAuth scopes in the "Selected OAuth Scopes" selector, and click **Save**. Then copy the value of "Consumer Key" and use it for this variable.
- CLIENT_SECRET — On the same page, click **Click to reveal**. Then copy the value of "Consumer Secret" and use it for this variable.

```
public class IsvSignupDriver {
    private static final String SERVER = server_name:port;
    private static final String USERNAME = tmo_username;
    private static final String PASSWORD = tmo_passwordsecurity_token;
    private static final String CLIENT_ID = consumer_key;
    private static final String CLIENT_SECRET = consumer_secret;

    private static SignupRequestInfo signupRequest = null;

    public static String createSignupRequest (SignupRequestInfo sr)
      throws JSONException, IOException {
       JSONObject createResponse = null;
       signupRequest = sr;
       JSONObject loginResponse = login(SERVER, USERNAME, PASSWORD);
       String instanceUrl = loginResponse.getString("instance_url");
       String accessToken = loginResponse.getString("access_token");
       createResponse = create(instanceUrl, accessToken);
       System.out.println("Created SignupRequest object: " + createResponse + "\n");
```

```
        return createResponse.toString();
    }

    /* Authenticates to the TMO using the required credentials */

    private static JSONObject login(String server, String username, String password)
      throws ClientProtocolException, IOException, JSONException {
        String authEndPoint = server + "/services/oauth2/token";
        HttpClient httpclient = new DefaultHttpClient();
        try {
            HttpPost post = new HttpPost(authEndPoint);

            List<NameValuePair> params = new ArrayList<NameValuePair>();
            params.add(new BasicNameValuePair("grant_type", "password"));
            params.add(new BasicNameValuePair("client_id", CLIENT_ID));
            params.add(new BasicNameValuePair("client_secret", CLIENT_SECRET));
            params.add(new BasicNameValuePair("username", username));
            params.add(new BasicNameValuePair("password", password));
            post.setEntity(new UrlEncodedFormEntity(params, Consts.UTF_8));

            BasicResponseHandler handler = new BasicResponseHandler();
            String response = httpclient.execute(post, handler);
            return new JSONObject(response);
        } finally {
            httpclient.getConnectionManager().shutdown();
        }
    }
    /* Posts a request to the SignupRequest object */

    private static JSONObject create(String instanceUrl, String accessToken)
      throws ClientProtocolException, IOException, JSONException {
        HttpClient httpClient = new DefaultHttpClient();
        try {
            HttpPost post = new HttpPost(instanceUrl +
              "/services/data/v27.0/sobjects/SignupRequest/");
                post.setHeader("Authorization", "Bearer " + accessToken);
                post.setHeader("Content-Type", "application/json");

                JSONObject requestBody = new JSONObject();
                requestBody.put("TemplateId", signupRequest.getTemplateID());
                requestBody.put("SignupEmail", signupRequest.getEmail());
                requestBody.put("username", signupRequest.getUsername());
                requestBody.put("Country", "US");
                requestBody.put("Company", signupRequest.getCompanyName());
                requestBody.put("lastName", signupRequest.getLastName());

                StringEntity entity = new StringEntity(requestBody.toString());
                post.setEntity(entity);
                BasicResponseHandler handler = new BasicResponseHandler();
                String response = httpClient.execute(post, handler);
                return new JSONObject(response);
        } finally {
            httpClient.getConnectionManager().shutdown();
        }
    }
}
```

Error Codes

If the signup fails, the system generates an error code that can help you identify the cause. This table shows the most important error codes.

Error Code	Description
C-1007	Duplicate username.

Error Code	Description
C-1015	Error while establishing the new organization's My Domain settings. Contact salesforce.com support for assistance.
C-1016	Error while configuring the OAuth connected app for Proxy Signup. Verify that your connected app has a valid consumer key, callback URL, and unexpired certificate (if applicable).
C-1018	Invalid subdomain value provided during signup.
C-1019	Subdomain in use. Please choose a new subdomain value.
C-9999	Generic "fatal error." Contact salesforce.com support for assistance.
S-1006	Invalild email address (not in a proper email address format).
S-2006	Invalid country code.
T-0001	Template ID not valid (not in the format 0TTxxxxxxxxxxxx).
T-0002	Template not found. Either the template doesn't exist (it may have been deleted), or it doesn't exist at the appropriate version.
T-0003	Template not approved for use by salesforce.com.

Site

Represents a public website that is integrated with an Organization. This object is generally available in API version 16.0 and later. To access this object, Force.com Sites or Site.com must be enabled for your organization.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

- Customer Portal users can't access this object.
- To view this object, you must have the "View Setup and Configuration" permission.

Fields

Field	Description
AdminId	**Type** reference **Properties** Filter, Group, Sort **Description** The site administrator designated as the contact for the site. This user receives site-related communications from site visitors and from salesforce.com.
AnalyticsTrackingCode	**Type** string

Field	Description
	Properties Filter, Group, Nillable, Sort **Description** The tracking code associated with your site. This code can be used by services like Google Analytics to track page request data for your site.
`ClickjackProtectionLevel`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Sets the clickjack protection level. The options are: • `AllowAllFraming`—Allow framing by any page (no protection) • `SameOriginOnly`—Allow framing by the same origin only (recommended) • `NoFraming`—Don't allow framing by any page (most protection) This field is available in API version 30.0 and later.
`DailyBandwidthLimit`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The rolling 24-hour daily bandwidth limit for the sites in your organization.
`DailyBandwidthUsed`	**Type** double **Properties** Filter, Group, Nillable, Sort **Description** The current rolling 24-hour daily bandwidth usage for the sites in your organization.
`DailyRequestTimeLimit`	**Type** int **Properties** Filter, Group, Nillable, Sort

Field	Description
	Description The rolling 24-hour daily service request time limit for the sites in your organization. Service request time is calculated as the total server time in minutes required to generate pages for the site.
`DailyRequestTimeUsed`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The current rolling 24-hour daily service request time for the sites in your organization.
`Description`	**Type** textarea **Properties** Filter, Nillable, Sort **Description** An optional description of the site.
`GuestUserId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The site or Salesforce Communities specific user that anonymous, unauthenticated users run as when interacting with the site.
`MasterLabel`	**Type** string **Properties** Filter, Group, Sort **Description** The name of the site as it appears in the user interface.
`MonthlyPageViewsEntitlement`	**Type** int **Properties** Filter, Group, Nillable, Sort

Field	Description
	Description The number of page views allowed for the current calendar month for the sites in your organization.
`Name`	**Type** string **Properties** Filter, Group, Sort **Description** The name used when referencing the site in the API.
`OptionsAllowHomePage`	**Type** boolean **Properties** Filter **Description** The option to enable the standard page associated with the Home tab (`/home/home.jsp`).
`OptionsAllowStandardAnswersPages`	**Type** boolean **Properties** Filter **Description** The option to enable standard pages associated with an answers community. If you want to use default Answers pages (such as AnswersHome), enable these pages.
`OptionsAllowStandardIdeasPages`	**Type** boolean **Properties** Filter **Description** The option to enable standard pages associated with an Ideas community. If you want to use default Ideas pages (such as IdeasHome), enable these pages.
`OptionsAllowStandardLookups`	**Type** boolean

Field	Description
	Properties Filter **Description** The option to enable the standard lookup pages. These are the popup windows associated with lookup fields on Visualforce pages.
`OptionsAllowStandardSearch`	**Type** boolean **Properties** Filter **Description** The option to enable the standard search pages. To allow public users to perform standard searches, enable these pages.
`OptionsEnableFeeds`	**Type** boolean **Properties** Filter **Description** The option that displays the Syndication Feeds related list, where you can create and manage syndication feeds for users on your public sites. This field is visible only if you have the feature enabled for your organization.
`SiteType`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Identifies whether the site is a Visualforce (Force.com Sites) or a Site.com site. `SiteType` is available in API version 21.0 and later. In API version 26.0 and later, if the Salesforce Communities pilot is enabled for your organization, the site could also be a Network Visualforce or Network Site.com site.
`Status`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The status for the site. For example, `Active` or `In Maintenance`.

Field	Description
`Subdomain`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The Force.com domain that you registered for your site. For example, if your Force.com domain is `mycompany.force.com`, then `mycompany` is the subdomain.
`TopLevelDomain`	**Type** url **Properties** Filter, Nillable **Description** The optional branded custom Web address that you registered with a third-party domain name registrar. The custom Web address acts as an alias to your Force.com address. Beginning with API version 21.0, `TopLevelDomain` is no longer available. Instead, use the Domain and DomainSite objects.
`UrlPathPrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The unique Force.com URL that the public uses to access this site.

Usage

Use this read-only object to query or retrieve information on your Force.com site.

SiteDomain

SiteDomain is a read-only object, and a one-to-many replacement for the Site.TopLevelDomain field. This object is available in API version 21.0, and has been deprecated as of API version 26.0. In API version 26.0 and later, use the Domain and DomainSite objects instead.

To access this object, Force.com Sites or Site.com must be enabled for your organization.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

- Customer Portal users can’t access this object.
- To view this object, you must have the “View Setup and Configuration” permission.

Fields

Field	Description
`Domain`	**Type** url **Properties** Filter, Sort **Description** The branded custom Web address within the global namespace identified by this domain's type. In the Domain Name System (DNS) global namespace, this field is the custom Web address that you registered with a third-party domain name registrar. The custom Web address can be used to access the site of this domain.
`SiteId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the associated Site.
`DomainType`	**Type** picklist **Properties** Filter, Group, Sort, Nillable **Description** The global namespace that this custom Web address belongs to. This value is set to DNS for custom Web addresses in the global DNS. This field is available in version 24.0 of the API.

Usage

Use this read-only object to query the custom Web addresses that are associated with each website in your organization.

SiteHistory

Represents the history of changes to the values in the fields of a site. This object is generally available in API version 18.0 and later. To access this object, Force.com Sites must be enabled for your organization.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

- Customer Portal users can't access this object.
- To view this object, you must have the “View Setup and Configuration” permission.

Fields

Field	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The last value of the field before it was changed.
`SiteId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the associated Site.

Skill

Represents a category or group that Live Agent users can be assigned to. This object is available in API version 24.0 and later.

Note: For information about Work.com skills on a user's profile, see the ProfileSkill topic.

Supported Calls

`create()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`Language`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The language of the skill.
`MasterLabel`	**Type** string

Field Name	Details
	Properties Create, Filter, Group, Sort, Update **Description** The name of the skill.

Usage

Use this object to assign Live Agent users to groups based on their abilities. The skills associated with a LiveChatButton determine which agents receive chat requests that come in through that button.

SkillProfile

Represents a join between Skill and Profile. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`ProfileId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** The ID of the profile.
`SkillId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the skill.

Usage

Use this object to assign specific skills to specific profiles.

SkillUser

Represents a join between Skill and User. This object is available in API version 24.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`SkillId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the skill.
`UserId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** The ID of the user.

Usage

Use this object to assign specific skills to specific users.

SlaProcess

Represents an entitlement process associated with an Entitlement. An entitlement process is a timeline that includes all the steps (MilestoneType records) that your support team must complete to resolve cases. Each process includes the logic necessary to determine how to enforce the correct service level for your customers.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
BusinessHoursId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Required. ID of the BusinessHours associated with the entitlement. Must be a valid business hours ID.
Description	**Type** textarea **Properties** Filter, Nillable **Description** A description of the entitlement process.
IsActive	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the entitlement process is active (`true`) or not (`false`).
IsVersionDefault	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the entitlement process is the default version (`true`) or not (`false`). This field is available in API version 28.0 and later in organizations that have entitlement versioning enabled.
Name	**Type** string **Properties** Filter, idLookup

Field	Details
	Description The name of the entitlement process.
NameNorm	**Type** string **Properties** Filter, Group, Sort **Description** The read-only value for the unique name of the entitlement process or the entitlement process version. If entitlement versioning is enabled, this value is automatically generated for each version of an entitlement process in this form: *process name*+_v + *x*, where *x* is the version number (for example, “gold_support_v2”). If entitlement versioning isn’t enabled, this value is the same as Name. This field is available in API version 28.0 and later.
StartDateField	**Type** picklist **Properties** Filter, Restricted picklist **Description** The criteria for cases to enter the entitlement process. Cases can enter the process based on: • The creation date on a case • A custom date/time field on a case
VersionMaster	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Identifies the sequence of versions to which this entitlement process belongs. This field’s contents can be any value as long as it is identical among all versions of the entitlement process. This field is available in API version 28.0 and later in organizations that have entitlement versioning enabled.
VersionNotes	**Type** textarea

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** The description of the entitlement process version. This field is available in API version 28.0 and later in organizations that have entitlement versioning enabled.
`VersionNumber`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** The version number of the entitlement process. Must be 1 or greater. This field is available in API version 28.0 and later in organizations that have entitlement versioning enabled.

Usage

Use this object to query entitlement processes on entitlements.

See Also:

Entitlement
MilestoneType
CaseMilestone

Solution

Represents a detailed description of a customer issue and the resolution of that issue.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Details
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter

Field	Details
	Description Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`IsHtml`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the Solution is an HTML solution (`true`) or not (`false`).
`IsOutOfDate`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Read-only field that indicates whether a solution master has been updated since the translated version was created (`true`) or not (`false`). Note that this field does not appear in the page layout of master solutions.
`IsPublished`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the Solution has been published (`true`) or not (`false`). A solution's published state does not affect how it can be used, or whether you can query, update, or delete it. Label is **Public**. **Note:** Prior to Spring '14, the label was **Visible in Self-Service Portal**.
`IsPublishedInPublicKb`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the Solution has been published as a Public Solution (`true`) or not (`false`). Label is **Visible in Public Knowledge Base**. This field only applies to solutions, not articles in the public knowledge base.

Field	Details
IsReviewed	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the Solution has been reviewed (`true`) or not (`false`). This flag can only be set indirectly via the `Status` picklist. Each predefined `Status` value implies an `IsReviewed` value. Label is **Reviewed**.
LastReferencedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
LastViewedDate	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the User who owns the Solution.
ParentId	**Type** reference **Properties** Create, Defaulted on create, Filter, Update **Description** ID of the master solution, if this is the translation of a master solution.

Field	Details
`RecordTypeId`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** ID of the RecordType to which the Solution is associated.
`SolutionLanguage`	**Type** picklist **Properties** Create, Filter, Restricted picklist, Update **Description** The language that the solution is written in, such as `French` or `Chinese (Traditional)`.
`SolutionName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. If a client application creates a new Solution and a value for this field is unspecified, a hyphen (-), the default value for this field, is used. Limit: 255 characters. Label is **Title**.
`SolutionNote`	**Type** textarea **Properties** Create, Nillable, Update **Description** The details of the Solution record. Limit: 32,000 characters. Label is **Solution Details**. **Note:** If you have HTML Solutions enabled, any HTML tags used in this field are verified before the object is created or updated. If invalid HTML is entered, an error is thrown. Any JavaScript used in this field is removed before the object is created or updated.
`SolutionNumber`	**Type** string

Field	Details
	Properties Autonumber, Defaulted on create, Filter, Sort **Description** An identifying number that is assigned automatically when a solution is created. It can't be set directly, and it can't be modified.
`Status`	**Type** picklist **Properties** Create, Filter, Group, Sort, Update **Description** Required. The status of the solution. Directly controls the `IsReviewed` value. To obtain the status values in the picklist, a client application can query the SolutionStatus.
`TimesUsed`	**Type** int **Properties** Filter, Group, Sort **Description** Number of times this solution has been used. Label is **Num Related Case**.

Usage

Use this object to manage your organization's solutions. Client applications can create, update, delete, and query Attachment records associated with a solution.

See Also:

CategoryData
CategoryNode

SolutionFeed

Represents a single feed item in the feed displayed on the detail page for a solution record. A solution feed shows recent changes to a solution record for any fields that are tracked in feeds, as well as comments and posts about the record. It is a useful way to stay up-to-date with changes to solutions. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- “Modify All Data”
- “Modify All” on the Solution object
- “Moderate Chatter”

Note: Users with the “Moderate Chatter” permission can delete only the feed items and comments they see.

Fields

Field Name	Details
`Body`	**Type** textarea **Properties** Nillable, Sort **Description** The content of the SolutionFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ConnectionId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
`ContentData`	**Type** base64 **Properties** Nillable

Field Name	Details
	Description The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`FeedPostId`	**Type** reference

Field Name	Details
	Properties Filter. Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. Use FeedItem instead. The ID of the associated FeedPost. A FeedPost represents the following types of changes in a SolutionFeed: status updates, changes to tracked fields, text posts, link posts, and content posts.
InsertedById	**Type** reference **Properties** Group, Nillable, Sort **Description** The ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into the feed, `InsertedById` is set to the ID of the logged-in user.
LastModifiedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on the feed item, then `LastModifiedDate` becomes the `CreatedDate` for the FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.
LikeCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.

Field Name	Details
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of the LinkPost.
NetworkScope	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. NetworkScope can have the following values: • NetworkId—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • AllNetworks—The feed item is available in all communities. Note the following exceptions for NetworkScope: • Only feed items with a Group or User parent can set a NetworkId or a null value for NetworkScope. • For feed items with a record parent, users can set NetworkScope only to AllNetworks. • You can't filter a FeedItem on the NetworkScope field.
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the solution record that is tracked in the feed. The feed is displayed on the detail page for this record.
RelatedRecordId	**Type** reference

Field Name	Details
	Properties Group, Nillable, Sort **Description** ID of the ContentVersion article associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as a SolutionFeed object of `Type` `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the SolutionFeed. When the `Type` is `LinkPost`, the `LinkUrl` is the URL, and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of SolutionFeed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file.

Field Name	Details
	• CreatedRecordEvent—generated when a user creates a record from the publisher. • DashboardComponentAlert—generated when a dashboard metric or gauge exceeds a user-defined threshold. • DashboardComponentSnapshot—created when a user posts a dashboard snapshot on a feed. • LinkPost—a post with an attached URL. • PollPost—a poll posted on a feed. • ProfileSkillPost—generated when a skill is added to a user's Chatter profile. • QuestionPost—generated when a user posts a question. • ReplyPost—generated when Chatter Answers posts a reply. • RypplePost—generated when a user creates a Thanks badge in Work.com. • TextPost—a direct text entry on a feed. • TrackedChange—a change or group of changes to a tracked field. • UserStatus—automatically generated when a user adds a post. Deprecated. The following values appear in the Type picklist for all feed objects but apply only to CaseFeed: • AttachArticleEvent—generated event when a user attaches an article to a case. • CallLogPost—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • CaseCommentPost—generated event when a user adds a case comment for a case object. • ChangeStatusPost—generated event when a user changes the status of a case. • ChatTranscriptPost—generated event when Live Agent transcript is saved to a case. • EmailMessageEvent—generated event when an email related to a case object is sent or received. • FacebookPost—generated when a Facebook post is created from a case. Deprecated. • MilestoneEvent—generated when a case milestone is completed or reaches violation status. • SocialPost—generated when a social post is created from a case.
Visibility	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization.

Field Name	Details
	`Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a solution record.

See Also:

Solution
EntitySubscription
NewsFeed
UserProfileFeed

SolutionHistory

Represents the history of changes to the values in the fields of a solution.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed. Label is **Custom Field Definition ID**.
`IsDeleted`	**Type** boolean

Field	Details
	Properties Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed. Limited to 255 characters.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed. Limited to 255 characters.
`SolutionId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the Solution. Label is **Solution ID**.

Usage

Use this read-only object to identify changes to a solution.

This object respects field-level security on the parent object.

See Also:

SolutionStatus

SolutionStatus

Represents the status of a Solution, such as Draft, Reviewed, and so on.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`IsDefault`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this is the default solution status value (`true`) or not (`false`) in the picklist. Only one value can be the default value.
`IsReviewed`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this solution status value represents a reviewed Solution (`true`) or not (`false`). Multiple solution status values can represent a reviewed Solution.
`MasterLabel`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Master label for this solution status value. This display value is the internal label that does not get translated.
`SortOrder`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Number used to sort this value in the solution status picklist. These numbers are not guaranteed to be sequential, as some previous solution status values might have been deleted.

Usage

This object represents a value in the solution status picklist. The solution status picklist provides additional information about the status of a Solution, such as whether a given status value represents a reviewed or unreviewed solution. Your client application can query this object to retrieve the set of values in the solution status picklist, and then use that information while processing Solution objects to determine more information about a given solution. For example, the application could test whether a given case has been reviewed or not based on its `Status` value and the value of the `IsReviewed` property in the associated SolutionStatus record.

See Also:

Solution

SolutionTag

Associates a word or short phrase with a Solution.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter

Field Name	Details
	Description ID of the parent TagDefinition object that owns the tag.
`Type`	**Type** picklist **Properties** Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

SolutionTag stores the relationship between its parent TagDefinition and the Solution being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

StaticResource

Represents a static resource that can be used in Visualforce markup.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
`Body`	**Type** base64 **Properties** Create, Nillable, Update **Description** Required. Encoded file data.
`BodyLength`	**Type** int

Field	Details
	Properties Filter, Group, Sort **Description** Size of the file (in bytes).
`CacheControl`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The sharing policy for the static resource when cached. The cache control can have one of the following values: • `Private` specifies that the static resource data cached on the Salesforce server shouldn't be shared with other users. The static resource is only stored in cache for the current user's session. • `Public` specifies that the static resource data cached on the Salesforce server be shared with other users in your organization for faster load times.
`ContentType`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Type of content. Label is **Mime Type**. Limit: 120 characters.
`Description`	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Text description of the static resource. Limit: 255 characters.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of the static resource.

Field	Details
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects. This field can't be accessed unless the logged-in user has the "Customize Application" permission.

Usage

Use static resources to upload content that you can reference in Visualforce markup, including archives (such as .zip and .jar files), images, stylesheets, JavaScript, and other files. Using a static resource is preferable to uploading a file to the Documents tab because:

- You can package a collection of related files into a directory hierarchy and upload that hierarchy as a .zip or .jar archive.
- You can reference a static resource in page markup by name using the `$Resource` global variable instead of hard-coding document IDs.

Encoded Data

The API sends and receives the binary file data encoded as a base64 data type. Prior to creating a record, clients must encode the binary file data as base64. Upon receiving an API response, clients must decode the base64 data to binary (this conversion is usually handled for you by the SOAP client).

Maximum Static Resource Size

You can create or update static resources to a maximum size of 5 MB. An organization can have up to 250 MB of static resources, total.

See Also:

ApexComponent
ApexPage
Developer's Guide: Visualforce Developer's Guide

StreamingChannel

Represents a channel that is the basis for notifying listeners of generic Streaming API events. This is available from API version 29.0 or later.

Note: Generic streaming is currently available through a pilot program. For information on enabling generic streaming for your organization, contact salesforce.com, inc..

Supported Calls

REST: DELETE, GET, PATCH, POST (query requests are specified in the URI)

SOAP: `create()`, `delete()`, `describe()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`

Special Access Rules

- This object is only available if Streaming API is enabled for your organization.
- Only users with “Create” permission can create this record. Users with “View All Data” can view StreamingChannel records and see streaming messages.
- You can apply user sharing to StreamingChannel. You can restrict access to receiving or sending events on a channel by sharing channels with specific users or groups. Channels shared with public read only or read-write access will only send events to clients subscribed to the channel that also are using a user session associated with the set of shared users or groups. Only users with read-write access to a shared channel can generate events on the channel, or modify the actual StreamingChannel record.

Fields

Field	Field Type	Description
`Description`	string	Description of the StreamingChannel. Limit: 255 characters. **Field Properties:** Create, Filter, Group, Nillable, Sort, Update **Label:** Description
`ID`	ID	System field: Globally unique string that identifies a StreamingChannel record. **Field Properties:** Default on create, Filter, Group, idLookup, Sort
`IsDeleted`	boolean	System field: Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). **Field Properties:** Default on create, Filter, Group, Sort
`IsDynamic`	boolean	`true` if the channel gets dynamically created on subscribe if necessary, `false` otherwise. To enable dynamic streaming channels in your organization, from Setup, go to **Customize** > **User Interface** and enable **Enable Dynamic Streaming Channel Creation**. **Field Properties:** Default on create, Filter, Group, Sort
`LastReferencedDate`	date	The timestamp for when the current user last viewed a record related to this record. **Field Properties:** Filter, Sort

Field	Field Type	Description
LastViewedDate	date	The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (LastReferencedDate) and not viewed. **Field Properties:** Filter, Sort
Name	string	Required. Descriptive name of the StreamingChannel. Limit: 80 characters, alphanumeric and "_", "/" characters only. Must start with "/u/". This value identifies the channel. **Field Properties:** Create, Filter, Group, idLookup, Sort, Update **Label:** Streaming Channel Name
OwnerId	reference	The ID of the owner of the StreamingChannel. **Field Properties:** Create, Default on create, Filter, Group, Sort, Update **Label:** Owner Name

TagDefinition

Defines the attributes of child Tag objects.

Supported Calls

`delete()`, `describeSObjects()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`

Fields

Field	Detail
Name	**Type** string **Properties** Filter, Nillable, Update **Description** Identifies the tag word or phrase.
Type	**Type** picklist **Properties** Filter, Nillable, Restricted picklist **Description** Defines the visibility of a tag. Possible value are: • **Public**: The tag can be viewed and manipulated between all users in an organization.

Field	Detail
	• **Personal**: The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

When you create a tag for a record, an association is created with to a corresponding TagDefinition:

- If the value in the tag's `Name` field is new, a new TagDefinition record is automatically created and becomes the parent of the tag.
- If the value in the tag's `Name` field already exists in a TagDefinition, that TagDefinition automatically becomes the parent of the tag.

Each TagDefinition record has a one-to-many relationship with its child tag records.

The following standard objects represent tags for records:

- AccountTag
- AssetTag
- CampaignTag
- CaseTag
- ContactTag
- ContractTag
- DocumentTag
- EventTag
- LeadTag
- NoteTag
- OpportunityTag
- SolutionTag
- TaskTag

Custom objects may also be tagged. Tags for custom objects are identified by a suffix of two underscores immediately followed by the word `tag`. For example, a custom object named `Meeting` has a corresponding tag named Meeting__tag in that organization's WSDL. Meeting__tag is only valid for `Meeting` objects.

TagDefinition is useful for mass operations on any tag record. For instance, if you want to rename existing tags, you can search for the appropriate TagDefinition object, update it, and the child tag's `Name` values are also changed. The following Java example replaces all `WC` tags with the phrase `West Coast`:

```
public void tagDefinitionSample() {
  String soqlQuery = "SELECT Id, Name FROM TagDefinition " +
      "WHERE Name = 'WC'";
  QueryResult qResult = null;
  try {
      qResult = connection.query(soqlQuery);
    TagDefinition tagDef = (TagDefinition) qResult.getRecords()[0];
    tagDef.setName("West Coast");
    connection.update(new SObject[]{tagDef});
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

Task

Represents a business activity such as making a phone call or other to-do items. In the user interface, Task and Event records are collectively referred to as activities.

Note: Task fields related to calls are exclusive to Salesforce CRM Call Center. Additionally, `query()`, `delete()`, and `update()` aren't allowed with tasks related to more than one contact in API versions 23.0 and earlier.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Field	Field Type
`AccountId`	**Type** reference **Properties** Group, Sort, Filter, Nillable **Description** Represents the ID of the related Account. The `AccountId` is determined as follows. If the value of `WhatId` is any of the following objects, then Salesforce uses that object's `AccountId`. • Account • Opportunity • Contract • Custom object that is a child of Account If the value of the `WhatId`field is any other object, and the value of the `WhoId` field is a Contact object, then Salesforce uses that contact's `AccountId`. (If your organization uses Shared Activities, then Salesforce uses the `AccountId` of the primary contact.) Otherwise, Salesforce sets the value of the `AccountId` field to `null`. For information on IDs, see ID Field Type.
`ActivityDate`	**Type** date **Properties** Group, Sort, Create, Filter, Nillable, Update **Description** Represents the due date of the task. This field has a timestamp that is always set to midnight in the Coordinated Universal Time (UTC) time zone. The timestamp is

Field	Field Type
	not relevant; do not attempt to alter it in order to accommodate time zone differences. Label is **Due Date**. **Note:** This field can't be set or updated for a recurring task (`IsRecurrence` is `true`).
`CallDisposition`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Represents the result of a given call, for example, "we'll call back," or "call unsuccessful." Limit is 255 characters. Not subject to field-level security, available for any user in an organization with Salesforce CRM Call Center.
`CallDurationInSeconds`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Duration of the call in seconds. Not subject to field-level security, available for any user in an organization with Salesforce CRM Call Center.
`CallObject`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Name of a call center. Limit is 255 characters. Not subject to field-level security, available for any user in an organization with Salesforce CRM Call Center.
`CallType`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update

Field	Field Type
	Description The type of call being answered: Inbound, Internal, or Outbound.
ConnectionReceivedId	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that shared this record with your organization. This field is only available if you have enabled Salesforce to Salesforce.
ConnectionSentId	**Type** reference **Properties** Filter, Nillable **Description** ID of the PartnerNetworkConnection that you shared this record with. This field is only available if you have enabled Salesforce to Salesforce. Beginning with API version 15.0, the `ConnectionSentId` field is no longer supported. The `ConnectionSentId` field is still visible, but the value is null. You can use the new PartnerNetworkRecordConnection object to forward records to connections.
Description	**Type** textarea **Properties** Create, Nillable, Update **Description** Contains a text description of the task.
IsArchived	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the event has been archived.
IsClosed	**Type** boolean

Field	Field Type
	Properties Defaulted on create, Filter, Group, Sort **Description** Indicates whether the task has been completed (`true`) or not (`false`). Is only set indirectly via the `Status` picklist. Label is **Closed**.
`IsHighPriority`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates a high-priority task. This field is derived from the `Priority` field.
`IsRecurrence`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** Indicates whether the task is scheduled to repeat itself (`true`) or only occurs once (`false`). This is a read-only field on update, but not on create. If this field value is `true`, then `RecurrenceStartDateOnly`, `RecurrenceEndDateOnly`, `RecurrenceType`, and any recurrence fields associated with the given recurrence type must be populated. See **Recurring Tasks**.
`IsReminderSet`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether a popup reminder has been set for the task (`true`) or not (`false`).
`IsVisibleInSelfService`	**Type** boolean **Properties** Defaulted on create, Filter, Update **Description** Indicates whether a task associated with an object can be viewed in the Customer Portal (`true`) or not (`false`).

Field	Field Type
	If your organization has enabled Communities, tasks marked `IsVisibleInSelfService` are visible to any external user in the community, as long as the user has access to the record the task was created on. This field is available when Customer Portal or partner portal are enabled OR Communities is enabled and you have Customer Portal or partner portal licenses.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the User who owns the record. Label is **Assigned To ID**.
`Priority`	**Type** picklist **Properties** Group, Sort, Create, Filter, Update **Description** Required. Indicates the importance or urgency of a task, such as high or low.
`RecurrenceActivityId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Read-only. Not required on create. ID of the main record of the recurring task. Subsequent occurrences have the same value in this field.
`RecurrenceDayOfMonth`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The day of the month in which the task repeats.
`RecurrenceDayOfWeekMask`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Field Type
	Description The day or days of the week on which the task repeats. This field contains a bitmask. The values are as follows: • Sunday = 1 • Monday = 2 • Tuesday = 4 • Wednesday = 8 • Thursday = 16 • Friday = 32 • Saturday = 64 Multiple days are represented as the sum of their numerical values. For example, Tuesday and Thursday = 4 + 16 = 20.
`RecurrenceEndDateOnly`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The last date on which the task repeats.
`RecurrenceInstance`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The frequency of the recurring task. For example, "2nd" or "3rd."
`RecurrenceInterval`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The interval between recurring tasks.
`RecurrenceMonthOfYear`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The month of the year in which the task repeats.

Field	Field Type
RecurrenceRegeneratedType	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Represents what triggers a repeating task to repeat. Add this field to a page layout together with the `RecurrenceInterval` field, which determines the number of days between the triggering date (due date or close date) and the due date of the next repeating task in the series. Label is **Repeat This Task**. This field has the following picklist values: • **None**: The task doesn't repeat. • **After due date**: The next repeating task will be due the specified number of days after the current task's due date. • **After the task is closed**: The next repeating task will be due the specified number of days after the current task is closed. • **(Task closed)**: This task, now closed, was opened as part of a repeating series.
RecurrenceStartDateOnly	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The date when the recurring task begins. Must be a date and time before `RecurrenceEndDateOnly`.
RecurrenceTimeZoneSidKey	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The time zone associated with the recurring task. For example, "UTC-8:00" for Pacific Standard Time.
RecurrenceType	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Indicates how often the task repeats. For example, daily, weekly, or every nth month (where "nth" is defined in `RecurrenceInstance`).

Field	Field Type
ReminderDateTime	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** Represents the time when the reminder is scheduled to fire, if `IsReminderSet` is set to `true`. If `IsReminderSet` is set to `false`, then the user may have deselected the reminder checkbox in the Salesforce user interface, or the reminder has already fired at the time indicated by the value.
Status	**Type** picklist **Properties** Create, Filter, Group, Sort, Update **Description** Required. The current status of the task, such as In Progress or Completed. Each predefined `Status` field implies a value for the `IsClosed` flag. To obtain picklist values, query the TaskStatus object. **Note:** This field can't be updated for recurring tasks (`IsRecurrence` is `true`).
Subject	**Type** combobox **Properties** Create, Filter, Nillable, Sort, Update **Description** The subject line of the task, such as "Call" or "Send Quote." Limit: 255 characters.
Type	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** The type of task, such as Call or Meeting.
WhatCount	**Type** int

Field	Field Type
	Properties Filter, Group, Nillable, Sort **Description** Available to organizations that have Shared Activities enabled. Count of related TaskRelations pertaining to `WhatId`. Count of the `WhatId` must be `1` or less.
`WhatId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of a related Account, Opportunity, Campaign, Case, or custom object. Label is **Opportunity/Account ID**.
`WhoCount`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Available to organizations that have Shared Activities enabled. Count of related TaskRelations pertaining to `WhoId`.
`WhoId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Indicates the ID of a related Contact or Lead. If the `WhoId` refers to a lead, then the `WhatId` field must be empty. Label is **Contact/Lead ID**. If Shared Activities is enabled, the value of this field is the ID of a related Lead or primary Contact.

Usage

Recurring Tasks

- Recurring tasks are available in API version 16.0 and later.
- After a task is created, it can't be changed from recurring to nonrecurring or vice versa.
- When you delete a recurring task series through the API, all open and closed task occurrences in the series are removed. However, when you delete a recurring task series through the user interface, only open tasks occurrences (`IsClosed` is `false`) in the series are removed.
- If `IsRecurrence` is `true`, then `RecurrenceStartDateOnly`, `RecurrenceEndDateOnly`, `RecurrenceType`, and any properties associated with the given recurrence type (see the following table) must be populated.

- When you change the `RecurrenceStartDateOnly` field or the recurrence pattern, all open tasks occurrences in the series are deleted and new open task occurrences are created based on the new recurrence pattern. The recurrence pattern is determined by the following fields: `RecurrenceType`, `RecurrenceTimeZoneSidKey`, `RecurrenceInterval`, `RecurrenceDayOfWeekMask`, `RecurrenceDayOfMonth`, `RecurrenceInstance`, and `RecurrenceMonthOfYear`.
- When you change the value of `RecurrenceEndDateOnly` to an earlier date (for example, from January 20th to January 10th), all open task occurrences in the series with the `ActivityDate` value greater than the new end date value are deleted. Other open and closed task occurrences in the series are not affected.
- When you change the value of `RecurrenceEndDateOnly` to a later date (for example, from January 10th to January 20th), new task occurrences are created up to the new end date. Existing open and closed tasks in the series are not affected.

The following table describes the usage of recurrence fields. Each recurrence type must have all of its properties set. All unused properties must be set to null.

RecurrenceType Value	Properties	Example Pattern
RecursDaily	RecurrenceInterval	Every second day
RecursEveryWeekday	RecurrenceDayOfWeekMask	Every weekday - can't be Saturday or Sunday
RecursMonthly	RecurrenceDayOfMonth RecurrenceInterval	Every second month, on the third day of the month
RecursMonthlyNth	RecurrenceInterval RecurrenceInstance RecurrenceDayOfWeekMask	Every second month, on the last Friday of the month
RecursWeekly	RecurrenceInterval RecurrenceDayOfWeekMask	Every three weeks on Wednesday and Friday
RecursYearly	RecurrenceDayOfMonth RecurrenceMonthOfYear	Every March on the twenty-sixth day of the month
RecursYearlyNth	RecurrenceDayOfWeekMask RecurrenceInstanceRecurrenceMonthOfYear	The first Saturday in every October

See Also:

Standard and Custom Object Basics

TaskFeed

Represents a single feed item in the feed on an Task.

Represents a

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
Body	**Type** textarea **Properties** Nillable, Sort **Description** The content of TaskFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
CommentCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
ConnectionId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
ContentData	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.

Field	Details
ContentDescription	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in ContentData.
ContentFileName	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if Type is ContentPost. The name of the file uploaded to the feed. Setting ContentFileName automatically sets Type to ContentPost.
ContentSize	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
ContentType	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
CreatedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by CreatedDateDESC sorts the feed by the most recent feed item.

Field	Details
FeedPostId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
InsertedById	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
LastModifiedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.

Field	Details
LikeCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
NetworkScope	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
ParentId	**Type** reference **Properties** Filter, Group, Sort

Field	Details
	Description ID of the account record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the task record. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file.

Field	Details
	• `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`:

Field	Details
	• For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a task record.

TaskPriority

Represents the importance or urgency of a Task, such as High, Normal, or Low.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`IsDefault`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this is the default task priority value (`true`) or not (`false`) in the picklist. Only one value in the picklist can be the default value.
`IsHighPriority`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this task priority value represents a high priority Task (`true`) or not (`false`). Multiple task priority values can represent a high-priority Task.
`MasterLabel`	**Type** string

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** Master label for this task priority value. This display value is the internal label that does not get translated. Limit: 255 characters.
`SortOrder`	**Type** int **Properties** Filter, Nillable, Group, Sort **Description** Number used to sort this value in the task priority picklist. These numbers are not guaranteed to be sequential, as some previous task priority values might have been deleted.

Usage

This object represents a value in the task priority picklist. The task priority picklist provides additional information about the importance of a Task, such as whether a given priority value represents a high priority. Your client application can query on this object to retrieve the set of values in the task priority picklist, and then use that information while processing Task objects to determine more information about a given task. For example, the application could test whether a given Task is high priority based on its `Priority` value and the value of the `IsHighPriority` in the associated TaskPriority object.

See Also:

Standard and Custom Object Basics

TaskRelation

Represents the relationship between a task and a lead, contacts, and other objects related to the task.

TaskRelation allows a variable number of relationships, as follows:

- A task can be related to one lead or up to 50 contacts.
- A task can also be related to one account, asset, campaign, case, contract, opportunity, product, solution, or custom object.

Available in API versions 24.0 and later. Available only if you've enabled Shared Activities for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `queryAll()`,`retrieve()`

Fields

Field Name	Details
`AccountId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the Account ID of the relation. For information on IDs, see ID Field Type.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether a task has been deleted; label is **Deleted**. When a TaskRelation record is deleted, it isn't moved to the Recycle Bin and can't be undeleted, unless the record was cascade-deleted when the parent object was deleted. Don't use the `IsDeleted` field to detect deleted records in SOQL queries or `queryAll()` calls on directly deleted relation records. Instead, use the call `getDeleted()`.
`IsWhat`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** Indicates whether the relation is an Account, Opportunity, Campaign, Case, other standard object, or a custom object. Value is `false` if `RelationId` is a contact or lead and `true` otherwise.
`RelationId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Indicates the `WhatId` or `WhoId` in the relationship. For more information, see `Task`. For information on IDs, see ID Field Type.

Field Name	Details
TaskId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Represents the ID of the associated Task. For information on IDs, see ID Field Type.

Usage

See contacts associated with a task

```
public void queryWhosOfTaskSample() {
     String soqlQuery = "SELECT Id, Subject, (SELECT RelationId, Relation.Name, IsWhat
 from TaskRelations WHERE isWhat = false) FROM Task WHERE Id = '00T x00000050KEN'";
    QueryResult qResult = null;
    try {
        qResult = connection.query(soqlQuery);
        TaskRelation relation1 =
(TaskRelation)qResult.getRecords()[0].getTaskRelations().getRecords()[0];
    }catch (ConnectionException ce) {
        ce.printStackTrace();
     }
 }
```

See Also:

Task
TaskWhoRelation

TaskStatus

Represents the status of a Task, such as Not started, Completed, or Closed.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`IsClosed`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this task status value represents a closed Task (`true`) or not (`false`). Multiple task status values can represent a closed Task.
`IsDefault`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether this is the default task status value (`true`) or not (`false`) in the picklist.
`MasterLabel`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Master label for this task status value. This display value is the internal label that does not get translated. Limit: 255 characters.
`SortOrder`	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Number used to sort this value in the task status picklist. These numbers are not guaranteed to be sequential, as some previous task status values might have been deleted.

Usage

This object represents a value in the task status picklist. The task status picklist provides additional information about the status of a Task, such as whether a given status value represents an open or closed task. Your client application can query this object to retrieve the set of values in the task status picklist, and then use that information while processing Task records to determine more information about a given task. For example, the application could test whether a given task is open or closed based on the Task `Status` value and the value of the `IsClosed` property in the associated TaskStatus record.

See Also:

Standard and Custom Object Basics

TaskTag

Associates a word or short phrase with a Task.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`ItemId`	**Type** reference **Properties** Create, Filter **Description** ID of the tagged item.
`Name`	**Type** string **Properties** Create, Filter **Description** Name of the tag. If this value does not already exist, a new TagDefinition is created and becomes the parent of this Tag object. Otherwise, a TagDefinition with the same name becomes the parent of this Tag object. Parent relationships are created automatically.
`TagDefinitionId`	**Type** reference **Properties** Filter **Description** ID of the parent TagDefinition object that owns the tag.
`Type`	**Type** picklist

Field Name	Details
	Properties Create, Filter, Restricted picklist **Description** Defines the visibility of a tag. Valid values: • `Public`—The tag can be viewed and manipulated by all users in an organization. • `Personal`—The tag can be viewed or manipulated only by a user with a matching `OwnerId`.

Usage

TaskTag stores the relationship between its parent TagDefinition and the Task being tagged. Tag objects act as metadata, allowing users to describe and organize their data.

When a tag is deleted, its parent TagDefinition will also be deleted if the name is not being used; otherwise, the parent remains. Deleting a TagDefinition sends it to the Recycle Bin, along with any associated tag entries.

TaskWhoRelation

TaskWhoRelation represents the relationship between a task and a lead or contacts. This derived object is a filtered version of the TaskRelation on page 1241 object; that is, IsParent is `true` and IsWhat is `false`. It doesn't represent relationships to accounts, opportunities, or other objects.

TaskWhoRelation allows a variable number of relationships: one lead or up to 50 contacts. Available only if you've enabled Shared Activities for your organization. This object is available in API versions 29.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`RelationId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the contacts or lead related to the task.
`TaskId`	**Type** reference **Properties** Filter, Group, Nillable, Sort

Field Name	Details
	Description Indicates the ID of the task.
Type	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Indicates whether the person related to the task is a lead or contact.

Usage

Apex example that queries contacts associated with a task

```
public void queryWhosOfTaskSample() {
    String soqlQuery = "SELECT Id, Subject, (SELECT RelationId, Relation.Name, IsWhat
from TaskWhoRelations) FROM Task WHERE Id = '00Tx0000005OKEN'";
    QueryResult qResult = null;
    try {
        qResult = connection.query(soqlQuery);
        TaskWhoRelation relation1 =
(TaskWhoRelation)qResult.getRecords()[0].getTaskWhoRelations().getRecords()[0];
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

See Also:

Task
TaskRelation

Territory

Represents a flexible collection of accounts and users where the users have at least read access to the accounts, regardless of who owns the accounts. Only available if territory management has been enabled for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field	Details
AccountAccessLevel	**Type** picklist

Field	Details
	Properties Create, Filter, Group, Sort, Update **Description** Account access level granted to users assigned to this territory.
`CaseAccessLevel`	**Type** picklist **Properties** Create, Filter, Nillable, Group, Sort, Update **Description** Case access level granted to users assigned to this territory.
`ContactAccessLevel`	**Type** picklist **Properties** Filter, Group, Nillable, Sort **Description** A value that represents the type of access granted to the target Group, UserRole, or User for any associated contacts. The possible values are: • `None` • `Read` • `Edit` **Note:** When `DefaultContactAccess` is set to “Controlled by Parent,” you can’t create or update this field.
`Description`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the territory that is 1,000 characters or less.
`DeveloperName`	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field	Details
	Description The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Territory Name** in the user interface. This field is available in API version 24.0 and later. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`ForecastUserId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the Forecast Manager, who is the user to whom forecasts from this territory's child territories roll up.
`MayForecastManagerShare`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the forecast manager can manually share their own forecast.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** A name for the territory. Limit is 80 characters. Corresponds to **Label** on the user interface.
`OpportunityAccessLevel`	**Type** picklist

Field	Details
	Properties Create, Filter, Group, Sort, Update **Description** Opportunity access level granted to users assigned to this territory.
`ParentTerritoryID`	**Type** reference **Properties** Create, Filter, Nillable, Update **Description** Territory immediately above this territory in the territory hierarchy. Label is **Parent Territory ID**.
`RestrictOppTransfer`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Update **Description** Indicates whether the opportunities associated with this territory are kept within the bounds of this territory and this territory's children when account assignment rules are run (`true`), or if opportunities associated with this territory can be assigned to other nodes of the territory hierarchy when account assignment rules are run (`false`). Label is **Confine Opportunity Assignment**.

Usage

Use the Territory object to query your organization's territory hierarchy. Use it to obtain valid territory IDs when querying or modifying records associated with territories.

See Also:

AccountTerritoryAssignmentRule
AccountTerritoryAssignmentRuleItem
UserTerritory

Territory2

Represents a sales territory. Available only if Territory Management 2.0 has been enabled for your organization.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

If a territory model is in `Active` state, any user can view that model, including its territories and assignment rules. For territories in an active model, any user can view assigned records and assigned users subject to your organization's sharing settings. Users cannot view territory models in other states (such as `Planning` or `Archived`).

Fields

Field Name	Details
AccountAccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Represents the default account record access levels for users that are assigned to the territory. Values are: • `Read Only` • `Read/Write` • `Owner`
CaseAccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Represents the default case record access levels for users that are assigned to the territory. Values are: • `Private` • `Read Only` • `Read/Write`
ContactAccessLevel	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Represents the default contact record access levels for users that are assigned to the territory. Values are: • `Private` • `Read Only` • `Read/Write`

Field Name	Details
Description	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** The description of the territory.
DeveloperName	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. The unique name of the object in the API. This name can contain only underscores and alphanumeric characters and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. The field label in the user interface is `Territory Name`. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The name of the territory. The field label in the user interface is `Label`.
OpportunityAccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Represents the default opportunity record access levels for users that are assigned to the territory. Values are: • `Private` • `Read Only` • `Read/Write`

Field Name	Details
ParentTerritory2Id	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the territory's parent territory (if any). If the territory has no parent territory, this value is `null`.
Territory2ModelId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the territory model that the territory belongs to.
Territory2TypeId	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** The ID of the territory type that the territory belongs to.

Territory2Model

Represents a territory model. Available only if Territory Management 2.0 has been enabled for your organization.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

If a territory model is in `Active` state, any user can view that model, including its territories and assignment rules. For territories in an active model, any user can view assigned records and assigned users subject to your organization's sharing settings. Users cannot view territory models in other states (such as `Planning` or `Archived`).

Fields

Field Name	Details
ActivatedDate	**Type** dateTime

Field Name	Details
	Properties Filter, Nillable, Sort **Description** The date when the territory model was activated.
DeactivatedDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date when the territory model was archived.
Description	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The description of the territory model.
DeveloperName	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. The unique name of the object in the API. This name can contain only underscores and alphanumeric characters and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. The field label in the user interface is `Territory Model Name`. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
LastRunRulesEndDate	**Type** dateTime **Properties** Filter, Nillable, Sort

Field Name	Details
	Description The date when the last rules run was completed.
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** The territory model name. The field label in the user interface is `Label`.
State	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The state of the territory model. Values are: `Planning`, `Activating`, `Activation Failed`, `Active`, `Archiving`, `Archiving Failed`, `Archived`, `Deleting`, and `Deletion Failed`.

Territory2ModelHistory

Represents the history of changes to the values in the fields on a territory model. Available only if Territory Management 2.0 has been enabled for your organization.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
Field	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field whose value was changed.
NewValue	**Type** anyType

Field Name	Details
	Properties Nillable, Sort **Description** The new value of the changed field.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The previous value of the changed field.
`Territory2ModelId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the territory model whose history is tracked.

Usage

This object is automatically generated whenever any field value changes on a territory model record. Use this object it to identify those changes.

Territory2Type

Represents a category for territories (Territory2). Every Territory2 must have a Territory2Type. Available only if Territory Management 2.0 has been enabled for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

All users (including standard users) have access to this object in the user interface

Fields

Field Name	Details
`Description`	**Type** textarea

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The description of the territory type.
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. The unique name of the object in the API. This name can contain only underscores and alphanumeric characters and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. The field label in the user interface is `Territory Type Name`. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`Language`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The language of the master label in the user interface.
`MasterLabel`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required The user interface label for the territory type.
`Priority`	**Type** int **Properties** Filter, Group, Sort

Field Name	Details
	Description Reserved for future use.

Topic

Represents a topic on a Chatter post or record. This object is available in API version 28.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `update()`, `upsert()`

Fields

Field Name	Details
`Description`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** Description of the topic.
`Name`	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update **Note:** You can change only the spacing and capitalization of a topic name with the update property. **Description** Name of the topic.
`NetworkId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** Identifier of the community to which the Topic belongs. This field is available only if Salesforce Communities is enabled in your organization.

Field Name	Details
TalkingAbout	**Type** int **Properties** Filter, Group, Sort **Description** Number of people talking about the topic over the last two months, based on factors such as topic additions and comments on posts with the topic.

Usage

Use this object to query a specific topic or to get a list of all topics, even those used solely in private groups and on records, and the number of people talking about them.

Use this object to create, edit, or delete topics. To create a topic, you must have the “Create Topics” permission. To edit a topic, you must have the “Edit Topics” permission. To delete a topic, you must have the ”Delete Topics” or “Modify All Data” permission.

TopicAssignment

Represents the assignment of a topic to a specific feed item or record. This object is available in API version 28.0 and later. Topics for objects is available in API version 30.0 and later. Administrators must enable topics for objects before users can add topics to records of that object type.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
EntityId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Identifier of the feed item or record.
NetworkId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort

Field Name	Details
	Description Identifier of the community to which the TopicAssignment belongs. This field is available only if Salesforce Communities is enabled in your organization.
`TopicId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Identifier of the topic.

Usage

Use this object to query the assignments of topics to feed items or records.

No SOQL limit if logged-in user has “View All Data” permission. If not, do one of the following:

- Specify a LIMIT clause of 1,100 records or fewer.
- Filter on `Id` or `Entity` when using a `WHERE` clause with "=".

To assign and remove topics from feed items or records, you must have the “Assign Topics” permission.

See Also:

Topic

FeedItem

TopicFeed

Represents a single feed tracked change for a topic. This object is available in API version 29.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items you created. To delete feed items you didn’t create, you must have one of these permissions:

- “Modify All Data”
- “Moderate Chatter”

Note: Users with the “Moderate Chatter” permission can delete only the feed items and comments they see.

Fields

Field Name	Details
Body	**Type** textarea **Properties** Nillable, Sort **Description** The content of the FeedItem. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
CommentCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
ConnectionId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
ContentData	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.

Field Name	Details
ContentDescription	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
ContentFileName	**Type** string **Properties** Group, Nillable, Sort **Description** The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
ContentSize	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
ContentType	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
InsertedById	**Type** reference **Properties** Group, Nillable, Sort

Field Name	Details
	Description ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a CollaborationGroup or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`.

Field Name	Details
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the topic that is tracked in the feed.
RelatedRecordId	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`.
Title	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the FeedItem. When the `Type` is `LinkPost`, the `LinkUrl` is the URL and this field is the link name.
Type	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of FeedItem: • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence.

Field Name	Details
	• `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case.

Field Name	Details
	Note: If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a topic.

UndecidedEventRelation

Represents invitees with the status `Not Responded` for a given event.

This object is available in API versions 29.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field Name	Details
`EventId`	**Type** reference

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** Indicates the ID of the event.
`RelationId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** Indicates the ID of the invitee.
`RespondedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** This field is always `null`.
`Response`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Indicates the content of the response field. Label is `Comment`.
`Type`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** Indicates whether the invitee is a user, lead or contact, or resource.

Usage

Query invitees who have not responded to an invitation to an event

```
SELECT eventId, type, response FROM UndecidedEventRelation WHERE
eventid='00UTD000000ZH5LA'
```

See Also:

AcceptedEventRelation
DeclinedEventRelation

User

Represents a user in your organization.

Supported Calls

`create()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `update()`, `upsert()`

Special Access Rules

- To create or update a User record, you must have the "Manage Internal Users" permission. Additionally, if the user is a Customer Portal user, you must have the "Edit Self-Service Users" permission, and if the user is a partner portal user, you must have the "Manage External Users" permission.
- If Communities is enabled in your organization, to create or update external users for Customer Portal, partner portal, or Communities, you must also have the "Manage External Users" permission.
- Information in hidden fields in a user's profile is not searchable by external users (with a portal profile) in a community. For example, if a user in a community has a hidden email address and an external user searches for it, the user record is not returned in the search results. Hidden field values are also not returned when external users perform searches on non-hidden fields. So if an external user searches for a user's name (cannot be hidden), any hidden field values associated with the user record (for example, a hidden email address) are not returned in the search results.

  Internal users from your organization who belong to the same community, however, can both search for and view hidden field values in search results.
- When requested by portal users, queries that look up to the User object, such as `owner.name` or `owner.email` might not return values when the portal user making the request doesn't have Read access to the User record being queried.

  The behavior depends on the number of domains associated with the lookup field. If the object can look up to more than one domain, `owner.name` returns a value, but other detail fields don't. For example, Case owner can look up to the User or Queue objects. In this case, portal users can see only the value of `owner.name`. Other User detail fields, such as `owner.email` or `owner.phone` don't return a value.

  If the object can look up to only a single domain, such as Account owner, then no detail fields return values, including `owner.name`.
- Changing ownership of a record by updating its `OwnerId` field requires the user making the change to have both the "Transfer Record" permission and Read access to the User record of the new record owner.

Fields

Field	Details
`AboutMe`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Information about the user, such as areas of interest or skills.
`AccountId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the Account associated with a Customer Portal user. This field is null for Salesforce users.
`Address` (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the address. Read-only. See Address Compound Fields for details on compound address fields.
`Alias`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. The user's alias. For example, `jsmith`.
`CallCenterId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description If Salesforce CRM Call Center is enabled, represents the call center to which this user is assigned.
`City`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The city associated with the user. Up to 40 characters allowed.
`CommunityNickname`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Name used to identify this user in the Community application, which includes the ideas and answers features.
`CompanyName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The name of the user's company.
`ContactId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** ID of the Contact associated with this account. The contact must have a value in the `AccountId` field or an error occurs.
`Country`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description The country associated with the user. Up to 80 characters allowed.
`CountryCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO country code associated with the user.
`CurrentStatus`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Text that describes what the user is working on. **Note:** If you update this field, the API automatically adds a post of type `UserStatus` on the user's profile in Chatter. This field is deprecated in API version 25.0. To achieve similar behavior, post to the user directly by creating a FeedItem with the user's ParentId.
`DefaultCurrencyIsoCode`	**Type** picklist **Properties** Create, Filter, Nillable, Restricted picklist, Update **Description** The user's default currency setting for new records. For example, a user in France could have a `DefaultCurrencyIsoCode` set to Euros, and that would be their default currency in the application. Only applicable for organizations that use multiple currencies.
`DefaultDivision`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field	Details
	Description This record’s default division. Only applicable if divisions are enabled for your organization.
`DefaultGroupNotificationFrequency`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Required. The default frequency for sending the user's Chatter group email notifications when the user joins groups. The valid values are: • `P`—Email on each post • `D`—Daily digests • `W`—Weekly digests • `N`—Never The default value is `N`. For Professional, Enterprise, Unlimited, and Developer Edition organizations that existed prior to API version 22.0, the default value remains `D`. This field is available in API version 21.0 and later.
`DelegatedApproverId`	**Type** reference **Properties** Create, Filter, Group, Nillable,Sort, Update **Description** Id of the user who is a delegated approver for this user.
`Department`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The company department associated with the user.
`DigestFrequency`	**Type** picklist

Field	Details
	Properties Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Required. The frequency at which the system sends the user's Chatter personal email digest. The valid values are: • `D` = Daily • `W` = Weekly • `N` = Never The default value is `D`.
`Division`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The division associated with this user, similar to Department and unrelated to `DefaultDivision`.
`Email`	**Type** email **Properties** Create, Filter, Group, idLookup, Sort, Update **Description** Required. The user's email address.
`EmailEncodingKey`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. The email encoding for the user, such as `ISO-8859-1` or `UTF-8`.
`EmailPreferencesAutoBcc`	**Type** boolean **Properties** Create, Filter, Update

Field	Details
	Description Determines whether the user will receive copies of sent emails. This option applies only if compliance BCC emails are not enabled.
`EmailPreferencesAutoBccStayInTouch`	**Type** boolean **Properties** Create, Filter, Update **Description** Determines whether the user will receive copies of sent Stay-in-Touch emails. This option applies only if compliance BCC emails are not enabled.
`EmailPreferencesStayInTouchReminder`	**Type** boolean **Properties** Create, Filter, Update **Description** Determines whether the user is reminded to use the Stay-in-Touch feature when creating a new contact.
`EmployeeNumber`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The user's employee number.
`Extension`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The user's phone extension number.
`Fax`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description The user’s fax number.
`FederationIdentifier`	**Type** string **Properties** Create, Filter, idLookup, Nillable, Sort, Update **Description** Indicates the value that must be listed in the `Subject` element of a Security Assertion Markup Language (SAML) *IDP certificate* to authenticate the user for a client application using single sign-on. This value must be specified if the `SAML User ID Type` is Assertion contains Federation ID from the User record. Otherwise, this field can’t be edited.
`FirstName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The user’s first name.
`ForecastEnabled`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the user is enabled as a Forecast Manager (`true`) or not (`false`) in customizable forecasting. Forecast managers see forecast rollups from users below them in the forecast hierarchy.
`FullPhotoUrl`	**Type** string **Properties** Filter, Nillable, Sort **Description** The URL for the user's profile photo if Chatter is enabled. The URL is updated every time a photo is uploaded and reflects the most recent photo. The URL returned for an older photo is not guaranteed to return a photo if a newer photo has been

Field	Details
	uploaded. You should always query this field for the URL of the most recent photo. This field is available in API version 20.0 and later.
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the user has access to log in (`true`) or not (`false`). You can modify a User's active status from the user interface or via the API.
`IsBadged`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the user is visually badged (`true`) or not (`false`). Users of the same Chatter user type (internal, external) are badged. Different user types are not badged.
`IsPartner`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the user is a partner who has access to the partner portal (`true`) or not (`false`). This field is not available for release 9.0 and later. Instead, use `UserType` with the value `Partner` or `Power Partner`.
`IsPortalEnabled`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the user has access to partner portal, Customer Portal, or Communities (`true`) or not (`false`).

Field	Details
	This field is only available if partner relationship management (partner portal) or Customer Portal is enabled, OR if Communities is enabled and you have partner portal, Customer Portal, or High-Volume Portal User licenses.
`IsPortalSelfRegistered`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort **Description** Indicates whether the user is a Customer Portal user who self-registered for your organization's Customer Portal (`true`) or not (`false`). This field is not available for release 9.0 and earlier.
`IsPrmSuperUser`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the user has super user access in the partner portal (`true`) or not (`false`). This field is available in API version 24.0 and later. **Note:** This field is not automatically enabled. Contact salesforce.com to enable this field.
`JigsawImportLimitOverride`	**Type** int **Properties** Create, Filter, Group Nillable, Sort, Update **Description** The Data.com user's monthly addition limit. The value must be between zero and the organization's monthly addition limit. Label is **Data.com Monthly Addition Limit**. This field is available in API version 27.0 and later.
`LanguageLocaleKey`	**Type** picklist

Field	Details
	Properties Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. The user's language, such as "French" or "Chinese (Traditional)." Label is **Language**.
`LastLoginDate`	**Type** dateTime **Properties** Filter, Sort, Nillable **Description** The date and time when the user last logged in.
`LastName`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. The user's last name.
`LastReferencedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed a record related to this record.
`LastViewedDate`	**Type** date **Properties** Filter, Nillable, Sort, Update **Description** The timestamp for when the current user last viewed this record. If this value is null, this record might only have been referenced (`LastReferencedDate`) and not viewed.
`Latitude` (beta)	**Type** double

Field	Details
	Properties Create, Filter, Nillable, Sort, Update **Description** Used with `Longitude` to specify the precise geolocation of an address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`LocaleSidKey`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. This field is a restricted picklist field. The value of the field affects formatting and parsing of values, especially numeric values, in the user interface. It does not affect the API. The field values are named according to the language, and country if necessary, using two-letter ISO codes. The set of names is based on the ISO standard. It can often be more convenient to manually set a user's locale in the user interface, and then use that value for inserting or updating other users via the API.
`Longitude` (beta)	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** Used with `Latitude` to specify the precise geolocation of an address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.
`Manager`	**Type** picklist **Properties** Create, Filter, Restricted picklist, Update **Description** User lookup field used to select the user's manager. This establishes a hierarchical relationship, preventing you from selecting a user that directly or indirectly reports to itself.
`ManagerId`	**Type** reference

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The Id of the user who manages this user.
`MiddleName` (beta)	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The user's middle name. Maximum size is 40 characters. Contact salesforce.com Customer Support to enable this field.
`MobilePhone`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The user's mobile or cellular phone number.
`Name`	**Type** string **Properties** Filter, Group, Sort **Description** Concatenation of `FirstName` and `LastName`. Limited to 121 characters.
`OfflineTrialExpirationDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The date and time when the user's Connect Offline trial expires.
`Phone`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description The user's phone number.
`PortalRole`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort **Description** The role of the user in the Customer Portal (either Executive, Manager, User, or PersonAcount). Prior to API version 16.0, you could set this field to null and the system automatically included a portal role. In API version 16.0 and above, when you set this field to null, a portal role is not automatically created. When this field is null and a `ContactId` is provided, the user is assigned to the User role. The field is available if Customer Portal is enabled OR Communities is enabled and have available partner portal, Customer Portal, or High-Volume Portal User licenses.
`PostalCode`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The user's postal or ZIP code. Label is **Zip/Postal Code**.
`ProfileId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** Required. ID of the user's Profile. Use this value to cache metadata based on profile. In earlier releases, this was `RoleId`.
`ReceivesAdminInfoEmails`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field	Details
	Description Indicates whether the user receives email for administrators from salesforce.com (`true`) or not (`false`).
`ReceivesInfoEmails`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the user receives informational email from salesforce.com (`true`) or not (`false`).
`SenderEmail`	**Type** email **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The email address used as the From address when the user sends emails. This is the same value shown in Setup on the My Email Settings page.
`SenderName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The name used as the email sender when the user sends emails. This is the same value shown in Setup on the My Email Settings page.
`Signature`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** The signature text added to emails. This is the same value shown in Setup on the My Email Settings page.
`SmallPhotoUrl`	**Type** string

Field	Details
	Properties Filter, Nillable, Sort **Description** The URL for a thumbnail of the user's profile photo if Chatter is enabled. The URL is updated every time a photo is uploaded and reflects the most recent photo. The URL returned for an older photo is not guaranteed to return a photo if a newer photo has been uploaded. You should always query this field for the URL of the most recent photo. This field is available in API version 20.0 and later.
`State`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The state associated with the User. Up to 80 characters allowed.
`StateCode`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ISO state code associated with the user.
`StayInTouchNote`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** The body specified for the Stay-in-Touch email.
`StayInTouchSignature`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** The signature specified for the Stay-in-Touch email.

Field	Details
StayInTouchSubject	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The subject specified for the Stay-in-Touch email.
Street	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The street address associated with the User.
Suffix (beta)	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The user's name suffix. Maximum size is 40 characters. Contact salesforce.com Customer Support to enable this field.
TimeZoneSidKey	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. This field is a restricted picklist field. A User time zone affects the offset used when displaying or entering times in the user interface. However, the API does not use a User time zone when querying or setting values. Values for this field are named using region and key city, according to ISO standards. It can often be more convenient to manually set one User time zone in the user interface, and then use that value for creating or updating other User records via the API.
Title	**Type** string

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The user's business title, such as "Vice President."
`Username`	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update **Description** Required. Contains the name that a user enters to log into the API or the user interface. The value for this field must be in the form of an email address. It must also be unique across all organizations. If you try to create or update a User with a duplicate value for this field, the operation is rejected. Each inserted User also counts as a license. Every organization has a maximum number of licenses. If you attempt to exceed the maximum number of licenses by inserting User records, the create is rejected.
`UserPermissionsCallCenterAutoLogin`	**Type** boolean **Properties** Create, Filter, Update **Description** Required if Salesforce CRM Call Center is enabled. Indicates whether the user is enabled to use the auto login feature of the call center (`true`) or not (`false`).
`UserPermissionsChatterAnswersUser`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates whether the portal user is enabled to use the Chatter Answers feature (`true`) or not (`false`). This field defaults to `false` when a Customer Portal user is created from the API.
`UserPermissionsInteractionUser`	**Type** boolean

Field	Details
	Properties Create, Filter, Update **Description** Indicates whether the user can run flows or not. Label is **Force.com Flow User**.
`UserPermissionsJigsawProspectingUser`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates whether the user is allocated one Data.com user license (`true`) or not (`false`). The Data.com user license lets the user add Data.com contact and lead records to Salesforce in supported editions. Label is **Data.com User**.
`UserPermissionsKnowledgeUser`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates whether the user is enabled to use Salesforce Knowledge (`true`) or not (`false`). Label is **Knowledge User**.
`UserPermissionsMarketingUser`	**Type** boolean **Properties** Create, Filter, Update **Description** Required. Indicates whether the user is enabled to manage campaigns in the user interface (`true`) or not (`false`). Label is **Marketing User**.
`UserPermissionsMobileUser`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates whether the user is allocated one Salesforce Classic license (`true`) or not (`false`). Label is **Apex Mobile User**. The

Field	Details
	Salesforce Classic license grants the user access to the Salesforce Classic application on supported mobile devices.
`UserPermissionsOfflineUser`	**Type** boolean **Properties** Create, Filter, Update **Description** Required. Indicates whether the user is enabled to use Offline Edition (`true`) or not (`false`). Label is **Offline User**.
`UserPermissionsSFContentUser`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates whether the user is allocated one Salesforce CRM Content User License (`true`) or not (`false`). Label is **Salesforce CRM Content User**. The Salesforce CRM Content User license grants the user access to the Salesforce CRM Content application.
`UserPermissionsSiteforceContributorUser`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates whether the user is allocated one Site.com Contributor feature license (`true`) or not (`false`). Label is **Site.com Contributor User**. The Site.com Contributor feature license grants the user access to the Site.com application. Users with a Contributor license can use Site.com Studio to edit site content only.
`UserPermissionsSiteforcePublisherUser`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates whether the user is allocated one Site.com Publisher feature license (`true`) or not (`false`). Label is **Site.com Publisher User**. The Site.com Publisher feature license grants the user access to the Site.com application. Users with a Publisher

Field	Details
	license can build and style websites, control the layout and functionality of pages and page elements, and add and edit content.
UserPermissionsSupportUser	**Type** boolean **Properties** Create, Filter, Update **Description** When `true`, the user can use the Salesforce console.
UserPermissionsWirelessUser	**Type** boolean **Properties** Create, Update **Description** Required if the Wireless permission is enabled for your organization. Indicates whether the user is enabled to use Wireless Edition (`true`) or not (`false`). Label is **Wireless User**. **Note:** As of November 2005, Salesforce Wireless Edition is no longer available for purchase. If you are a Professional Edition customer and purchased Wireless Edition prior to November 7, 2005 or are an Enterprise Edition customer who has signed or renewed your Salesforce contract prior to November 7, 2005, you may continue using Wireless Edition through the end of your existing contract term.
UserPermissionsWorkDotComUserFeature	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates whether the Work.com feature is enabled for the user (`true`) or not (`false`).
UserPreferencesActivityRemindersPopup	**Type** boolean **Properties** Create, Filter, Update

Field	Details
	Description When `true`, a reminder popup window automatically opens when an activity reminder is due. Corresponds to the `Trigger alert when reminder comes due` checkbox at the Reminders page in the personal settings in the user interface.
`UserPreferencesApexPagesDeveloperMode`	**Type** boolean **Properties** Create, Filter, Update **Description** When `true`, indicates the user has enabled developer mode for editing Visualforce pages and controllers.
`UserPreferencesContentEmailAsAndWhen`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, a user with Salesforce CRM Content subscriptions will receive a once daily email summary if activity occurs on his or her subscribed content, libraries, tags, or authors. To receive email, the `UserPreferencesContentNoEmail` field must also be `false`. The default value is `false`. **Note:** This field is only visible when Salesforce CRM Content is enabled for your organization.
`UserPreferencesContentNoEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, a user with Salesforce CRM Content subscriptions will receive email notifications if activity occurs on his or her subscribed content, libraries, tags, or authors. To receive real-time email alerts, set this field to `false` and set the `UserPreferencesContentEmailAsAndWhen` field to `true`. The default value is `false`.

Field	Details
	Note: This field is only visible when Salesforce CRM Content is enabled for your organization.
UserPreferencesEnableAutoSubForFeeds	**Type** boolean **Properties** Create, Filter, Update **Description** When `true`, the user automatically subscribes to feeds for any objects that the user creates. This field is available in API version 25.0 and later.
UserPreferencesDisableAllFeedsEmail	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email for all updates to Chatter feeds, based on the types of feed emails and digests the user has enabled. This field is available in API version 24.0 and later.
UserPreferencesDisableAutoSubForFeeds	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically subscribes to feeds for any objects that the user creates. This field is deprecated in API version 25.0 and later. Starting with API version 25.0, use `UserPreferencesEnableAutoSubForFeeds` to enable or disable auto-follow for objects a user creates.
UserPreferencesDisableBookmarkEmail	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email every time someone comments on a Chatter feed item after the user has

Field	Details
	bookmarked it. This field is available in API version 24.0 and later.
UserPreferencesDisableChangeCommentEmail	**Type** boolean **Properties** Create, Filter, Update **Description** When false, the user automatically receives email every time someone comments on a change the user has made, such as an update to their profile. This field is available in API version 24.0 and later.
UserPreferencesDisableEndorsementEmail	**Type** boolean **Properties** Create, Filter, Update **Description** When false, the member automatically receives email every time someone endorses them for a topic.
UserPreferencesDisableFileShareNotificationsForApi	**Type** boolean **Properties** Create, Filter, Update **Description** When false, email notifications are sent from the person who has shared a file to the users with whom the file has been shared. This field is available in API version 25.0 and later.
UserPreferencesDisableFollowersEmail	**Type** boolean **Properties** Create, Filter, Update **Description** When false, the user automatically receives email every time someone starts following the user in Chatter.This field is available in API version 24.0 and later.
UserPreferencesDisableLaterCommentEmail	**Type** boolean

Field	Details
	Properties Create, Filter, Update **Description** When `false`, the user automatically receives email every time someone comments on a feed item after the user has commented on the feed item. This field is available in API version 24.0 and later.
`UserPreferencesDisableLikeEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email every time someone likes a post or comment the user has made. This field is available in API version 24.0 and later.
`UserPreferencesDisableMentionsPostEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email every time the user is mentioned in posts. This field is available in API version 24.0 and later.
`UserPreferencesDisableProfilePostEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email every time someone posts to the user's profile. This field is available in API version 24.0 and later.
`UserPreferencesDisableSharePostEmail`	**Type** boolean **Properties** Create, Filter, Update

Field	Details
	Description When `false`, the user automatically receives email every time the user's post is shared. This field is available in API version 24.0 and later.
`UserPreferencesDisableFeedbackEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives emails related to Work.com feedback. This includes when someone requests or offers feedback, shares feedback with the user, or reminds the user to answer a feedback request.
`UserPreferencesDisCommentAfterLikeEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email every time someone comments on a post the user has liked. This field is available in API version 24.0 and later.
`UserPreferencesDisMentionsCommentEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email every time the user is mentioned in comments. This field is available in API version 24.0 and later.
`UserPreferencesDisableMessageEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email for Chatter messages sent to the user. This field is available in API version 24.0 and later.

Field	Details
UserPreferencesDisableRewardEmail	**Type** boolean **Properties** Filter **Description** When false, the user automatically receives emails related to Work.com rewards. This includes when someone someone gives a reward to the user.
UserPreferencesDisableWorkEmail	**Type** boolean **Properties** Filter **Description** When false, the user receives emails related to Work.com feedback, goals, and coaching. The user must also sign up for individual emails listed on the Work.com email settings page. When true, the user will not receive any emails related to Work.com feedback, goals, or coaching even if they are signed up for individual emails.
UserPreferencesDisProfPostCommentEmail	**Type** boolean **Properties** Create, Filter, Update **Description** When false, the user automatically receives email every time someone comments on posts on the user's profile. This field is available in API version 24.0 and later.
UserPreferencesEventRemindersCheckboxDefault	**Type** boolean **Properties** Create, Filter, Update **Description** When true, a reminder popup is automatically set on the user's events. Corresponds to the By default, set reminder on Events to... checkbox on the Reminders page in the user interface. This field is related to UserPreference and customizing activity reminders.

Field	Details
UserPreferencesHideChatterOnboardingSplash	**Type** boolean **Properties** Filter **Description** When `true`, the initial Chatter onboarding prompts do not appear.
UserPreferencesHideCSNDesktopTask	**Type** boolean **Properties** Create, Filter, Update **Description** When `true`, the Chatter recommendations panel never displays the recommendation to install Chatter Desktop. This field is available in API version 26.0 and later.
UserPreferencesHideCSNGetChatterMobileTask	**Type** boolean **Properties** Create, Filter, Update **Description** When `true`, the Chatter recommendations panel never displays the recommendation to install Chatter Mobile. This field is available in API version 26.0 and later.
UserPreferencesHideSecondChatterOnboardingSplash	**Type** boolean **Properties** Filter **Description** When `true`, the secondary Chatter onboarding prompts do not appear.
UserPreferencesHideS1BrowserUI	**Type** boolean **Properties** Create, Filter, Update

Field	Details
	Description Controls the interface that the user sees when logging in to Salesforce from a supported mobile browser. If `false`, the user is automatically redirected to the Salesforce1 mobile browser app. If `true`, the user sees the full Salesforce site. The default value is `false`. Label is **Salesforce1 User**. This field is available in API version 29.0 or later.
`UserPreferencesJigsawListUser`	**Type** boolean **Properties** Create, Filter, Update **Description** When `true`, the user is a Data.com List user and, therefore, shares record additions from a pool. UserPermissionsJigsawProspectingUser must also be set to `true`. Label is **Data.com List User**. This field is available in API version 27.0 and later.
`UserPreferencesOptOutOfTouch`	**Type** boolean **Properties** Create, Filter, Update **Description** This field is deprecated in API version 29.0. When `false`, the user automatically accesses the Salesforce Touch app when logging in to Salesforce from an iPad. If `true`, automatic access to the Salesforce Touch app is turned off and the user's iPad is directed to the full Salesforce site instead. The default value is `false`. **Note:** Salesforce Touch must be enabled before this field is visible.
`UserPreferencesReminderSoundOff`	**Type** boolean **Properties** Create, Filter, Update

Field	Details
	Description When `true`, a sound automatically plays when an activity reminder is due. Corresponds to the `Play a reminder sound` checkbox on the Reminders page in the user interface.
`UserPreferencesShowCityToExternalUsers`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the city field in the user's contact information. City is visible only to internal members of the user's organization when: • This field is `false`. When `false`, this field returns the value `#N/A`. City is visible to external members in a community when: • This field is `true`, or • This field is `false` but `UserPreferencesShowCityToGuestUsers` is `true`, which overrides this field's value. External users are users with Community, Customer Portal, or partner portal licenses. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowCityToGuestUsers`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the city field in the user's contact information. When `true`, city is visible to guest users. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the value `#N/A`. When `true`, this field overrides the value `false` in `UserPreferencesShowCityToExternalUsers`, making the user's city visible to external members. The default value is `false`. This field is available in API version 28.0 and later.

Field	Details
UserPreferencesShowCountryToExternalUsers	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the country field in the user's contact information. Country is visible only to internal members of the user's organization when: • This field is `false`. When `false`, this field returns the value `#N/A`. Country is visible to external members in a community when: • This field is `true`, or • This field is `false` but `UserPreferencesShowCountryToGuestUsers` is `true`, which overrides this field's value. External users are users with Community, Customer Portal, or partner portal licenses. The default value is `false`. This field is available in API version 26.0 and later.
UserPreferencesShowCountryToGuestUsers	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the country field in the user's contact information. When `true`, country is visible to guest users. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the value `#N/A`. When `true`, this field overrides the value `false` in `UserPreferencesShowCountryToExternalUsers`, making the user's country visible to external members. The default value is `false`. This field is available in API version 28.0 and later.
UserPreferencesShowEmailToExternalUsers	**Type** boolean **Properties** Create, Filter, Update

Field	Details
	Description Indicates the visibility of the email address field in the user's contact information. Email address is visible only to internal members of the user's organization when this field is `false`. Email address is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowFaxToExternalUsers`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the fax number field in the user's contact information. Fax number is visible only to internal members of the user's organization when this field is `false`. Fax number is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowManagerToExternalUsers`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the manager field in the user's contact information. Manager is visible only to internal members of the user's organization when this field is `false`. Manager is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowMobilePhoneToExternalUsers`	**Type** boolean **Properties** Create, Filter, Update

Field	Details
	Description Indicates the visibility of the mobile or cellular phone number field in the user's contact information. The number is visible only to internal members of the user's organization when this field is `false`. The number is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowPostalCodeToExternalUsers`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the postal or ZIP code field in the user's contact information. Postal code is visible only to internal members of the user's organization when: • This field is `false`. When `false`, this field returns the value `#N/A`. Postal code is visible to external members in a community when: • This field is `true`, or • This field is `false` but `UserPreferencesShowPostalCodeToGuestUsers` is `true`, which overrides this field's value. External users are users with Community, Customer Portal, or partner portal licenses. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowPostalCodeToGuestUsers`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the postal or ZIP code field in the user's contact information. When `true`, postal code is visible to guest users. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the value `#N/A`.

Field	Details
	When `true`, this field overrides the value `false` in `UserPreferencesShowPostalCodeToExternalUsers`, making the user's postal code visible to external members. The default value is `false`. This field is available in API version 28.0 and later.
`UserPreferencesShowProfilePicToGuestUsers`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the user's profile photo. When `true`, the photo is visible to guest users in a community. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the stock photo. The default value is `false`. This field is available in API version 28.0 and later.
`UserPreferencesShowStateToExternalUsers`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the state field in the user's contact information. State is visible only to internal members of the user's organization when: • This field is `false`. When `false`, this field returns the value `#N/A`. State is visible to external members in a community when: • This field is `true`, or • This field is `false` but `UserPreferencesShowStateToGuestUsers` is `true`, which overrides this field's value. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowStateToGuestUsers`	**Type** boolean

Field	Details
	Properties Create, Filter, Update **Description** Indicates the visibility of the state field in the user's contact information. When `true`, state is visible to guest users. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the value `#N/A`. When `true`, this field overrides the value `false` in `UserPreferencesShowStateToExternalUsers`, making the user's state visible to external members. The default value is `false`. This field is available in API version 28.0 and later.
`UserPreferencesShowStreetAddressToExternalUsers`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the street address field in the user's contact information. The address is visible only to internal members of the user's organization when this field is `false`. The address is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowTitleToExternalUsers`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the business title field in the user's contact information. Title is visible only to internal members of the user's organization when: • This field is `false`. When `false`, this field returns the value `#N/A`. Title is visible to external members in a community when: • This field is `true`, or

Field	Details
	• This field is `false` but `UserPreferencesShowTitleToGuestUsers` is `true`, which overrides this field's value. External users are users with Community, Customer Portal, or partner portal licenses. The default value is `true`. This field is available in API version 26.0 and later.
UserPreferencesShowTitleToGuestUsers	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the business title field in the user's contact information. When `true`, title is visible to guest users. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the value `#N/A`. When `true`, this field overrides the value `false` in `UserPreferencesShowTitleToExternalUsers`, making the user's title visible to external members. The default value is `false`. This field is available in API version 28.0 and later.
UserPreferencesShowWorkPhoneToExternalUsers	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates the visibility of the work phone number field in the user's contact information. The number is visible only to internal members of the user's organization when this field is `false`. The number is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
UserPreferencesTaskRemindersCheckboxDefault	**Type** boolean

Field	Details
	Properties Create, Filter, Update **Description** When `true`, a reminder popup is automatically set on the user's tasks. Corresponds to the `By default, set reminder on Tasks to...` checkbox on the Reminders page in the user interface. This field is related to UserPreference and customizing activity reminders.
`UserRoleId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the user’s UserRole. Label is **Role ID**.
`UserType`	**Type** picklist **Properties** Filter, Group, Nillable, Sort, Restricted picklist **Description** The category of user license. Each `UserType` is associated with one or more UserLicense records. Each UserLicense is associated with one or more profiles. In API version 10.0 and later, valid values include: • Standard: user license. This user type also includes Salesforce Platform and Salesforce Platform One user licenses. Label is **Standard**. • PowerPartner: User whose access is limited because he or she is a partner and typically accesses the application through a partner portal or community. Label is **Partner**. • CSPLitePortal: user whose access is limited because he or she is an organization's customer and accesses the application through aCustomer Portal or community. Label is **High Volume Portal**. • CustomerSuccess: user whose access is limited because he or she is an organization's customer and accesses the application through a Customer Portal. Label is **Customer Portal User**. • PowerCustomerSuccess: user whose access is limited because he or she is an organization's customer and accesses the application through a Customer Portal. Label is **Customer Portal Manager**.

Field	Details
	Users with this license type can view and edit data they directly own or data owned by or shared with users below them in the Customer Portal role hierarchy. • CsnOnly: user whose access to the application is limited to Chatter. This user type includes Chatter Free and Chatter moderator users. Label is **Chatter Free**.
`WirelessEmail`	**Type** email **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Wireless email address associated with this user. For use with Salesforce Wireless Edition. This field is available only if the Wireless and Email permissions are enabled for your organization. **Note:** As of November 2005, Salesforce Wireless Edition is no longer available for purchase. If you are a Professional Edition customer and purchased Wireless Edition prior to November 7, 2005 or are an Enterprise Edition customer who has signed or renewed your Salesforce contract prior to November 7, 2005, you may continue using Wireless Edition through the end of your existing contract term.

Usage

Use this object to query information about users and to provision and modify users in your organization. Unlike other objects, the records in the User table represent actual users—not data owned by users. Any user can query or describe User records.

For example, the following SOQL code finds users with a particular user role.

```
SELECT Id, Username
FROM User
WHERE UserRoleId='00ED0000000xicT'
```

Each portal user is associated with a portal account. A portal account can have a maximum of three portal roles (Executive, Manager, and User). You can select the default number of roles to be created from the user interface. The role hierarchy is maintained when you insert and delete portal roles, and roles are created bottom-up. Deleting the User role causes the Manager role to be renamed to User role. Deleting both the Executive and User roles causes the Manager role to be renamed to User role. Before deleting a role, you must assign users under that role to another role.

Deactivate Users

You can't delete a user in the user interface or the API. You can deactivate a user in the user interface; and you can deactivate or disable a Customer Portal or partner portal user in the user interface or the API. Because users can never be deleted, we recommend that you exercise caution when creating them.

If you deactivate a user, any EntitySubscription where the user is associated with the ParentId or SubscriberId field, meaning all subscriptions both to and from the user, are soft deleted. If the user is reactivated, the subscriptions are restored. However, if you deactivate multiple users at once and these users follow each other, their subscriptions are hard deleted. In this case, the user-to-user EntitySubscription is deleted twice (double deleted). Such subscriptions can’t be restored upon user reactivation.

Passwords

For security reasons, you can’t query User passwords via the API or the user interface. However, the API allows you to set and “reset” User passwords using the `setPassword()` and `resetPassword()` calls. The password lockout status and the ability to reset the User locked-out status is not available via the API. You must check and reset the User password lockout status using the user interface.

See Also:

getUserInfo()
create()
update()
query()
search()
retrieve()
upsert()
update()
getUpdated()
getDeleted()
describeSObjects()
Frequently-Occurring Fields
UserRole
UserLicense

UserAccountTeamMember

Represents a User on the default account team of another User.

See also OpportunityTeamMember, which represents a User on the opportunity team of an Opportunity

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`AccountAccessLevel`	**Type** picklist **Properties** Create, Filter, Restricted picklist, Update

Field	Details
	Description Required. For Account records that the user has added to his or her default account team, the level of access the account team member has. . The possible values are: • `Read` • `Edit` This field must be set to an access level that is higher than the organization's default access level for accounts.
`CaseAccessLevel`	**Type** picklist **Properties** Create, Filter, Restricted picklist, Update **Description** Required. Level of access that the account team member has to Case records related to the account. The possible values are: • `None` • `Read` • `Edit` This field must be set to an access level that is higher than the organization's default access level for cases.
`ContactAccessLevel`	**Type** picklist **Properties** Create, Filter, Restricted picklist, Update **Description** Required. ForContact records related to the account, the level of access that the account team member has. The possible values are: • `None` • `Read` • `Edit` This field must be set to an access level that is higher than the organization's default access level for contacts. **Note:** When `DefaultContactAccess` is set to `Controlled by Parent`, you can't create or update this field.
`OpportunityAccessLevel`	**Type** picklist

Field	Details
	Properties Create, Filter, Restricted picklist, Update **Description** Required. Level of access that the team member has to Opportunity records related to the account. The possible values are: • `None` • `Read` • `Edit` This field must be set to an access level that is higher than the organization's default access level for opportunities.
`OwnerId`	**Type** reference **Properties** Create, Filter **Description** Required. ID of the User who owns the default account team.
`TeamMemberRole`	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** Role that the team member has on opportunities for which the user has added his or her default account team. The valid values are set by the organization's administrator in the Account Team Roles picklist. Label is **Team Role**.
`UserId`	**Type** reference **Properties** Create, Filter **Description** Required. ID of the User who is a member of the default account team. This field cannot be updated.

Usage

This object is available only in organizations that have enabled the account teams functionality, which can be done using the user interface.

If you attempt to create a record that matches an existing record, the create call updates any modified fields and returns the existing record.

You can set up a User record so the default account team includes the others who typically work with them on accounts.

UserConfigTransferButton

Represents the association between a Live Agent configuration and a live chat button. This association allows users associated with a specific configuration to transfer chats to a button queue.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`LiveChatButtonId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** The ID of the live chat button that agents can transfer chats to.
`LiveChatUserConfigId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the Live Agent configuration; agents associated with this configuration can transfer chats to the chat button indicated by the `LiveChatButtonId`.

UserConfigTransferSkill

Represents the association between a Live Agent configuration and a skill. This association allows users associated with a specific configuration to transfer chats to agents who have that skill.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`LiveChatUserConfigId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the Live Agent configuration; agents associated with this configuration can transfer chats to the chat button indicated by the `LiveChatButtonId`.
`SkillId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** The ID of the skill group that agents can transfer chats to.

UserFeed

Represents a single feed item in the feed displayed on a Chatter user profile feed. A user profile feed shows changes to a user record for fields that are tracked in feeds, as well as posts and comments about the record.

A user feed shows recent changes to a user record for any fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to users in the organization. This object is available in API version 18.0 and later.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

If you use the FeedComment object to comment on a user record, the user can delete the comment. For example, if John Smith makes a comment about Sasha Jones, Sasha can delete the comment.

You can delete all feed items you created. To delete feed items you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the User object
- "Moderate Chatter"

Note: Users with the "Moderate Chatter" permission can delete only the feed items and comments they see.

Fields

Field	Details
Body	**Type** textarea **Properties** Nillable, Sort **Description** The content of UserFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
CommentCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
ConnectionId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** When a PartnerNetworkConnection makes a change to a record that is tracked, the `CreatedBy` field contains the ID of the system administrator while the `ConnectionId` contains the ID of the PartnerNetworkConnection that made the change. Available if Salesforce to Salesforce is enabled for your organization.
ContentData	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
ContentDescription	**Type** textarea **Properties** Nillable, Sort

Field	Details
	Description The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`CreatedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
`FeedPostId`	**Type** reference

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.
`LastModifiedDate`	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, `LastModifiedDate` is the same as `CreatedDate`. If a FeedComment is inserted on that feed item, then `LastModifiedDate` becomes the `CreatedDate` for that FeedComment. Deleting the FeedComment does not change the `LastModifiedDate`. Ordering by `LastModifiedDate DESC` sorts the feed by both the most recent feed item or comment.

Field	Details
LikeCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
NetworkScope	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
ParentId	**Type** reference **Properties** Filter, Group, Sort

Field	Details
	Description ID of the account record that is tracked in the feed. The feed is displayed on the detail page for this record.
RelatedRecordId	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a ContentPost. This field is null for all posts except ContentPost.
Title	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the user record. When the Type is LinkPost, the LinkUrl is the URL and this field is the link name.
Type	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • ActivityEvent—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • AdvancedTextPost—created when a user posts a group announcement. • AnnouncementPost—Not used. • ApprovalPost—generated when a user submits an approval. • BasicTemplateFeedItem—Not used. • CanvasPost—a post made by a canvas app posts on a feed. • CollaborationGroupCreated—generated when a user creates a public group. • CollaborationGroupUnarchived—Not used. • ContentPost—a post with an attached file.

Field	Details
	• `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`:

Field	Details
	• For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to track changes for a user record.

See Also:

User
EntitySubscription
NewsFeed
UserProfileFeed

UserLicense

Represents a user license in your organization. A user license entitles a user to specific functionality and determines the profiles and permission sets available to the user.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Fields

Field	Details
`LicenseDefinitionKey`	**Type** string **Properties** Filter, Nillable **Description** A string that uniquely identifies a particular user license. Label is `License Def. ID`. Values are: **AUL**: corresponds to the Salesforce Platform user license **AUL1**: corresponds to the Salesforce Platform One user license **AUL_LIGHT** corresponds to the Salesforce Platform Light user license **FDC_ONE** corresponds to the Force.com - One App user license **FDC_SUB** corresponds to the Force.com App Subscription user license **High Volume Customer Portal_User**: corresponds to the High Volume Customer Portal user license

Field	Details
	Overage_Platform_Portal_User corresponds to the Overage Authenticated Website user license **PID_STRATEGIC_PRM**: corresponds to the Gold Partner user license **PID_CHATTER** corresponds to the Chatter Only user license **PID_CONTENT** corresponds to the Content Only user license **PID_Customer_Portal_Basic**: corresponds to the Customer Portal Manager Standard user license and the Customer Portal User license **PID_Customer_Portal_Standard**: corresponds to the Customer Portal Manager Custom user license **PID_FDC_FREE** corresponds to the Force.com Free user license **PID_IDEAS** corresponds to the Ideas Only user license **PID_Ideas_Only_Portal** corresponds to the Ideas Only Portal user license **PID_Ideas_Only_Site** corresponds to the Ideas Only Site user license **PID_KNOWLEDGE** corresponds to the Knowledge Only user license **PID_Customer_Community** corresponds to the Customer Community license. **PID_Customer_Community_Login** corresponds to the Customer Community Login license. **PID_Partner_Community** corresponds to the Partner Community license. **PID_Partner_Community_Login** corresponds to the Partner Community Login license. **PID_Limited_Customer_Portal_Basic**: corresponds to the Limited Customer Portal Manager Standard user license **PID_Limited_Customer_Portal_Standard**: corresponds to the Limited Customer Portal Manager Custom user license **PID_Overage_Customer_Portal_Basic**: corresponds to the Overage Customer Portal Manager Standard user license **PID_Overage_High Volume Customer Portal** corresponds to the Overage High Volume Customer Portal user license **Platform_Portal_User**: corresponds to the Authenticated Website user license **POWER_PRM**: corresponds to the Partner user license **POWER_SSP**: corresponds to the Customer Portal Manager user license **SFDC**: corresponds to the Full CRM user license
`MonthlyLoginsEntitlement`	**Type** int **Properties** Filter, Nillable

Field	Details
	Description The maximum number of customer or partner portal logins allowed per month. A `null` value in this field means the user license is charged according to the number of users rather than the number of logins. This field is available in API version 20.0 and later.
`MonthlyLoginsUsed`	**Type** int **Properties** Nillable **Description** The number of successful logins for all users associated with a customer or partner portal user license. This field has a non-`null` value if `MonthlyLoginsEntitlement` has a non-`null` value. This field is available in API version 20.0 and later.
`Name`	**Type** string **Properties** Filter, Group, Sort **Description** The internal name of the user license. **Note:** Your organization may also include custom user licenses.

Usage

Users with the "View Setup and Configuration" permission can use the UserLicense object to view the set of currently defined user licenses in your organization.

The UserLicense object is currently used by bulk user creation to determine the user license to which each profile and permission set belongs. For example, if you use the API to create portal users and you want to know which profile belongs to each portal user license, you can query this object for each profile and check the `LicenseDefinitionKey` to identify the associated user license.

See Also:

Profile

UserLogin

Represents the settings that affect a user's ability to log into an organization. To access this object, you need the `UserPermissions.ManageUsers` permission. This object is available in API version 29.0 and later.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`, `update()`

Fields

Field Name	Details
`IsFrozen`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort, Update **Description** If `true`, the user account associated with this object is frozen.
`IsPasswordLocked`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort, Update **Description** If `true`, the user account associated with this object is locked because of too many login failures. From the API, you can set this field to `false`, but not `true`
`UserId`	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** ID of the associated user account. This field can't be updated.

Usage

To query for all frozen users in your organization:

```
SELECT Id, UserId
FROM UserLogin
WHERE IsFrozen = true
```

UserMembershipSharingRule

Represents the rules for sharing user records from a source group to a target group. A user record contains details about a user. Users who are members of the source group can be shared with members of the target group. The source and target groups can be based on roles, portal roles, public groups, or territories. This object is available in API version 26.0 and later.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`

Fields

Field	Details
`Description`	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** A description of the sharing rule. Maximum size is 1000 characters. This field is available in API version 29.0 and later.
`DeveloperName`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Rule Name** in the user interface. **Note:** When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`GroupId`	**Type** reference **Properties** Create, Filter, Group, Sort

Field	Details
	Description The ID representing the source group.
`Name`	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update **Description** Label of the sharing rule as it appears in the user interface. Limited to 80 characters. Corresponds to **Label** on the user interface.
`UserAccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** A value that represents the type of sharing being allowed. The possible values are: • `Read` • `Edit`
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID representing the target group being given access.

Usage

Use this object to manage sharing rules for user records. Source and target groups can include internal users, portal users, Chatter or Chatter External users.

UserPackageLicense

Represents a license for an installed managed package, assigned to a specific user. This object is available in API version 31.0 and later.

Supported Calls

`create()`, `delete()`, `query()`, `retrieve()`, `update()`

Fields

Field Name	Details
PackageLicenseId	**Type** reference **Properties** Filter, Group, Sort **Description** The 18-character Globally Unique ID (GUID) that identifies the package license
UserId	**Type** reference **Properties** Filter, Group, Sort **Description** The User ID of the user licensed to use this package

Usage

Use this object, in conjunction with PackageLicense, to provide users access to a managed package installed in your organization.

UserPreference

Represents a functional preference for a specific user in your organization.

Supported Calls

`describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
Preference	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the user preference. Supported values are `Event Reminder Default Lead Time`, `Task Reminder Default Time`, `Visualforce Viewstate Inspector`, and `Forecasting Displayed Type`. These values are related to

Field	Details
	`UserPreferencesEventRemindersCheckboxDefault` and `UserPreferencesTaskRemindersCheckboxDefault` on the User object.
`UserId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the user associated with this role. Label is **User ID**.
`Value`	**Type** string **Properties** Filter, Nillable, Sort **Description** The value of the user preference. For `Event Reminder Default Lead Time`, the values are increasing intervals of time from 0 minutes to 2 days. For `Task Reminder Default Time`, the values are half-hours from 12:00 AM to 11:30 PM. To view the respective sets of values, access the Reminders in your personal settings in the online application.

Usage

Use this object to query the set of currently configured user preferences in your organization. In your client application, you can query the User object to obtain valid User IDs in order to access the UserPreference object.

All users can invoke `query()` or `describeSObjects()` with this object.

UserProfile

Represents a Chatter user profile.

Supported Calls

`describeLayout()`, `query()`, `retrieve()`

Special Access Rules

- Information in hidden fields in a user's profile is not searchable by external users (with a portal profile) in a community. For example, if a user in a community has a hidden email address and an external user searches for it, the user record is not returned in the search results. Hidden field values are also not returned when external users perform searches on non-hidden fields. So if an external user searches for a user's name (cannot be hidden), any hidden field values associated with the user record (for example, a hidden email address) are not returned in the search results.

  Internal users from your organization who belong to the same community, however, can both search for and view hidden field values in search results.

- Any fields that have been restricted in visibility will be returned empty, whether or not they are, and will not be removed from the field listing.

Fields

Field	Details
AboutMe	**Type** textarea **Properties** Filter, Nillable, Sort **Description** Information about the user, such as areas of interest or skills.
Address (beta)	**Type** address **Properties** Filter, Nillable **Description** The compound form of the address. Read-only. See Address Compound Fields for details on compound address fields.
City	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The city associated with the user profile.
CompanyName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The company associated with the user profile.
Country	**Type** string **Properties** Filter, Group, Nillable, Sort

Field	Details
	Description The country associated with the user profile.
Email	**Type** email **Properties** Filter, Group, idLookup, Sort **Description** The email address associated with the user profile.
Fax	**Type** phone **Properties** Filter, Group, Nillable, Sort **Description** The fax number associated with the user profile.
FirstName	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The user’s first name.
FullPhotoUrl	**Type** url **Properties** Filter, Nillable, Sort **Description** The URL for the user's profile photo if Chatter is enabled. The URL is updated every time a photo is uploaded and reflects the most recent photo. The URL returned for an older photo is not guaranteed to return a photo if a newer photo has been uploaded. You should always query this field for the URL of the most recent photo.
IsActive	**Type** boolean

Field	Details
	Properties Defaulted on create, Filter, Group, Sort **Description** Indicates whether the user has access to log in (`true`) or not (`false`). You can modify a User's active status from the user interface or via the API.
`IsBadged`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the user is visually badged (`true`) or not (`false`). Users of the same Chatter user type (internal, external) are badged. Different user types are not badged.
`LastName`	**Type** string **Properties** Filter, Group, Sort **Description** The user's last name.
`Latitude` (beta)	**Type** double **Properties** Filter, Nillable, Sort **Description** Used with `Longitude` to specify the precise geolocation of an address. Acceptable values are numbers between –90 and 90 with up to 15 decimal places.
`Longitude` (beta)	**Type** double **Properties** Filter, Nillable, Sort **Description** Used with `Latitude` to specify the precise geolocation of an address. Acceptable values are numbers between –180 and 180 with up to 15 decimal places.

Field	Details
ManagerId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** The ID of the user who manages this user.
MobilePhone	**Type** phone **Properties** Filter, Group, Nillable, Sort **Description** The user’s mobile or cellular phone number.
Name	**Type** string **Properties** Filter, Group, Sort **Description** Concatenation of FirstName and LastName.
Phone	**Type** phone **Properties** Filter, Group, Nillable, Sort **Description** The user’s phone number.
PostalCode	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The user’s postal or ZIP code. Label is **Zip/Postal Code**.
SmallPhotoUrl	**Type** url

Field	Details
	Properties Filter, Nillable, Sort **Description** The URL for a thumbnail of the user's profile photo if Chatter is enabled. The URL is updated every time a photo is uploaded and reflects the most recent photo. The URL returned for an older photo is not guaranteed to return a photo if a newer photo has been uploaded. You should always query this field for the URL of the most recent photo.
`State`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The state associated with the user profile.
`Street`	**Type** textarea **Properties** Filter, Group, Nillable, Sort **Description** The street address associated with the user profile.
`Title`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The user's business title, such as "Vice President."
`UserPreferencesActivityRemindersPopup`	**Type** boolean **Properties** Filter **Description** When `true`, a reminder popup window automatically opens when an activity reminder is due. Corresponds to the `Trigger`

Field	Details
	`alert when reminder comes due` checkbox at the Reminders page in the personal settings in the user interface.
`UserPreferencesApexPagesDeveloperMode`	**Type** boolean **Properties** Filter **Description** When `true`, indicates the user has enabled developer mode for editing Visualforce pages and controllers.
`UserPreferencesDisableAllFeedsEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email for all updates to Chatter feeds, based on the types of feed emails and digests the user has enabled.
`UserPreferencesDisableBookmarkEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email every time someone comments on a Chatter feed item after the user has bookmarked it.
`UserPreferencesDisableChangeCommentEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email every time someone comments on a change the user has made, such as an update to their profile.
`UserPreferencesDisableEndorsementEmail`	**Type** boolean

Field	Details
	Properties Filter **Description** When `false`, the member automatically receives email every time someone endorses them for a topic.
`UserPreferencesDisableFeedbackEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives emails related to Work.com feedback. This includes when someone requests or offers feedback, shares feedback with the user, or reminds the user to answer a feedback request.
`UserPreferencesDisableFileShareNotificationsForApi`	**Type** boolean **Properties** Filter **Description** When `false`, email notifications are sent from the person who has shared a file to the users with whom the file has been shared.
`UserPreferencesDisableFollowersEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email every time someone starts following the user in Chatter.
`UserPreferencesDisableLaterCommentEmail`	**Type** boolean **Properties** Filter

Field	Details
	Description When `false`, the user automatically receives email every time someone comments on a feed item after the user has commented on the feed item.
`UserPreferencesDisableLikeEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email every time someone likes a post or comment the user has made.
`UserPreferencesDisableMentionsPostEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email every time the user is mentioned in posts.
`UserPreferencesDisableMessageEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email for Chatter messages sent to the user.
`UserPreferencesDisableProfilePostEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email every time someone posts to the user's profile.
`UserPreferencesDisableRewardEmail`	**Type** boolean

Field	Details
	Properties Filter **Description** When `false`, the user automatically receives emails related to Work.com rewards. This includes when someone someone gives a reward to the user.
`UserPreferencesDisableSharePostEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email every time the user's post is shared.
`UserPreferencesDisableWorkEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user receives emails related to Work.com feedback, goals, and coaching. The user must also sign up for individual emails listed on the Work.com email settings page. When `true`, the user will not receive any emails related to Work.com feedback, goals, or coaching even if they are signed up for individual emails.
`UserPreferencesDisCommentAfterLikeEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email every time someone comments on a post the user has liked.
`UserPreferencesDisMentionsCommentEmail`	**Type** boolean **Properties** Filter

Field	Details
	Description When `false`, the user automatically receives email every time the user is mentioned in comments.
`UserPreferencesDisProfPostCommentEmail`	**Type** boolean **Properties** Filter **Description** When `false`, the user automatically receives email every time someone comments on posts on the user's profile.
`UserPreferencesEnableAutoSubForFeeds`	**Type** boolean **Properties** Filter **Description** When `true`, the user automatically subscribes to feeds for any objects that the user creates.
`UserPreferencesEventRemindersCheckboxDefault`	**Type** boolean **Properties** Filter **Description** When `true`, a reminder popup is automatically set on the user's events. Corresponds to the `By default, set reminder on Events to...` checkbox on the Reminders page in the user interface. This field is related to UserPreference and customizing activity reminders.
`UserPreferencesHideChatterOnboardingSplash`	**Type** boolean **Properties** Filter **Description** When `true`, the initial Chatter onboarding prompts do not appear.
`UserPreferencesHideCSNDesktopTask`	**Type** boolean

Field	Details
	Properties Filter **Description** When `true`, the Chatter recommendations panel never displays the recommendation to install Chatter Desktop.
`UserPreferencesHideCSNGetChatterMobileTask`	**Type** boolean **Properties** Filter **Description** When `true`, the Chatter recommendations panel never displays the recommendation to install Chatter Mobile.
`UserPreferencesHideS1BrowserUI`	**Type** boolean **Properties** Filter **Description** Controls the interface that the user sees when logging in to Salesforce from a supported mobile browser. If `false`, the user is automatically redirected to the Salesforce1 mobile browser app. If `true`, the user sees the full Salesforce site. The default value is `false`. Label is **Salesforce1 User**. This field is available in API version 29.0 or later.
`UserPreferencesHideSecondChatterOnboardingSplash`	**Type** boolean **Properties** Filter **Description** When `true`, the secondary Chatter onboarding prompts do not appear.
`UserPreferencesReminderSoundOff`	**Type** boolean **Properties** Filter

Field	Details
	Description When true, a sound automatically plays when an activity reminder is due. Corresponds to the Play a reminder sound checkbox on the Reminders page in the user interface.
UserPreferencesShowCityToExternalUsers	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the city field in the user's contact information. City is visible only to internal members of the user's organization when: • This field is false. When false, this field returns the value #N/A. City is visible to external members in a community when: • This field is true, or • This field is false but UserPreferencesShowCityToGuestUsers is true, which overrides this field's value. External users are users with Community, Customer Portal, or partner portal licenses. The default value is false. This field is available in API version 26.0 and later.
UserPreferencesShowCityToGuestUsers	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the city field in the user's contact information. When true, city is visible to guest users. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When false, this field returns the value #N/A. When true, this field overrides the value false in UserPreferencesShowCityToExternalUsers, making the user's city visible to external members. The default value is false. This field is available in API version 28.0 and later.

Field	Details
`UserPreferencesShowCountryToExternalUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the country field in the user's contact information. Country is visible only to internal members of the user's organization when: • This field is `false`. When `false`, this field returns the value `#N/A`. Country is visible to external members in a community when: • This field is `true`, or • This field is `false` but `UserPreferencesShowCountryToGuestUsers` is `true`, which overrides this field's value. External users are users with Community, Customer Portal, or partner portal licenses. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowCountryToGuestUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the country field in the user's contact information. When `true`, country is visible to guest users. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the value `#N/A`. When `true`, this field overrides the value `false` in `UserPreferencesShowCountryToExternalUsers`, making the user's country visible to external members. The default value is `false`. This field is available in API version 28.0 and later.
`UserPreferencesShowEmailToExternalUsers`	**Type** boolean **Properties** Filter

Field	Details
	Description Indicates the visibility of the email address field in the user's contact information. Email address is visible only to internal members of the user's organization when this field is `false`. Email address is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowFaxToExternalUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the fax number field in the user's contact information. Fax number is visible only to internal members of the user's organization when this field is `false`. Fax number is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowManagerToExternalUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the manager field in the user's contact information. Manager is visible only to internal members of the user's organization when this field is `false`. Manager is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowMobilePhoneToExternalUsers`	**Type** boolean

Field	Details
	Properties Filter **Description** Indicates the visibility of the mobile or cellular phone number field in the user's contact information. The number is visible only to internal members of the user's organization when this field is `false`. The number is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowPostalCodeToExternalUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the postal or ZIP code field in the user's contact information. Postal code is visible only to internal members of the user's organization when: • This field is `false`. When `false`, this field returns the value `#N/A`. Postal code is visible to external members in a community when: • This field is `true`, or • This field is `false` but `UserPreferencesShowPostalCodeToGuestUsers` is `true`, which overrides this field's value. External users are users with Community, Customer Portal, or partner portal licenses. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowPostalCodeToGuestUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the postal or ZIP code field in the user's contact information. When `true`, postal code is visible to guest users. Guest users can access public Site.com and

Field	Details
	Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the value `#N/A`. When `true`, this field overrides the value `false` in `UserPreferencesShowPostalCodeToExternalUsers`, making the user's postal code visible to external members. The default value is `false`. This field is available in API version 28.0 and later.
`UserPreferencesShowProfilePicToGuestUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the user's profile photo. When `true`, the photo is visible to guest users in a community. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the stock photo. The default value is `false`. This field is available in API version 28.0 and later.
`UserPreferencesShowStateToExternalUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the state field in the user's contact information. State is visible only to internal members of the user's organization when: • This field is `false`. When `false`, this field returns the value `#N/A`. State is visible to external members in a community when: • This field is `true`, or • This field is `false` but `UserPreferencesShowStateToGuestUsers` is `true`, which overrides this field's value. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.

Field	Details
`UserPreferencesShowStateToGuestUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the state field in the user's contact information. When `true`, state is visible to guest users. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the value `#N/A`. When `true`, this field overrides the value `false` in `UserPreferencesShowStateToExternalUsers`, making the user's state visible to external members. The default value is `false`. This field is available in API version 28.0 and later.
`UserPreferencesShowStreetAddressToExternalUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the street address field in the user's contact information. The address is visible only to internal members of the user's organization when this field is `false`. The address is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.
`UserPreferencesShowTitleToExternalUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the business title field in the user's contact information. Title is visible only to internal members of the user's organization when: • This field is `false`. When `false`, this field returns the value `#N/A`.

Field	Details
	Title is visible to external members in a community when: • This field is `true`, or • This field is `false` but `UserPreferencesShowTitleToGuestUsers` is `true`, which overrides this field's value. External users are users with Community, Customer Portal, or partner portal licenses. The default value is `true`. This field is available in API version 26.0 and later.
`UserPreferencesShowTitleToGuestUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the business title field in the user's contact information. When `true`, title is visible to guest users. Guest users can access public Site.com and Force.com sites, and public pages in Communities, via the Guest User license associated with each site or community. When `false`, this field returns the value `#N/A`. When `true`, this field overrides the value `false` in `UserPreferencesShowTitleToExternalUsers`, making the user's title visible to external members. The default value is `false`. This field is available in API version 28.0 and later.
`UserPreferencesShowWorkPhoneToExternalUsers`	**Type** boolean **Properties** Filter **Description** Indicates the visibility of the work phone number field in the user's contact information. The number is visible only to internal members of the user's organization when this field is `false`. The number is visible to external members in a community when this field is `true`. External users are users with Community, Customer Portal, or partner portal licenses. When `false`, this field returns the value `#N/A`. The default value is `false`. This field is available in API version 26.0 and later.

Field	Details
UserPreferencesTaskRemindersCheckboxDefault	**Type** boolean **Properties** Filter **Description** When `true`, a reminder popup is automatically set on the user's tasks. Corresponds to the `By default, set reminder on Tasks to...` checkbox on the Reminders page in the user interface. This field is related to UserPreference and customizing activity reminders.

Usage

Use this object to query Chatter—related information about the user. While the User object contains all the information about a user and is historically tied to user management, UserProfile is a read-only entity that contains the information that is relevant in a Chatter context.

UserProfileFeed

Represents a user profile feed, which tracks all actions by a user on records that can be tracked in a feed. This feed is displayed on the user profile page.

UserProfileFeed is available in API version 18.0 through API 26.0. In API version 27.0 and later, UserProfileFeed is no longer available in the SOAP API. Use the Chatter REST API to access UserProfileFeed.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
Body	**Type** textarea **Properties** Nillable, Sort **Description** The content of UserProfileFeed. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
CommentCount	**Type** int **Properties** Filter, Group, Sort

Field	Details
	Description The number of FeedComments associated with this feed item.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort

Field	Details
	Description Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
CreatedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was created. This is a standard system field. Ordering by `CreatedDateDESC` sorts the feed by the most recent feed item.
FeedPostId	**Type** reference **Properties** Filter, Group, Nillable, Sort **Description** This field was removed in API version 22.0, and is available in earlier versions for backward compatibility only. ID of the associated FeedPost. A FeedPost represents the following types of changes in an FeedItem: changes to tracked fields, text posts, link posts, and content posts.
InsertedById	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then the `InsertedBy` value is set to the ID of the logged-in user.
IsDeleted	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the record has been moved to the Recycle Bin (`true`) or not (`false`). This is a standard system field.

Field	Details
LastModifiedDate	**Type** dateTime **Properties** Defaulted on create, Filter, Sort **Description** Date and time when this record was last modified by a user. This is a standard system field. When a feed item is created, LastModifiedDate is the same as CreatedDate. If a FeedComment is inserted on that feed item, then LastModifiedDate becomes the CreatedDate for that FeedComment. Deleting the FeedComment does not change the LastModifiedDate. Ordering by LastModifiedDate DESC sorts the feed by both the most recent feed item or comment.
LikeCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of a LinkPost.
NetworkScope	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. NetworkScope can have the following values: • NetworkId—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community.

Field	Details
	• AllNetworks—The feed item is available in all communities. Note the following exceptions for NetworkScope: • Only feed items with a Group or User parent can set a NetworkId or a null value for NetworkScope. • For feed items with a record parent, users can set NetworkScope only to AllNetworks. • You can't filter a FeedItem on the NetworkScope field.
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the account record that is tracked in the feed. The feed is displayed on the detail page for this record.
RelatedRecordId	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a ContentPost. This field is null for all posts except ContentPost.
Title	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the user profile record. When the Type is LinkPost, the LinkUrl is the URL and this field is the link name.
Type	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item: • ActivityEvent—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on

Field	Details
	cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `CaseCommentPost`—generated event when a user adds a case comment for a case object • `EmailMessageEvent`—generated event when an email related to a case object is sent or received • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `ChangeStatusPost`—generated event when a user changes the status of a case • `AttachArticleEvent`—generated event when a user attaches an article to a case **Note:** If you set `Type` to `ContentPost`, you must specify `ContentData` and `ContentFileName`.
`Visibility`	**Type** picklist

Field	Details
	Properties Filter, Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available to all users or internal users only. This field is available in API version 26.0 and later, if Salesforce Communities is enabled for your organization. `Visibility` can have the following values: • `AllUsers`—The feed item is available to all users who have permission to see the feed item. • `InternalUsers`—The feed item is available to internal users only. Note the following exceptions for `Visibility`: • For record posts, `Visibility` is set to `InternalUsers` for all internal users by default. • External users can set `Visibility` only to `AllUsers`. • On user and group posts, only internal users can set `Visibility` to `InternalUsers`.

Usage

Use this object to query and retrieve record changes tracked in a user profile feed. Note the following when working with user profile feeds:

- This object is read only in the API.
- Queries retrieve feed items that include mentions from other users.
- Use this object to query and retrieve lead feed items that were associated with a converted lead record.
- You must include a `WITH` clause and specify the `UserId` of the user whose profile you want to query. The `WITH` clause must come after a `WHERE` clause.

Users that do not have the “View All Data” permission have the following limitations when querying records:

- Must specify a `LIMIT` clause and the limit must be less than or equal to 1000.
- May include a `WHERE` clause that references UserProfileFeed fields, but may not include references to fields in related objects. For example, you can filter by `CreatedDate` or `ParentId`, but not by `Parent.Name`.
- May include an `ORDER BY` clause that references UserProfileFeed fields, but may not include references to fields in related objects. For example, you can `ORDER BY CreatedDate` or `ParentId`, but not by `Parent.Name`.

Tip: To query for the most recent feed items, you should `ORDER BY CreatedDate DESC, Id DESC`.

Note the following SOQL restrictions.

- No SOQL limit if logged-in user has “View All Data” permission. If not, specify a LIMIT clause of 1,000 records or fewer.

- SOQL ORDER BY on fields using relationships is not available. Use ORDER BY on fields on the root object in the SOQL query.

See Also:

EntitySubscription
FeedComment
FeedTrackedChange

UserRecordAccess

Represents a user's access to a set of records. This object is read only and is available in API version 24.0 and later.

Supported Calls

`describeSObjects()`, `query()`

Fields

Field	Details
`HasAllAccess`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether a user has all access–read, edit, delete, and transfer—to the record (`true`) or not (`false`).
`HasDeleteAccess`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether a user has delete access to the record (`true`) or not (`false`).
`HasEditAccess`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether a user has edit access to the record (`true`) or not (`false`).
`HasTransferAccess`	**Type** boolean

Field	Details
	Properties Defaulted on create, Filter, Group, Sort **Description** Indicates whether a user has transfer access to the record (`true`) or not (`false`).
`HasReadAccess`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether a user has read access to the record (`true`) or not (`false`).
`MaxAccessLevel`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Indicates a user's maximum level of access to a record. Valid values are: • `None` • `Read` • `Edit` • `Delete` • `Transfer` • `All`
`RecordId`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** ID of the record.
`UserId`	**Type** reference **Properties** Filter, Group

Field	Details
	Description ID of the user.

Usage

Use this object to query a user's access to records. You can't create, delete, or update any records using this object.

Up to 200 record IDs can be queried. You can include an `ORDER BY` clause for any field that is being selected in the query.

The following sample query returns the records, whether the queried user has read and transfer access to each record, and the user's maximum access level to each record.

```
SELECT RecordId, HasReadAccess, HasTransferAccess, MaxAccessLevel
     FROM UserRecordAccess
     WHERE UserId = [single ID]
     AND RecordId = [single ID]      //or Record IN [list of IDs]
```

The following query returns the records to which a queried user has read access.

```
SELECT RecordId
     FROM UserRecordAccess
     WHERE UserId = [single ID]
     AND RecordId = [single ID]      //or Record IN [list of IDs]
     AND HasReadAccess = true
```

Using API version 30.0 and later, UserRecordAccess is a foreign key on the records. You can't filter by or provide the `UserId` or `RecordId` fields when using this object as a lookup or foreign key. The previous sample queries can be run as:

```
SELECT Id, Name, UserRecordAccess.HasReadAccess, UserRecordAccess.HasTransferAccess,
UserRecordAccess.MaxAccessLevel
      FROM Account
```

```
SELECT Id, Name, UserRecordAccess.HasReadAccess
      FROM Account
```

SOQL restrictions for API version 29.0 and earlier:

- When the running user is querying a user's access to a set of records, records that the running user does not have read access to are filtered out of the results.
- When filtering by `UserId` and `RecordId` only, you must use `SELECT RecordId` and optionally one or more of the access level fields:`HasReadAccess`, `HasEditAccess`, `HasDeleteAccess`, `HasTransferAccess`, and `HasAllAccess`. You may include `MaxAccessLevel`.
- When filtering by `UserId`, `RecordId`, and an access level field, you must use `SELECT RecordId` only.

UserRole

Represents a user role in your organization.

Note: This object was called "Role" in previous versions of the API documentation.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`CaseAccessForAccountOwner`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The case access level for the account owner.
`ContactAccessForAccountOwner`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The contact access level for the account owner. **Note:** When `DefaultContactAccess` is set to `Controlled by Parent`, you can't create or update this field.
`DeveloperName`	**Type** string **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The unique name of the object in the API. This name can contain only underscores and alphanumeric characters, and must be unique in your organization. It must begin with a letter, not include spaces, not end with an underscore, and not contain two consecutive underscores. In managed packages, this field prevents naming conflicts on package installations. With this field, a developer can change the object's name in a managed package and the changes are reflected in a subscriber's organization. Corresponds to **Role Name** in the user interface. This field is available in API version 24.0 and later.

Field	Details
	Note: When creating large sets of data, always specify a unique `DeveloperName` for each record. If no `DeveloperName` is specified, performance may slow while Salesforce generates one for each record.
`ForecastUserId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the forecast manager associated with this role. Label is **User ID**.
`IsPartner`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the user role is a partner who has access to the partner portal (`true`) or not (`false`). This field is not available for release 9.0 and later. Instead, use `PortalType` with the value `Partner`.
`MayForecastManagerShare`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the forecast manager can manually share their own forecast.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name of the role. Corresponds to **Label** on the user interface.

Field	Details
OpportunityAccessForAccountOwner	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. The opportunity access level for the account owner. Note that you can't set a user role with an opportunity access less than that specified in organization-wide defaults.
ParentRoleId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the parent role.
PortalRole	**Type** picklist **Properties** Filter, Nillable **Description** The portal role: Executive, Manager, User, or PersonAccount.
PortalType	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort **Description** This value indicates the type of portal for the role: • None: Salesforce application role. • CustomerPortal: Customer portal role. • Partner: partner portal role. The field `IsPartner` used in release 8.0 will map to this value. This field replaces `IsPartner` beginning with release 9.0.
RollupDescription	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update

Field	Details
	Description Description of the forecast rollup. Label is **Description**.

Usage

Use this object to query the set of currently configured user roles in your organization. Use it in your client application to obtain valid UserRole IDs to use when querying or modifying a User record.

All users have access to invoke query or describe this object. If your client application logs in with the “Manage Users” permission, it can query, create, update, or delete UserRole records.

Note: You can’t update any field for a portal role.

For example, the following code finds all roles that are not assigned to any users.

```
SELECT Id, Name, DeveloperName,
FROM UserRole
WHERE Id NOT IN (SELECT UserRoleId
                 FROM User
                 WHERE UserRoleId !='000000000000000')
```

See Also:

Standard and Custom Object Basics

UserShare

Represents a sharing entry on a user record. This object is available in API version 26.0 and later.

Note: This object is available through a pilot program. For more information on enabling this object for your organization, contact salesforce.com.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

Customer Portal users can't access this object.

Fields

Field	Details
`IsActive`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort

Field	Details
	Description Read-only. Indicates whether the User has access to log in (`true`) or not (`false`). You can modify a User's active status from the user interface or via the API.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Read-only. Reason that this sharing entry exists. Possible values include: • `Manual Sharing`—The User or Group has access to the user record because a User with “All” access manually shared the User with them. • `Sharing Rule`—The User or Group has access to the user record via a User sharing rule.
`UserAccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Level of access that the User or Group has to the specified user. The specified user is denoted by the `UserId`. The possible values are: • `Read` • `Edit` This field must be set to an access level that is at least equal to the organization’s default `UserAccessLevel`. `UserAccessLevel` can be updated only if `RowCause` is set to `Manual Sharing`.
`UserId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the User being shared.
`UserOrGroupId`	**Type** reference

Field	Details
	Properties Create, Filter, Group, Sort **Description** ID of the User or Group that has been given access to the User. This field can't be updated.

Usage

This object allows you to determine which users and groups can view or edit User records owned by other users.

UserTeamMember

Represents a single User on the default opportunity team of another User.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- This object is available only in organizations that have enabled the team selling functionality.
- Customer Portal users can't access this object.

Fields

Field	Details
`OpportunityAccessLevel`	**Type** picklist **Properties** Create, Filter, Restricted picklist, Update **Description** Required. Level of access that the team member has to opportunities for which the user has added his or her default opportunity team. The possible values are: • `Read` • `Edit` This field must be set to an access level that is higher than the organization's default access level for opportunities.
`OwnerId`	**Type** reference **Properties** Create, Filter

Field	Details
	Description Required. ID of the User who owns the default opportunity team. This field can’t be updated.
`TeamMemberRole`	**Type** picklist **Properties** Create, Filter, Nillable, Update **Description** Role that the team member has on opportunities for which the User has added his or her default opportunity team. The valid values are set by the organization’s administrator in the Opportunity Team Roles picklist. Label is **Team Role**.
`UserId`	**Type** reference **Properties** Create, Filter **Description** Required. ID of the User who is a member of the default opportunity team. This field can’t be updated.

Usage

If you attempt to create a record that matches an existing record, the create request updates any modified fields and returns the existing record.

Users can set up their default opportunity team to include other users that typically work with them on opportunities.

See Also:

OpportunityTeamMember

UserTerritory

Represents a User who has been assigned to a Territory.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

- Only available if territory management has been enabled for your organization.
- Customer Portal users can’t access this object.

Fields

Field	Details
`IsActive`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the user is active in the given territory (`true`), or inactive in the given territory (`false`): • Users who are active in a territory are explicitly assigned to the territory and can have open opportunities, closed opportunities, or no opportunities associated with that territory. • Users who are inactive in a territory are not explicitly assigned to the territory, but own an open or closed opportunity that is associated with the territory. For example, a user may have been transferred out of a territory, but still own opportunities in his or her old territory. Until a user is deleted from a territory (not simply removed from the territory), the record is not returned in a `getDeleted()` call.
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (`true`) or not (`false`). Label is **Deleted**.
`TerritoryId`	**Type** reference **Properties** Create, Filter **Description** ID of the Territory to which the user has been assigned. This field is required when creating a record in API version 20.0 and later.
`UserId`	**Type** reference **Properties** Create, Filter **Description** ID of the user. This field is required when creating a record.

Usage

If a user is inactive in a territory, and the opportunities they own that are associated with the territory are all closed, then the user is not returned.

See Also:

Territory
AccountTerritoryAssignmentRule
AccountTerritoryAssignmentRuleItem

UserTerritory2Association

Represents an association (by assignment) between a territory and a user record. Available only if Territory Management 2.0 has been enabled for your organization.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`

Special Access Rules

If a territory model is in `Active` state, any user can view that model, including its territories and assignment rules. For territories in an active model, any user can view assigned records and assigned users subject to your organization's sharing settings. Users cannot view territory models in other states (such as `Planning` or `Archived`).

Fields

Field Name	Details
`IsActive`	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Indicates whether the user is active (`true`) or inactive (`false`) in the given territory.
`Territory2Id`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the territory that the user is assigned to.
`UserId`	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Sort **Description** The ID of the user who is assigned to the territory.

Vote

Represents a vote that a user has made on an Idea or a Reply.

Note: In API version 16.0 and earlier, SOQL queries on the Vote object only return votes for the Idea object. Starting in API version 17.0, SOQL queries return votes for both Idea and Reply.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field	Details
`IsDeleted`	**Type** boolean **Properties** Defaulted on create, Filter **Description** Indicates whether the object has been moved to the Recycle Bin (true) or not (false). Label is **Deleted**.
`ParentId`	**Type** reference **Properties** Group, Sort, Create, Filter **Description** ID of the Idea or Reply associated with this vote.
`Type`	**Type** picklist **Properties** Group, Sort, Create, Filter, Restricted picklist

Field	Details
	Description Picklist that indicates the type of vote. The value `Up` indicates that the vote is a user's positive endorsement of the associated idea or reply. The value `Down` indicates that the vote is a user's negative endorsement of the associated idea or reply.

Note: If you are importing Vote data into Salesforce and need to set the value for an audit field, such as `CreatedDate`, contact salesforce.com. Audit fields are automatically updated during API operations unless you request to set these fields yourself..

Usage

In version 12.0 and later, use this object to track the votes that users made on ideas. For more information on ideas, see "Ideas Overview" in the Salesforce online help.

In version 17.0 and later, use this object to track the votes users made on replies. For more information, see "Answers Overview" in the Salesforce online help.

In version 17.0 and later, you must filter using the following syntax when querying this object in a SOQL query: `ParentId = `***`single ID`***, `Parent.Type = `***`single Type`***, `Id = `***`single ID`***, or `Id IN (`***`list of IDs`***). See Comparison Operators in the *Salesforce SOQL and SOSL Reference Guide* for a sample query.

A SOQL query must filter using one of the following Parent or Id clauses.

- `ParentId = [`***`single ID`***`]`
- `Parent.Type = [`***`single type`***`]`
- `Id = [`***`single ID`***`]`
- `Id IN = [`***`list of IDs`***`]`

See Also:

Idea
IdeaComment

WebLink

Represents a custom link to a URL or Scontrol.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- To create a custom link, the client application must be logged in with the "Customize Application" permission.
- Customer Portal users can't access this object.

Fields

Field Name	Details
Availability	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description**
Description	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Description of the custom link. Limit is 1,000 characters.
DisplayType	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Type of display: button, link, or mass-action button.
EncodingKey	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. Encoding of parameters on the URL link.
HasMenubar	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the popup window shows a menu bar (`true`) or not (`false`).

Field Name	Details
HasScrollbars	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the popup window shows scroll bars (true) or not (false).
HasToolbar	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the popup window shows browser toolbars (true) or not (false). Toolbars normally contain navigation buttons like Back, Forward, and Print.
Height	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Height of the popup in pixels.
IsProtected	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the object is protected (true) or not (false). Protected components that have been installed in other organizations can't be linked to or referenced by components created in the subscriber organization. A developer can easily delete a protected component contained in a managed package in a future release of the package without worrying about failing installations. However, once a component is marked as unprotected and is released globally, the developer can't delete it.
IsResizable	**Type** boolean

Field Name	Details
	Properties Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether users are allowed to resize the popup window (`true`) or not (`false`).
`LinkType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. Type of link (S-control or URL).
`MasterLabel`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Master label for the link. Limit is 240 characters. This display value is the internal label that is not translated.
`Name`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. Name to display on page.
`NamespacePrefix`	**Type** string **Properties** Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***`namespacePrefix__componentName`*** notation. The namespace prefix can have one of the following values:

Field Name	Details
	• In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects. This field can't be accessed unless the logged-in user has the "Customize Application" permission.
`OpenType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Required. How the custom link opens when clicked in a browser—NewWindow, Sidebar, or NoSidebar.
`PageOrSobjectType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** Required. For standard objects, the name of the page on which to display the custom link. For custom objects, the name of the object.
`Position`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Location on the screen where the popup should open—TopLeft, FullScreen, or None.
`RequireRowSelection`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update

Field Name	Details
	Description Indicates whether the custom link requires a row selection (`true`) or not (`false`).
`ScontrolId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the custom s-control object (Scontrol) to link to. Can include fields as tokens within the custom s-control object. Label is **Custom S-Control ID**.
`ShowsLocation`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the popup window shows the browser's address bar containing the URL (`true`) or not (`false`).
`ShowsStatus`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Show the status bar at the bottom of the browser.
`Url`	**Type** textarea **Properties** Create, Nillable, Update **Description** Required. URL of the page to link to. Can include fields as tokens within the URL. Limit: 1,024 KB.
`Width`	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description Width of the popup in pixels.

Usage

Use this object to programmatically manage custom links, which allow client applications to integrate data with external URLs, an organization's intranet, or other back-end office systems. A custom link can point to:

- An external URL, such as `www.google.com` or your company's intranet.
- A custom s-control, such as a Java applet or Active-X control.

Custom links can include fields as tokens within the URL or custom s-control.

See Also:

Scontrol

WebLinkLocalization

When the Translation Workbench is enabled for your organization, the WebLinkLocalization object provides the translation of the field label of a custom link to a URL or s-control. For information on the Translation Workbench, see the Salesforce online help.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `update()`, `upsert()`

Special Access Rules

- Your organization must be using Professional, Enterprise, Developer, Unlimited, or Performance Edition and be enabled for the Translation Workbench.
- To view this object, you must have the "View Setup and Configuration" permission.

Fields

Field	Details
`LanguageLocaleKey`	**Type** picklist **Properties** Create, Filter, Nillable, Restricted picklist **Description** This field is available in API version 16.0 and earlier. It is the same as the `Language` field.

Field	Details
Language	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** This field is available in API version 17.0 and later. The combined language and locale ISO code, which controls the language for labels displayed in an application. This picklist contains the following fully-supported languages: • Chinese (Simplified): `zh_CN` • Chinese (Traditional): `zh_TW` • Danish: `da` • Dutch: `nl_NL` • English: `en_US` • Finnish: `fi` • French: `fr` • German: `de` • Italian: `it` • Japanese: `ja` • Korean: `ko` • Portuguese (Brazil): `pt_BR` • Russian: `ru` • Spanish: `es` • Spanish (Mexico): `es_MX`* * Spanish (Mexico) falls back to Spanish for customer-defined translations. • Swedish: `sv` • Thai: `th`* * Even though the Salesforce interface is fully translated to Thai, Help remains in English. The following end-user only languages are available. • Arabic: `ar` • Bulgarian: `bg` • Czech: `cs` • English (UK): `en_GB` • Greek: `el` • Hebrew: `iw` • Hungarian: `hu` • Indonesian: `in` • Norwegian: `no` • Polish: `pl` • Romanian: `ro` • Turkish: `tr`

Field	Details
	• Ukrainian: `uk` • Vietnamese: `vi` The following platform languages are available for organizations that use Salesforce exclusively as a platform. • Albanian: `sq` • Armenian: `hy` • Basque: `eu` • Bosnian: `bs` • Bengali: `bn` • Croatian: `hr` • English (Australia): `en_AU` • English (Canada): `en_CA` • English (India): `en_IN` • English (Malaysia): `en_MY` • English (Philippines): `en_PH` • Estonian: `et` • French (Canada): `fr_CA` • Georgian: `ka` • German (Austria): `de_AT` • German (Switzerland): `de_CH` • Hindi: `hi` • Icelandic: `is` • Irish: `ga` • Latvian: `lv` • Lithuanian: `lt` • Luxembourgish: `lb` • Macedonian: `mk` • Malay: `ms` • Maltese: `mt` • Romanian (Moldova): `ro_MD` • Montenegrin: `sh_ME` • Portuguese (European): `pt_PT` • Romansh: `rm` • Serbian (Cyrillic): `sr` • Serbian (Latin): `sh` • Slovak: `sk` • Slovenian: `sl` • Tagalog: `tl` • Tamil: `ta` • Urdu: `ur` • Welsh: `cy` The values in this field are not related to the default locale selection.
`NamespacePrefix`	**Type** string

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** The namespace prefix associated with this object. Each Developer Edition organization that creates a managed package has a unique namespace prefix. Limit: 15 characters. You can refer to a component in a managed package by using the ***namespacePrefix__componentName*** notation. The namespace prefix can have one of the following values: • In Developer Edition organizations, the namespace prefix is set to the namespace prefix of the organization for all objects that support it. There is an exception if an object is in an installed managed package. In that case, the object has the namespace prefix of the installed managed package. This is the namespace prefix of the Developer Edition organization of the package developer. • In organizations that are not Developer Edition organizations, `NamespacePrefix` is only set for objects that are part of an installed managed package. There is no namespace prefix for all other objects.
`Value`	**Type** string **Properties** Create, Filter, Nillable, Sort, Update **Description** The actual translated label of the custom link. Label is **Translation**.
`WebLinkId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** The ID of the WebLink that is being translated.

Usage

Use this object to translate your custom links to URLs or s-controls into the different languages supported by Salesforce. Users with the Translation Workbench enabled can view custom link translations, but either the “Customize Application” or “Manage Translation” permission is required to create or update custom link translations.

See Also:

CategoryNodeLocalization
ScontrolLocalization

WorkAccess

Used to grant or restrict user access to give badge definitions. Each badge definition record must have one WorkAccess record.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Additional Considerations and Related Objects

WorkAccess is not available through Schema Builder and is not customizable. A WorkAccess record is **required** for users to **Give** BadgeDefinitions. If a WorkAccess record is not created, BadgeDefinitions will not be available to users.

The sharing of WorkAccess records is through WorkAccessShare on page 1374 For each WorkBadgeDefinition record, you must create both a WorkAccess record (per WorkBadgeDefinition) and WorkAccessShare records for sharing to users or groups.

Fields

Field Name	Details
`AccessType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** Define the type of Access given to user (“Give”).
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Salesforce unique ID for owner of Access record.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** Salesforce unique ID for BadgeDefinition record associated with this Access record.

WorkAccessShare

WorkAccessShare is used to control Givers of WorkBadgeDefinition records.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Additional Considerations and Related Objects

Related to WorkAccess Object. WorkAccess is the parent of WorkAccessShare.

Fields

Field Name	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** CRUD Access Level (picklist values: Read Only, Read/Write, Owner).
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID for WorkAccess record.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Owner or Manual sharing.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** User or Group ID for WorkAccess.

WorkBadge

A WorkBadge record is created for each recipient of a WorkBadgeDefinition. Contains information about who the badge was given to and which badge was given.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Additional Considerations and Related Objects

WorkBadge is a lookup to WorkThanks. Each WorkBadge record must derive a SourceId from WorkThanks. There can be multiple WorkBadge records tied to a single WorkThanks record.

Fields

Field Name	Details
`DefinitionId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. Salesforce unique ID for the given WorkBadgeDefinition record given.
`NetworkId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the community that this WorkBadge is associated with. This field is available only if Salesforce Communities is enabled in your organization.
`RecipientId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Required. Salesforce unique ID for User who is the Recipient of Badge.
`RewardId`	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** Salesforce unique ID for Reward given with badge (if Reward Badge)
`SourceId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** Salesforce unique ID for Thanks record referenced to this badge.

WorkBadgeDefinition

The attributes of a badge including the badge name, description, and image. Each WorkBadge record must have a lookup to a WorkBadgeDefinition as badge attributes, such as badge name, badge description, and badge image, are derived from the WorkBadgeDefinition object.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Additional Considerations and Related Objects

WorkBadgeDefinition has a field called `ImageUrl` that references a DocumentID. This is a required field for creating a Badge.

To grant "giver" access to a WorkBadgeDefinition, you must also create the WorkAccess (and the related WorkAccessShare) records.

Each WorkBadgeDefinition has an `ImageUrl` field that must be populated with a DocumentID of the Document record containing the badge image.

Fields

Field Name	Details
`Description`	**Type** textarea **Properties** Create, Update **Description** Required. Limit: 4000 characters. The description of the badge and what it means to receive this badge.

Field Name	Details
ImageUrl	**Type** url **Properties** Create, Filter, Group, Sort, Update **Description** Required. This is the badge image that will be displayed in the UI. Use DocumentID or ImageURL.
IsActive	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Represents whether a WorkBadgeDefinition is active and available in the UI.
IsCompanyWide	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Represents a special class of badges known as Company Badges. Company badges are visible to the entire company and visible in specific list view filters.
IsLimitPerUser	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the badge limit is per user (`true`) or across all users (`false`). The default value is `false`.
LimitNumber	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The badge limit per user or across all users.

Field Name	Details
LimitStartDate	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** The start date of the badge limit. The date can be reset to the current date.
Name	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update **Description** Required. Name of the Badge. **Label:** Badge Title.
NetworkId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the community that this WorkBadgeDefinition is associated with. This field is available only if Salesforce Communities is enabled in your organization.
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Salesforce User ID for User who is the Owner of the WorkBadgeDefinition record (usually the creator of the record)
RewardFundId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description Salesforce unique ID for the WorkRewardFund that is associated with this WorkBadgeDefinition. WorkBadgeDefinition records with a RewardFundID indicate a Reward Badge.

WorkBadgeDefinitionHistory

Represents the history of changes to the values in the fields of a WorkBadgeDefinition object.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Type of change (e.g., created).
`NewValue`	**Type** Any Type **Properties** Nillable, Sort **Description** Updated value of record.
`OldValue`	**Type** Any Type **Properties** Nillable, Sort **Description** Previous value of record.
`WorkBadgeDefinitionId`	**Type** reference

Field Name	Details
	Properties Filter, Group, Sort **Description** ID of WorkBadgeDefinition record.

WorkBadgeDefinitionShare

Represents a sharing entry on a WorkBadgeDefinition object.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The user's or group's level of access to the work badge definition. The possible values are: • Read • Edit • All: This value is not valid when you create, update, or delete records This field must be set to an access level that is higher than the organization's default access level for work badge definitions.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the WorkBadgeDefinition object that is associated with this sharing entry.
`RowCause`	**Type** picklist

Field Name	Details
	Properties Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Valid values include: • Owner: User is the owner of the WorkBadgeDefinition or is in a user role above the WorkBadgeDefinition owner in the role hierarchy. • Manual: User or group has access, because a user with “All” access manually shared the WorkBadgeDefinition with the user or group. • Rule: User or group has access via a WorkBadgeDefinition sharing rule.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the user or group that was given access to the work badge definition. This field can’t be updated.

WorkCoaching

Represents a single coaching relationship between two users. One of the users is defined as the coach and the other is defined as a coachee. WorkCoaching is feed-enabled so there is a private feed available to the coach and coachee.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`,`undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
`CoachId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** [Required] The coach in this 1:1 coaching relationship.

Field Name	Details
CoachedId	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** [Required] The user being coached in this 1:1 coaching relationship.
IsInactive	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the coaching relationship is `Inactive` (`true`) or not (`false`).
LastReferencedDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The time stamp that indicates when the current user last viewed a record that is related to this coaching relationship.
LastViewedDate	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The time stamp that indicates when the current user last viewed this coaching relationship. If this value is null, this record might have been only referenced (`LastReferencedDate`) and not viewed.
Name	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** [Required] The record’s name. Max length is 255 characters.

Field Name	Details
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the contact who owns the WorkCoaching record.

WorkCoachingFeed

Represents a single feed item in the feed on the detail page for a coaching record. A coaching feed shows recent changes to a lead record for any fields that are tracked in feeds, as well as posts and comments about the record. It is a useful way to stay up-to-date with changes made to coaching in Salesforce.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Special Access Rules

You can delete all feed items that you created. To delete feed items that you didn't create, you must have one of these permissions:

- "Modify All Data"
- "Modify All" on the WorkCoachingFeed object
- "Moderate Chatter"

Fields

Field Name	Details
Body	**Type** textarea **Properties** Nillable, Sort **Description** The content of the FeedItem. Required when Type is TextPost. Optional when Type is ContentPost or LinkPost. This is the message that appears in the feed.
CommentCount	**Type** int **Properties** Filter, Group, Sort

Field Name	Details
	Description The number of FeedComments associated with this feed item.
ContentData	**Type** base64 **Properties** Nillable **Description** The file that is uploaded to the feed. Required if Type is ContentPost. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets Type to ContentPost.
ContentDescription	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in ContentData.
ContentFileName	**Type** string **Properties** Group, Nillable, Sort **Description** The file that is uploaded to the feed. Required if Type is ContentPost. The name of the file that is uploaded to the feed. Setting ContentFileName automatically sets Type to ContentPost.
ContentSize	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) that is uploaded to the feed.
ContentType	**Type** string **Properties** Group, Nillable, Sort

Field Name	Details
	Description Read only. Automatically determined during insert. The MIME type of the file that is uploaded to the feed.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, InsertedById is set to the ID of the logged-in user.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes that are associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a LinkPost.
`NetworkScope`	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. If Salesforce Communities is enabled for your organization, this field is available in API version 28.0 and later. `NetworkScope` can have the following values. • `NetworkId`: The ID of the community in which the FeedItem is available. If left empty, the feed item is available only in the default community.

Field Name	Details
	• `AllNetworks`: The feed item is available in all communities. Note the following exceptions for `NetworkScope`. • Only feed items with a CollaborationGroup or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the WorkCoaching record that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a ContentPost. This field is null for all posts except ContentPost. For example, set this field to an existing ContentVersion and post it to a feed as a FeedItem object of type ContentPost.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the FeedItem. When the Type is LinkPost, the LinkUrl is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort

Field Name	Details
	Description The type of feed item. With the exception of `ContentPost`, `LinkPost`, and `TextPost`, feed item types should not be created directly from the API • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls.

Field Name	Details
	• `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case. **Note:** If you set Type to ContentPost, you must specify ContentData and ContentFileName.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The visibility of a FeedItem: • AllUsers: visible to all users • InternalUsers: visible to internal users

WorkCoachingHistory

Represents the history of changes to the values in the fields of a WorkCoaching object.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist

Field Name	Details
	Properties Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.
`WorkCoachingId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the WorkCoaching object.

WorkCoachingShare

Represents a sharing entry on a WorkCoaching object.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
AccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The user's or group's level of access to the coaching relationship. The possible values are: • Read • Edit • All: This value is not valid when you create, update, or delete records. This field must be set to an access level that is higher than the organization's default access level for coaching relationships.
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the WorkCoaching object that is associated with this sharing entry.
RowCause	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Valid values include: • Owner: User is the owner of the coaching relationship or is in a user role above the coaching relationship owner in the role hierarchy. • Manual: User or group has access, because a user with "All" access manually shared the coaching relationship with the user or group. • Rule: User or group has access via a sharing rule.
UserOrGroupId	**Type** reference **Properties** Create, Filter, Group, Sort

Field Name	Details
	Description ID of the user or group that was given access to the coaching relationship. This field can't be updated.

WorkFeedback

Answer to a specific question that the person was asked via a request for feedback. Also used to store offered feedback without linking it to a particular question.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Additional Considerations and Related Objects

- Ownership is transferred to the requester on submit for certain types (ad-hoc feedback).
- The record is read-only after the request that it's linked to is set to Submitted.
- You can't link a feedback object to a request unless you are the recipient.
- The question that the feedback is linked to must be part of the same question set that the request is linked to.

Warning: Do not use the API to manage performance summary answers, and instead change answers in a performance summary cycle through the Work.com UI in Salesforce. The business logic that is associated with answering a performance summary question (that is, sharing rules as defined by the recipient and the management hierarchy and validation checks on specific answers), may not work if the API is used. Some adverse effects include:

- Creating and linking questions in the API could create multiple questions on the same question set.
- Editing a question in the API could impact the UI and the Calibration feature without the right steps and order of operations.

Fields

Field Name	Details
`Feedback`	**Type** textarea **Properties** Create, Nillable, Update **Description** Contains either the free-form text of the answer, or the choice selected by the user. Max length is 65536.
`OwnerId`	**Type** reference

Field Name	Details
	Properties Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the owner of the WorkFeedback record.
`QuestionId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The question this answer applies to. When this feedback is linked to a request of an unsolicited type, the question ID is null.
`RequestId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the request this response belongs to, in case of offered feedback.

WorkFeedbackQuestion

Represents a question within a set of questions of either a free form text type, or a multiple choice.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Additional Considerations and Related Objects

Warning: Do not use the API to manage performance summary answers, and instead change answers in a performance summary cycle through the Work.com UI in Salesforce. The business logic that is associated with answering a performance summary question (that is, sharing rules as defined by the recipient and the management hierarchy and validation checks on specific answers), may not work if the API is used. Some adverse effects include:

- Creating and linking questions in the API could create multiple questions on the same question set.
- Editing a question in the API could impact the UI and the Calibration feature without the right steps and order of operations.

Fields

Field Name	Details
Choices	**Type** textarea **Properties** Create, Nillable, Update **Description** Comma-separated list of valid choices for multiple choice questions. Maximum length is 1000 characters.
Detail	**Type** textarea **Properties** Create, Nillable, Update **Description** Detailed instructions on how to answer the question.
IsConfidentialAnswer	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Answers to questions marked confidential will not be shared with the subject of the review. This field applies only to performance summaries.
IsOptional	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** If this option is selected, the question is optional and isn't required to be answered. This field applies only to performance summaries.
Name	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update

Field Name	Details
	Description A short description of the question, which can be used as a header for reports and Calibration.
`Number`	**Type** int **Properties** Create, Filter, Group, Sort, Update **Description** The order of the question that is displayed within the question set, such as question number three in a question set that has five questions.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the owner of the WorkFeedbackQuestion.
`QuestionSetId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** The question set this question is a part of.
`Text`	**Type** textarea **Properties** Create, Update **Description** The body of the question. Max length is 16384 characters.
`Type`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update

Field Name	Details
	Description Allows for either a free-form text answer or a multiple choice question defined by new-line separate choices in the 'Choices' field. Valid picklist values are: • MultipleChoice • FreeText

WorkFeedbackQuestionHistory

Represents the history of changes to the values in the fields of a WorkFeedbackQuestion.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort

Field Name	Details
	Description The latest value of the field before it was changed.
`WorkFeedbackQuestionId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the WorkFeedbackQuestion.

WorkFeedbackQuestionSet

Represents a set of questions that are being asked. The question set is used to link all the individual requests where different recipients were asked the same set of questions on the same subject.

In the Work.com performance application, a question set defines the type of summaries and their due dates that will accompany the deployment of a specific performance summary cycle.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Additional Considerations and Related Objects

Warning: Do not use the API to manage performance summary answers, and instead change answers in a performance summary cycle through the Work.com UI in Salesforce. The business logic that is associated with answering a performance summary question (that is, sharing rules as defined by the recipient and the management hierarchy and validation checks on specific answers), may not work if the API is used. Some adverse effects include:

- Creating and linking questions in the API could create multiple questions on the same question set.
- Editing a question in the API could impact the UI and the Calibration feature without the right steps and order of operations.

Fields

Field Name	Details
`DueDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description The date that this specific question set is expected to be submitted by the recipient. This field applies only to performance summaries.
`FeedbackType`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The description of the collection of questions that are written in context to the type of recipient answering them, relative to the subject of the summary. This field applies only to performance summaries.
`Name`	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update **Description** The name of the question set. Maximum length is 225 characters.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the owner of the WorkFeedbackQuestionSet.
`PerformanceCycleId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** If a question set is associated to a performance summary cycle, then that cycle ID is referenced in this field. This field applies only to performance summaries.

WorkFeedbackQuestionSetShare

Represents a sharing entry on a WorkFeedbackQuestionSet.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The user's or group's level of access to the feedback question set. The possible values are: • Read • Edit • All: This value is not valid when you create, update, or delete records. This field must be set to an access level that is higher than the organization's default access level for feedback question sets.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the WorkFeedbackQuestionSet object that is associated with this sharing entry.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Valid values include:

Field Name	Details
	• Owner: User is the owner of the WorkFeedbackQuestionSet or is in a user role above the WorkFeedbackQuestionSet owner in the role hierarchy. • Manual: User or group has access, because a user with "All" access manually shared the WorkFeedbackQuestionSet with the user or group. • Rule: User or group has access via a WorkFeedbackQuestionSet sharing rule.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the user or group that was given access to the feedback question set. This field can't be updated.

WorkFeedbackQuestionShare

Represents a sharing entry on a WorkFeedbackQuestion.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The user's or group's level of access to the feedback question. The possible values are: • Read • Edit • All: This value is not valid when you create, update, or delete records. This field must be set to an access level that is higher than the organization's default access level for feedback questions.

Field Name	Details
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the WorkFeedbackQuestion object that is associated with this sharing entry.
RowCause	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Valid values include: • Owner: User is the owner of the WorkFeedbackQuestion or is in a user role above the WorkFeedbackQuestion owner in the role hierarchy. • Manual: User or group has access, because a user with “All” access manually shared the WorkFeedbackQuestion with the user or group. • Rule: User or group has access via a WorkFeedbackQuestion sharing rule.
UserOrGroupId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the user or group that was given access to the feedback question. This field can’t be updated.

WorkFeedbackRequest

In the feedback application, WorkFeedbackRequest represents a single feedback request on a subject or topic (question) to a single recipient. In the case of offered feedback, WorkFeedbackRequest represents feedback that is offered about a subject. In the performance application, WorkFeedbackRequest represents a request for feedback on a set of questions from a question set, on a subject—for the recipient to complete and submit.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Additional Considerations and Related Objects

- After a request's state is changed to Submitted, fields can't be changed, except for LastSharedDate and IsUnreadByOwner.
- If LastRemindDate is updated, a reminder notification will be sent to the request's recipient (only possible when request is in Draft state).
- When a new request is created, a notification is sent to the recipient.
- When a recipient of a request submits their feedback (Draft->Submitted), a notification will be sent to requester (except for offered feedback).
- Requester cannot modify the subject of the question set after a request is created.
- For offered feedback (to user, to manager, or both), the person who is offering feedback is both the creator of WorkFeedbackRequest as well as the recipient.

Warning: Do not use the API to manage performance summary answers, and instead change answers in a performance summary cycle through the Work.com UI in Salesforce. The business logic that is associated with answering a performance summary question (that is, sharing rules as defined by the recipient and the management hierarchy and validation checks on specific answers), may not work if the API is used. Some adverse effects include:

- Creating and linking questions in the API could create multiple questions on the same question set.
- Editing a question in the API could impact the UI and the Calibration feature without the right steps and order of operations.

Fields

Field Name	Details
`FeedbackRequestState`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Restricted picklist, Sort, Update **Description** The current state of the feedback request. Allowed picklist values are: • Draft • Submitted • Dismissed
`FeedbackType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort **Description** Specifies the type of request. Picklist values that are used for performance summaries: • Peer Summary

Field Name	Details
	• Self Summary • Manager Summary • Skip Level Summary Picklist values that are used for feedback: • Personal • Unsolicited to User • Unsolicited to Manager • Unsolicited to User and Manager • On Topic The type of the feedback determines the sharing and visibility rules that are applied to answers.
`IsUnreadByOwner`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** If `true`, the submitted request has not been seen by the requester.
`LastRemindDate`	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** The last time a reminder was sent to the recipient of this draft request.
`LastSharedDate`	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** The last time this request was shared with another user or group.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the owner of the WorkFeedbackRequest.

Field Name	Details
PerformanceCycleId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Used by performance summaries to link to a summary cycle. This field applies only to performance summaries.
QuestionSetId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** Question set associated with the current request.
RecipientId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** User asked to provide feedback on the subject.
RelatedObjectId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** Specifies a record in the system that this feedback request is related to. Used by ad-hoc feedback to gather feedback in the context of an opportunity or Work.com goal. Used by performance summaries to link to a summary cycle.
SubjectId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort **Description** The ID of the user that this request (or offer) is about.

Field Name	Details
SubmitFeedbackToId	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the person this performance summary feedback request (and its respective answers) is shared with. It's also the ID of the person who owns the requested subject's manager summary request. This field applies only to performance summaries.
SubmittedDate	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** The last time (in case it was reopened by admin) this request was submitted by the recipient. This field applies only to performance summaries.

WorkFeedbackRequestHistory

Represents the history of changes to the values in the fields of a WorkFeedbackRequest.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
Field	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
NewValue	**Type** anyType **Properties** Nillable, Sort

Field Name	Details
	Description The new value of the field that was changed.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.
`WorkFeedbackRequestId`	**Type** picklist **Properties** Filter, Group, Sort **Description** ID of the WorkFeedbackRequest.

WorkFeedbackRequestShare

Represents a sharing entry on a WorkFeedbackRequest.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The user's or group's level of access to the feedback request. The possible values are: • Read • Edit • All: This value is not valid when you create, update, or delete records.

Field Name	Details
	This field must be set to an access level that is higher than the organization’s default access level for feedback requests.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the WorkFeedbackRequest object that is associated with this sharing entry.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Valid values include: • Owner: User is the owner of the WorkFeedbackRequest or is in a user role above the WorkFeedbackRequest owner in the role hierarchy. • Manual: User or group has access, because a user with “All” access manually shared the WorkFeedbackRequest with the user or group. • Rule: User or group has access via a WorkFeedbackRequest sharing rule.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the user or group that was given access to the feedback request. This field can’t be updated.

WorkFeedbackShare

Represents a sharing entry on a WorkFeedback object.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
AccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The user's or group's level of access to the feedback. The possible values are: • Read • Edit • All: This value is not valid when you create, update, or delete records. This field must be set to an access level that is higher than the organization's default access level for feedback.
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the WorkFeedback object that is associated with this sharing entry.
RowCause	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Valid values include: • Owner: User is the owner of the WorkFeedback or is in a user role above the WorkFeedback owner in the role hierarchy. • Manual: User or group has access, because a user with "All" access manually shared the WorkFeedback with the user or group. • Rule: User or group has access via a WorkFeedback sharing rule.
UserOrGroupId	**Type** reference **Properties** Create, Filter, Group, Sort

Field Name	Details
	Description ID of the user or group that was given access to the feedback. This field can't be updated.

WorkGoal

The WorkGoal object represents the components of a goal such as its description and associated metrics.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Related

WorkGoalCollaborator, WorkGoalLink, WorkGoalFeed

Fields

Field Name	Details
`ActualValue`	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** The actual value of the WorkGoal metric. Applicable only to WorkGoal objects of `Type`: Metric.
`ActualValueExternalUrl`	**Type** url **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Contains a URL that references work.com data synchronization for the actual value of a metric. Applicable only to WorkGoal objects of `Type`: Metric.
`CompletionDate`	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update

Field Name	Details
	Description The completion date of the goal. **Note:** Field-level security limits access to only administrators and owners by default, and only they can complete a goal.
`Description`	**Type** textarea (max length 4000) **Properties** Create, Nillable, Update **Description** The description of the goal.
`DueDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The date the WorkGoal object is due (optional). Applicable only to WorkGoal objects of `Type`: Metric.
`FlaggedAs`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The progress of the WorkGoal object. Applicable only to WorkGoal objects of `Type`: Metric. Possible values: • On Track: Progress on the metric is on track. • Behind: Progress on the metric is behind schedule. • Postponed: The metric is postponed. • Critical: Progress on the metric is critical.
`ImageUrl`	**Type** url **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description The URL for the goal image. Applicable only to WorkGoal objects of `Type`: Goal.
`InitialValue`	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** The initial value of the WorkGoal metric. Applicable only to WorkGoal objects of `Type`: Metric and `MetricType`: Progress or Percent.
`IsKeyCompanyGoal`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Used to indicate if the goal is a key company goal. Used for the Company Goal Showcase. Applicable only to WorkGoal objects of `Type`: Goal.
`LastReferencedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort **Description** The time stamp that indicates when the current user last viewed a record that is related to this goal.
`LastSyncDate`	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** The time stamp that indicates when the actual value was last synced with the associated metrics report.
`LastViewedDate`	**Type** dateTime **Properties** Filter, Nillable, Sort

Field Name	Details
	Description The time stamp that indicates when the current user last viewed this goal.
`MetricType`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The type of metric that is represented. (See values in the following list). Applies only to WorkGoal objects of `Type`: Metric. Possible values: • Progress: ActualValue / TargetValue as a percentage • Percent: the metric as a percentage only • YesNo: the completed / not completed metric as a milestone • Absolute: Deprecated
`MetricTypeDataSource`	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** Specifies how the metric (ActualValue and CurrentValue) is updated. Applies only to WorkGoal objects of `Type`: Goal and Metric. Possible values: • Manual: indicates that the actual and target value of the metric is updated manually by the user • Rollup: indicates that the actual and target value of a goal is rolled up automatically by Work.com Goals • DataSyncActualOnly: indicates that the actual value of the metric is linked to a Salesforce report
`Name`	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update **Description** The name of the WorkGoal object. (Maximum length is 255.)
`OverallStatus`	**Type** string

Field Name	Details
	Properties Filter, Group, Nillable, Sort **Description** The overall calculated status of the WorkGoal based on `FlaggedAs` and `CompletionDate`.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the user who owns the WorkGoal.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Specifies the structural parent of the WorkGoal. For example, a goal that has a metric is represented by a WorkGoal of `Type` Metric, which has a parent of WorkGoal of Type Goal. **Note:** The root and the parent must be set to the parent goal for any child metrics.
`Progress`	**Type** percent **Properties** Filter, Nillable, Sort **Description** Read Only. The overall progress of the WorkGoal.
`RootId`	**Type** reference to a WorkGoal object **Properties** Create, Filter, Group, Nillable, Sort, Update

Field Name	Details
	Description Specifies the structural root of the WorkGoal. For example, a goal that has a metric is represented by a WorkGoal of `Type` Metric, which has a root of WorkGoal of `Type` Goal.
`State`	**Type** picklist **Properties** Create, Defaulted on create, Filter, Group, Restricted picklist, Sort, Update **Description** The state of the WorkGoal object. Applies only to WorkGoal objects of `Type`: Metric. Possible values: • Draft: the draft state for the WorkGoal • Published: published state for the WorkGoal • Archived: archived state for the WorkGoal (for example, goals that no longer apply)
`TargetValue`	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** The target value of the WorkGoal. Applies only to WorkGoal objects of `Type`: Metric.
`Type`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The type of the WorkGoal object, used to differentiate between the components of a goal. (This field is used to represent components of a goal such as its description and associated metrics.) Possible values: • Goal: a goal • Metric: a metric (typically associated with goals) • Objective: an objective • KeyResult: a key result (typically associated with objectives • V2Mom: a V2MOM (pilot feature) • Vision: a vision (pilot feature — typically associated with V2MOM)

Field Name	Details
	• Value: a value (pilot feature - typically associated with V2MOM) • Method: a method (pilot feature - typically associated with V2MOM) • Obstacle: an obstacle (pilot feature - typically associated with V2MOM) • Measure: a measure (pilot feature - typically associated with a method) **Note:** Administrators can rename goals and metrics to objectives and key results, respectively. If this preference is enabled, use the `Type` Objective or KeyResult. Otherwise, use the default `Type` Goal or KeyResult.
`Weight`	**Type** double **Properties** Create, Filter, Nillable, Sort, Update **Description** The weight of the goal or metric. The sum of the weights should equal 100%.

WorkGoalCollaborator

The WorkGoalCollaborator object represents collaborators on a WorkGoal object. Note that this doesn't include WorkGoal followers, which is handled by Chatter Feed Follow functionality.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
`InvitationDate`	**Type** date **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The date that a user was invited to become a collaborator (nill if the user was not invited).
`State`	**Type** picklist

Field Name	Details
	Properties Create, Defaulted on create, Filter, Group, Restricted picklist, Sort, Update **Description** Indicates the state of the collaborating user. Whether the user has not responded, joined, or declined collaboration. The possible values are: • PendingResponse: a user who was invited to collaborate but hasn't joined or declined • Joined: a user who is collaborating on a goal (joined/commit) • Declined: a user who declined to collaborate on a goal
`UserId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The collaborating user.
`WorkGoalId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The WorkGoal object that this collaborator is a part of.

WorkGoalCollaboratorHistory

This read-only object contains historical information about changes that were made to the WorkGoalCollaborator object.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort

Field Name	Details
	Description Name of the standard or custom field.
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** New value of the modified field.
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** Previous value of the modified field.
`WorkGoalCollaboratorId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the WorkGoalCollaborator object that is associated with this history entry.

WorkGoalFeed

Represents a single feed item in the feed displayed on the goal page for a Goal record. The goal feed shows changes to a goal for fields that are tracked in feeds, posts, and comments about the goal, as well as updates on metrics.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
Body	**Type** textarea **Properties** Nillable, Sort **Description** The content of the FeedItem. Required when Type is TextPost. Optional when Type is ContentPost or LinkPost. This is the message that appears in the feed.
CommentCount	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
ContentData	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if Type is ContentPost. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets Type to ContentPost.
ContentDescription	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in ContentData.
ContentFileName	**Type** string **Properties** Group, Nillable, Sort

Field Name	Details
	Description The file uploaded to the feed. Required if Type is ContentPost. The name of the file uploaded to the feed. Setting ContentFileName automatically sets Type to ContentPost.
`ContentSize`	**Type** int **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then InsertedById is set to the ID of the logged-in user.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.

Field Name	Details
LinkUrl	**Type** url **Properties** Nillable, Sort **Description** The URL of a LinkPost.
NetworkScope	**Type** picklist **Properties** Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. If Salesforce Communities is enabled for your organization, this field is available in API version 28.0 and later. NetworkScope can have the following values: • NetworkId—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • AllNetworks—The feed item is available in all communities. Note the following exceptions for NetworkScope: • Only feed items with a Group or User parent can set a NetworkId or a null value for NetworkScope. • For feed items with a record parent, users can set NetworkScope only to AllNetworks. • You can't filter a FeedItem on the NetworkScope field.
ParentId	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the account record that is tracked in the feed. The feed is displayed on the detail page for this record.
RelatedRecordId	**Type** reference

Field Name	Details
	Properties Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a ContentPost. This field is null for all posts except ContentPost. For example, set this field to an existing ContentVersion and post it to a feed as a FeedItem object of TypeContentPost.
`Title`	**Type** string **Properties** Group, Nillable, Sort **Description** The title of the FeedItem. When the Type is LinkPost, the LinkUrl is the URL and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item. With the exception of `ContentPost`, `LinkPost`, and `TextPost`, feed item types should not be created directly from the API • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used.

Field Name	Details
	• `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply. • `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The visibility of a FeedItem:

Field Name	Details
	• AllUsers: visible to all users • InternalUsers: visible to internal users

WorkGoalHistory

This read-only object contains historical information about changes that have been made to the WorkGoal object.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
`NewValue`	**Type** Any Type **Properties** Nillable, Sort **Description** The new value of the field that was changed.
`OldValue`	**Type** Any Type **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.
`WorkGoalId`	**Type** reference

Field Name	Details
	Properties Filter, Group, Sort **Description** ID of the Goal. Label is Goal ID.

WorkGoalLink

Represents the relationship between two goals (many to many relationship).

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()` , `upsert()`

Fields

Field Name	Details
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Whether the WorkGoalLink is active (`true`) or not (`false`)
`LinkType`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The type of link
`Name`	**Type** string **Properties** Autonumber, Defaulted on create, Filter, idLookup, Sort **Description** The auto-generated name of the goal link

Field Name	Details
`SourceGoalId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** The ID of the source WorkGoal object
`TargetGoalId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** The ID of the target WorkGoal object

WorkGoalShare

Represents a sharing entry on a WorkGoal object.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The user's or group's level of access to the goal. The possible values are: • Read • Edit • All: This value is not valid when you create, update, or delete records This field must be set to an access level that is higher than the organization's default access level for goals.

Field Name	Details
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the WorkGoal object that is associated with this sharing entry.
RowCause	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Valid values include: • Owner: User is the owner of the WorkGoal or is in a user role above the WorkGoal owner in the role hierarchy. • Manual: User or group has access, because a user with "All" access manually shared the WorkGoal with the user or group. • Rule: User or group has access via a WorkGoal sharing rule.
UserOrGroupId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the user or group that was given access to the goal. This field can't be updated.

WorkPerformanceCycle

WorkPerformanceCycle represents feedback that is gathered for the purpose of assessing the performance of a specific set of employees.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Additional Considerations and Related Objects

Warning: Do not use the API to manage performance summary answers, and instead change answers in a performance summary cycle through the Work.com UI in Salesforce. The business logic that is associated with answering a performance summary question (that is, sharing rules as defined by the recipient and the management hierarchy and validation checks on specific answers), may not work if the API is used. Some adverse effects include:

- Creating and linking questions in the API could create multiple questions on the same question set.
- Editing a question in the API could impact the UI and the Calibration feature without the right steps and order of operations.

Fields

Field Name	Details
ActivityFrom	**Type** date **Properties** Create, Filter, Group, Sort, Update **Description** The date that you want to start filtering the work.com objects to help requesters create accurate summaries. The start of the evaluation period.
ActivityTo	**Type** date **Properties** Create, Filter, Group, Sort, Update **Description** The date that you want to stop filtering the work.com objects to help requesters create accurate summaries. The end of the evaluation period.
LastManagerRequestsSharedDate	**Type** dateTime **Properties** Create, Filter, Nillable, Sort, Update **Description** The date when all manager requests are set to be shared.
Name	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update

Field Name	Details
	Description The name of the performance summary cycle that employees will participate in. This name is created by the administrator and is visible on all respective notifications and in the UI.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** ID of the owner of the WorkPerformanceCycle.
`State`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The state that the performance summary cycle is in. Available pick list values: • Setup: The summary is in draft. • In Progress: The summary is deployed and people are answering the questions that were created. • Finished: The summary is no longer in progress. • Process: The summary is processing. • Error: The summary encountered an error.

WorkPerformanceCycleFeed

Represents a single feed item in the feed that is displayed on a Work.com Performance detail page.

Supported Calls

`delete()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Body`	**Type** textarea **Properties** Nillable, Sort

Field Name	Details
	Description The content of the FeedItem. Required when `Type` is `TextPost`. Optional when `Type` is `ContentPost` or `LinkPost`. This is the message that appears in the feed.
`CommentCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedComments associated with this feed item.
`ContentData`	**Type** base64 **Properties** Nillable **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. Encoded file data. Cannot be 0 bytes. Can be any format. Setting this field automatically sets `Type` to `ContentPost`.
`ContentDescription`	**Type** textarea **Properties** Nillable, Sort **Description** The description of the file specified in `ContentData`.
`ContentFileName`	**Type** string **Properties** Group, Nillable, Sort **Description** The file uploaded to the feed. Required if `Type` is `ContentPost`. The name of the file uploaded to the feed. Setting `ContentFileName` automatically sets `Type` to `ContentPost`.
`ContentSize`	**Type** int

Field Name	Details
	Properties Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The size of the file (in bytes) uploaded to the feed.
`ContentType`	**Type** string **Properties** Group, Nillable, Sort **Description** Read only. Automatically determined during insert. The MIME type of the file uploaded to the feed.
`InsertedById`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the user who added this object to the feed. For example, if a client application migrates multiple posts and comments from another application into a feed, then `InsertedById` is set to the ID of the logged-in user.
`LikeCount`	**Type** int **Properties** Filter, Group, Sort **Description** The number of FeedLikes associated with this feed item.
`LinkUrl`	**Type** url **Properties** Nillable, Sort **Description** The URL of a `LinkPost`.
`NetworkScope`	**Type** picklist

Field Name	Details
	Properties Group, Nillable, Restricted picklist, Sort **Description** Specifies whether this feed item is available in the default community, a specific community, or all communities. If Salesforce Communities is enabled for your organization, this field is available in API version 30.0 and later. `NetworkScope` can have the following values: • `NetworkId`—The ID of the community in which the FeedItem is available. If left empty, the feed item is only available in the default community. • `AllNetworks`—The feed item is available in all communities. Note the following exceptions for `NetworkScope`: • Only feed items with a Group or User parent can set a `NetworkId` or a null value for `NetworkScope`. • For feed items with a record parent, users can set `NetworkScope` only to `AllNetworks`. • You can't filter a FeedItem on the `NetworkScope` field.
`ParentId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of the group that is tracked in the feed. The feed is displayed on the detail page for this record.
`RelatedRecordId`	**Type** reference **Properties** Group, Nillable, Sort **Description** ID of the ContentVersion object associated with a `ContentPost`. This field is null for all posts except `ContentPost`. For example, set this field to an existing ContentVersion and post it to a feed as a FeedItem object of `Type ContentPost`.
`Title`	**Type** string

Field Name	Details
	Properties Group, Nillable, Sort **Description** The title of the FeedItem. When the `Type` is `LinkPost`, the `LinkUrl` is the URL, and this field is the link name.
`Type`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The type of feed item. With the exception of `ContentPost`, `LinkPost`, and `TextPost`, feed item types should not be created directly from the API • `ActivityEvent`—indirectly generated event when a user or the API adds a Task associated with a feed-enabled parent record (excluding email tasks on cases). Also occurs when a user or the API adds or updates a Task or Event associated with a case record (excluding email and call logging). For a recurring Task with CaseFeed disabled, one event is generated for the series only. For a recurring Task with CaseFeed enabled, events are generated for the series and each occurrence. • `AdvancedTextPost`—created when a user posts a group announcement. • `AnnouncementPost`—Not used. • `ApprovalPost`—generated when a user submits an approval. • `BasicTemplateFeedItem`—Not used. • `CanvasPost`—a post made by a canvas app posts on a feed. • `CollaborationGroupCreated`—generated when a user creates a public group. • `CollaborationGroupUnarchived`—Not used. • `ContentPost`—a post with an attached file. • `CreatedRecordEvent`—generated when a user creates a record from the publisher. • `DashboardComponentAlert`—generated when a dashboard metric or gauge exceeds a user-defined threshold. • `DashboardComponentSnapshot`—created when a user posts a dashboard snapshot on a feed. • `LinkPost`—a post with an attached URL. • `PollPost`—a poll posted on a feed. • `ProfileSkillPost`—generated when a skill is added to a user's Chatter profile. • `QuestionPost`—generated when a user posts a question. • `ReplyPost`—generated when Chatter Answers posts a reply.

Field Name	Details
	• `RypplePost`—generated when a user creates a Thanks badge in Work.com. • `TextPost`—a direct text entry on a feed. • `TrackedChange`—a change or group of changes to a tracked field. • `UserStatus`—automatically generated when a user adds a post. Deprecated. The following values appear in the `Type` picklist for all feed objects but apply only to CaseFeed: • `AttachArticleEvent`—generated event when a user attaches an article to a case. • `CallLogPost`—generated event when a user logs a call for a case through the user interface. This event is also generated by CTI calls. • `CaseCommentPost`—generated event when a user adds a case comment for a case object. • `ChangeStatusPost`—generated event when a user changes the status of a case. • `ChatTranscriptPost`—generated event when Live Agent transcript is saved to a case. • `EmailMessageEvent`—generated event when an email related to a case object is sent or received. • `FacebookPost`—generated when a Facebook post is created from a case. Deprecated. • `MilestoneEvent`—generated when a case milestone is completed or reaches violation status. • `SocialPost`—generated when a social post is created from a case.
`Visibility`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** The visibility of a FeedItem: • AllUsers: visible to all users • InternalUsers: visible to internal users

WorkPerformanceCycleHistory

Represents the history of changes to the values in the fields of a WorkPerformanceCycle.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
Field	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed.
NewValue	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed.
OldValue	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed.
WorkPerformanceCycleId	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the WorkPerformanceCycle.

WorkPerformanceCycleShare

Represents a sharing entry on a WorkPerformanceCycle object.

Supported Calls

`create(), delete(), describeSObjects(), query(), retrieve(), update(), upsert()`

Fields

Field Name	Details
AccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The user's or group's level of access to the work performance cycle. The possible values are: • Read • Edit • All: This value is not valid when you create, update, or delete records. This field must be set to an access level that is higher than the organization's default access level for work performance cycles.
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID of the WorkPerformanceCycle object that is associated with this sharing entry.
RowCause	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Reason that this sharing entry exists. Read-only. Valid values include: • Owner: User is the owner of the WorkPerformanceCycle or is in a user role above the WorkPerformanceCycle owner in the role hierarchy. • Manual: User or group has access, because a user with "All" access manually shared the WorkPerformanceCycle with the user or group. • Rule: User or group has access via a WorkPerformanceCycle sharing rule.
UserOrGroupId	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Sort **Description** ID of the user or group that was given access to the performance cycle share. This field can't be updated.

WorkReward

Used to store reward codes tied to a Reward Fund. Reward Funds must have at least one WorkReward record.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

You must have the Reward permission enabled in order to use the Rewards feature, including WorkRewardFund and WorkReward.

Additional Considerations and Related Objects

WorkReward is a lookup to WorkRewardFund. WorkRewardFund must have at least one WorkReward record to be available for use. Each WorkBadge record with a `RewardId` indicates a reward badge given to a Recipient.

Fields

Field Name	Details
`Code`	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Represents a singe reward code tied to a RewardFundId.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Represents the User ID of Owner of WorkReward record

Field Name	Details
`RecipientId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** Salesforce User ID for User associated with this WorkReward record.
`RewardFundId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** Salesforce unique ID for WorkRewardFund record that is associated with WorkReward record.
`RewardFundTypeId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** Salesforce unique ID of the WorkRewardFundType associated with the WorkReward.
`Value`	**Type** double **Properties** Filter, Nillable, Sort **Description** The value of the WorkReward.

WorkRewardFund

Represents a Reward Fund and describes the Reward Fund attributes.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Special Access Rules

To use the Rewards feature, including WorkRewardFund and WorkReward, you must have the Reward permission enabled. To create Rewards, the user must have Create on WorkRewardFund, which is not a standard permission.

Additional Considerations and Related Objects

WorkReward is a lookup to WorkRewardFund. WorkRewardFund must have at least one WorkReward record available. Each WorkBadgeDefinition with a RewardFundId is a “Reward Badge.”

Fields

Field Name	Details
`IsActive`	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the WorkRewardFund is active (`true`) or not (`false`).
`Name`	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update **Description** Required. Name of the Reward Fund.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Salesforce unique ID of User who is the Owner of the WorkRewardFund record.
`RewardFundTypeId`	**Type** reference **Properties** Create, Filter, Group, Sort, Update **Description** Salesforce unique ID of the WorkRewardFundType that is associated with the WorkRewardFund.

Field Name	Details
TotalCodeCount	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Total reward codes that are available in the WorkRewardFund. Derived from WorkReward records that are associated with the WorkRewardFund.
Type	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** RewardType of the WorkRewardFund. Default is Amazon.com.
UsedCodeCount	**Type** int **Properties** Filter, Group, Nillable, Sort **Description** Total reward codes that are used in the WorkRewardFund. Derived from the total assigned WorkReward records that are associated with the WorkRewardFund.
Value	**Type** currency **Properties** Create, Filter, Sort, Update **Description** Value of each of the reward codes in the WorkRewardFund.

WorkRewardFundHistory

Represents the history of changes to the values in the fields of a WorkRewardFund object.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Type of change (e.g., created).
`NewValue`	**Type** Any Type **Properties** Nillable, Sort **Description** Updated value of record.
`OldValue`	**Type** Any Type **Properties** Nillable, Sort **Description** Previous value of record.
`WorkRewardFundId`	**Type** reference **Properties** Filter, Group, Sort **Description** ID of WorkRewardFund record.

WorkRewardFundShare

Share records for WorkRewardFund.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
AccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** CRUD Access Level (picklist values: Read Only, Read/Write, Owner).
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID for WorkRewardFund record.
RowCause	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Owner or Manual sharing.
UserOrGroupId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** User or Group ID for WorkRewardFund record.

WorkRewardFundType

Represents the type of WorkRewardFund object.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Fields

Field Name	Details
CreditSystem	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The credit system that is used by the WorkRewardFundType object (gift codes or points). If points are selected, the reward message will not consider the `CurrencyCode` field.
CurrencyCode	**Type** picklist **Properties** Create, Filter, Group, Nillable, Restricted picklist, Sort, Update **Description** The currency code of the WorkRewardFundType
IsActive	**Type** boolean **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Whether the WorkRewardFundType is active and available in the UI
IsPredefined	**Type** boolean **Properties** Defaulted on create, Filter, Group, Sort **Description** Whether the WorkRewardFundType is predefined (`true`) or not (`false`)
Name	**Type** string **Properties** Create, Filter, Group, idLookup, Sort, Update **Description** The name of the WorkRewardFundType

Field Name	Details
OwnerId	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** The ID of the WorkRewardFundType owner
RedemptionDisclaimer	**Type** textarea **Properties** Create, Nillable, Update **Description** The redemption disclaimer text for the WorkRewardFundType
RedemptionInfo	**Type** textarea **Properties** Create, Nillable, Update **Description** Redemption text for the WorkRewardFundType
RedemptionUrl	**Type** textarea **Properties** Create, Nillable, Update **Description** The URL that's linked to the redemption
UploadCodeColumn	**Type** int **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The column where the reward code is contained in the CSV file. The upload uses the second value by default.
UploadValueColumn	**Type** int

Field Name	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The column where the reward value is contained in the CSV file. The upload uses the third column by default.

WorkRewardFundTypeHistory

Represents the history of changes to the values in the fields of a WorkRewardFundType object.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed
`NewValue`	**Type** anyType **Properties** Nillable, Sort **Description** The value of the changed field
`OldValue`	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed

Field Name	Details
`WorkRewardFundTypeId`	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the WorkRewardFundType

WorkRewardFundTypeShare

Represents a sharing entry on a WorkRewardFundType.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
`AccessLevel`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** The user's or group's level of access to the work reward fund type. The possible values are: • Read • Edit • All: This value is not valid when you create, update, or delete records. This field must be set to an access level that is higher than the organization's default access level for WorkRewardFundType objects.
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the WorkRewardFundType object that is associated with this sharing entry.

Field Name	Details
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The reason that this sharing entry exists. Read-only. Valid values include: • Owner: User is the owner of the WorkRewardFundType or is in a user role above the WorkRewardFundType owner in the role hierarchy. • Manual: User or group has access, because a user with “All” access manually shared the WorkRewardFundType with the user or group. • Rule: User or group has access via a WorkRewardFundType sharing rule.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** The ID of the user or group that was given access to the WorkRewardFundType. This field can’t be updated.

WorkRewardHistory

Represents the history of changes to the fields of a WorkReward.

Supported Calls

`describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`

Fields

Field Name	Details
`Field`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** The name of the field that was changed

Field Name	Details
NewValue	**Type** anyType **Properties** Nillable, Sort **Description** The new value of the field that was changed
OldValue	**Type** anyType **Properties** Nillable, Sort **Description** The latest value of the field before it was changed
WorkRewardId	**Type** reference **Properties** Filter, Group, Sort **Description** The ID of the WorkReward

WorkRewardShare

Share records for WorkReward object.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
AccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** CRUD Access Level (picklist values: Read Only, Read/Write, Owner).

Field Name	Details
`ParentId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID for WorkReward record.
`RowCause`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Owner or Manual sharing.
`UserOrGroupId`	**Type** reference **Properties** Create, Filter, Group, Sort **Description** User or Group ID for WorkReward record.

WorkThanks

Represents the source and message of a thanks post.

Supported Calls

`create()`, `delete()`, `describeLayout()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `undelete()`, `update()`, `upsert()`

Additional Considerations and Related Objects

WorkBadge is a lookup to WorkThanks. Each WorkBadge record must derive a SourceId from WorkThanks.

Fields

Field Name	Details
`GiverId`	**Type** reference

Field Name	Details
	Properties Create, Filter, Group, Sort **Description** Salesforce user ID for the giver of the Thanks record.
`Message`	**Type** textarea **Properties** Create **Description** Required. Message associated with the Thanks record.
`NetworkId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The ID of the community that this WorkThanks is associated with. This field is available only if Salesforce Communities is enabled in your organization.
`OwnerId`	**Type** reference **Properties** Create, Defaulted on create, Filter, Group, Sort, Update **Description** Salesforce user ID for the owner of the badge record (typically the same user as the giver of the record).

WorkThanksShare

Share records for WorkThanks object.

Supported Calls

`create()`, `delete()`, `describeSObjects()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Field Name	Details
AccessLevel	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update **Description** CRUD Access Level (picklist values: Read Only, Read/Write, Owner).
ParentId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** ID for WorkThanks record.
RowCause	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort **Description** Owner or Manual sharing.
UserOrGroupId	**Type** reference **Properties** Create, Filter, Group, Sort **Description** User or Group ID for WorkThanks record.

Chapter 9

Core Calls

The following table lists supported calls in the API in alphabetical order, and provides a brief description for each. Click a call name to see syntax, usage, and more information for that call.

Note: For a list of API utility calls, see Utility Calls, and for a list of describe calls, see Describe Calls.

Call	Description
`convertLead()`	Converts a Lead into an Account, Contact, or (optionally) an Opportunity.
`create()`	Adds one or more new individual objects to your organization's data.
`delete()`	Deletes one or more individual objects from your organization's data.
`emptyRecycleBin()`	Delete records from the recycle bin immediately.
`getDeleted()`	Retrieves the IDs of individual objects of the specified object that have been deleted since the specified time. For information on IDs, see ID Field Type.
`getUpdated()`	Retrieves the IDs of individual objects of the specified object that have been updated since the specified time. For information on IDs, see ID Field Type.
`invalidateSessions()`	Ends one or more sessions specified by `sessionId`.
`login()`	Logs in to the login server and starts a client session.
`logout()`	Ends the session of the logged-in user.
`merge()`	Merges records of the same object type.
`process()`	Submits an array of approval process instances for approval, or processes an array of approval process instances to be approved, rejected, or removed.
`query()`	Executes a query against the specified object and returns data that matches the specified criteria.
`queryAll()`	Same as `query()`, but includes deleted and archived items.
`queryMore()`	Retrieves the next batch of objects from a query.
`retrieve()`	Retrieves one or more objects based on the specified object IDs.
`search()`	Executes a text search in your organization's data.
`undelete()`	Undelete records identified with `queryAll()`.
`update()`	Updates one or more existing objects in your organization's data.

Call	Description
upsert()	Creates new objects and updates existing objects; matches on a custom field to determine the presence of existing objects.

Samples

The samples in this section are based on the enterprise WSDL file. They assume that you have already imported the WSDL file and created a connection. To learn how to do so, see the Quick Start tutorial.

convertLead()

Converts a Lead into an Account, Contact, or (optionally) an Opportunity.

Syntax

```
LeadConvertResult[] = connection.convertLead(leadConverts LeadConvert[]);
```

Usage

Use `convertLead()` to convert a Lead into an Account and Contact, as well as (optionally) an Opportunity. To convert a Lead, your client application must be logged in with the "Convert Leads" permission and the "Edit" permission on leads, as well as "Create" and "Edit" on the Account, Contact, and Opportunity objects.

This call provides an easy way to convert the information in a qualified lead to a new or updated account, contact, and opportunity. Your organization can set its own guidelines for determining when a lead is qualified, but typically, a lead can be converted as soon as it becomes a real opportunity that you want to forecast.

If data is merged into existing account and contact objects, then only empty fields in the target object are overwritten—existing data (including IDs) are not overwritten. The only exception to this is if your client application sets `overwriteLeadSource` to `true`, in which case the `LeadSource` field in the target Contact object will be overwritten with the contents of the `LeadSource` field in the source Lead object.

When converting leads, consider the following rules and guidelines:

Field Mappings

The system automatically maps standard lead fields to standard account, contact, and opportunity fields. For custom lead fields, your Salesforce administrator can specify how they map to custom account, contact, and opportunity fields.

Record Types

If the organization uses record types, the default record type of the new owner is assigned to records created during lead conversion. For more information about record types, see the Salesforce online help.

Picklist Values

The system assigns the default picklist values for the account, contact, and opportunity when mapping any standard lead picklist fields that are blank. If your organization uses record types, blank values are replaced with the default picklist values of the new record owner.

String Values

Starting with API version 15.0, if you specify a value for a field that contains a string, and the value is too big for the field, the call fails and an error is returned. In previous versions of the API the value was truncated and the call succeeded. If you wish to keep the old behavior with versions 15.0 and later, use the AllowFieldTruncationHeader SOAP header.

Automatic Subscriptions for Chatter Feeds

When you convert a lead into a new account, contact, and opportunity, the lead owner is unsubscribed from the lead account. The lead owner, the owner of the generated records, and users that were subscribed to the lead aren't automatically subscribed to the generated records, unless they have automatic subscriptions enabled in their Chatter feed settings. They must have automatic subscriptions enabled to see changes to the account, contact, and opportunity records in their news feed.

A user can subscribe to a record or to another user. Changes to the record and updates from the users are displayed in the Chatter feed on the user's home page, which is a useful way to stay up-to-date with other users and with changes made to records in Salesforce. Feeds are available in API version 18.0 and later.

Basic Steps for Converting Leads

Converting leads involves the following basic steps:

1. The client application determines the IDs of any lead(s) to be converted.
2. Optionally, the client application determines the IDs of any account(s) to merge the lead into. The client application can use SOSL or SOQL to search for accounts that match the lead name, as in the following example:

```
select id, name from account where name='CompanyNameOfLeadBeingMerged'
```

3. Optionally, the client application determines the IDs of contact(s) to merge the lead into. The client application can use SOSL or SOQL to search for contacts that match the lead contact name, as in the following example:

```
select id, name from contact where firstName='FirstName' and lastName='LastName' and
accountId = '001...'
```

4. Optionally, the client application determines whether opportunities should be created from the leads.
5. The client application queries the LeadStatus table to obtain all of the possible converted status options (

```
SELECT Id, MasterLabel FROM LeadStatus WHERE IsConverted=true
```

   ), and then selects a value for the Converted Status.
6. The client application calls `convertLead()`.
7. The client application iterates through the returned result(s) and examine each LeadConvertResult object to determine whether conversion succeeded for each lead.
8. As an optional best practice, the client application creates tasks in which the `WhoId` is the `ContactId` and, if an opportunity is created, the `WhatId` is the `OpportunityId`.
9. Optionally, when converting leads owned by a queue, the owner must be specified. This is because accounts and contacts cannot be owned by a queue. Even if you are specifying an existing account or contact, you must still specify an owner.

Sample Code—Java

This sample shows how to convert leads. It creates two leads and converts them. Next, it iterates through the lead conversion results and writes the IDs of the account, contact, and opportunity created for each lead.

```
public String[] convertLeadRecords() {
   String[] result = new String[4];
   try {

      // Create two leads to convert
      Lead[] leads = new Lead[2];
      Lead lead = new Lead();
      lead.setLastName("Mallard");
      lead.setFirstName("Jay");
      lead.setCompany("Wingo Ducks");
      lead.setPhone("(707) 555-0328");
```

```
      leads[0] = lead;
      lead = new Lead();
      lead.setLastName("Platypus");
      lead.setFirstName("Ogden");
      lead.setCompany("Denio Water Co.");
      lead.setPhone("(775) 555-1245");
      leads[1] = lead;
      SaveResult[] saveResults = connection.create(leads);

      // Create a LeadConvert array to be used
      // in the convertLead() call
      LeadConvert[] leadsToConvert = new LeadConvert[saveResults.length];

      for (int i = 0; i < saveResults.length; ++i) {
         if (saveResults[i].isSuccess()) {
            System.out
                  .println("Created new Lead: " + saveResults[i].getId());
            leadsToConvert[i] = new LeadConvert();
            leadsToConvert[i].setConvertedStatus("Closed - Converted");
            leadsToConvert[i].setLeadId(saveResults[i].getId());
            result[0] = saveResults[i].getId();
         } else {
            System.out.println("\nError creating new Lead: "
                  + saveResults[i].getErrors()[0].getMessage());
         }
      }
      // Convert the leads and iterate through the results
      LeadConvertResult[] lcResults = connection.convertLead(leadsToConvert);
      for (int j = 0; j < lcResults.length; ++j) {
         if (lcResults[j].isSuccess()) {
            System.out.println("Lead converted successfully!");
            System.out.println("Account ID: " + lcResults[j].getAccountId());
            System.out.println("Contact ID: " + lcResults[j].getContactId());
            System.out.println("Opportunity ID: "
                  + lcResults[j].getOpportunityId());
         } else {
            System.out.println("\nError converting new Lead: "
                  + lcResults[j].getErrors()[0].getMessage());
         }
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
   return result;
}
```

Sample Code—C#

This sample shows how to convert leads. It creates two leads and converts them. Next, it iterates through the lead conversion results and writes the IDs of the account, contact, and opportunity created for each lead.

```
public String[] convertLeadRecords()
{
   String[] result = new String[4];
   try
   {
      // Create two leads to convert
      Lead[] leads = new Lead[2];
      Lead lead = new Lead();
      lead.LastName = "Mallard";
      lead.FirstName = "Jay";
      lead.Company = "Wingo Ducks";
      lead.Phone = "(707) 555-0328";
      leads[0] = lead;
      lead = new Lead();
      lead.LastName = "Platypus";
      lead.FirstName = "Ogden";
```

```
      lead.Company = "Denio Water Co.";
      lead.Phone = "(775) 555-1245";
      leads[1] = lead;
      SaveResult[] saveResults = binding.create(leads);

      // Create a LeadConvert array to be used
      //   in the convertLead() call
      LeadConvert[] leadsToConvert =
            new LeadConvert[saveResults.Length]; ;
      for (int i = 0; i < saveResults.Length; ++i)
      {
         if (saveResults[i].success)
         {
            Console.WriteLine("Created new Lead: " +
                  saveResults[i].id);
            leadsToConvert[i] = new LeadConvert();
            leadsToConvert[i].convertedStatus = "Closed - Converted";
            leadsToConvert[i].leadId = saveResults[i].id;
            result[0] = saveResults[i].id;
         }
         else
         {
            Console.WriteLine("\nError creating new Lead: " +
                  saveResults[i].errors[0].message);
         }
      }
      // Convert the leads and iterate through the results
      LeadConvertResult[] lcResults =
            binding.convertLead(leadsToConvert);
      for (int j = 0; j < lcResults.Length; ++j)
      {
         if (lcResults[j].success)
         {
            Console.WriteLine("Lead converted successfully!");
            Console.WriteLine("Account ID: " +
                     lcResults[j].accountId);
            Console.WriteLine("Contact ID: " +
                     lcResults[j].contactId);
            Console.WriteLine("Opportunity ID: " +
                     lcResults[j].opportunityId);
         }
         else
         {
            Console.WriteLine("\nError converting new Lead: " +
                  lcResults[j].errors[0].message);
         }
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                        e.Message + "\n" + e.StackTrace);
   }
   return result;
}
```

LeadConvert Arguments

This call accepts an array of LeadConvert objects (100 maximum). A LeadConvert object contains the following properties.

Name	Type	Description
accountId	ID	ID of the Account into which the lead will be merged. Required only when updating an existing account, including person accounts. If no accountID is specified, then the API creates a new account. To create a new account, the client application must be logged in with sufficient access rights. To merge a lead into an existing account, the client

Name	Type	Description
		application must be logged in with read/write access to the specified account. The account name and other existing data are not overwritten. For information on IDs, see ID Field Type.
`contactId`	ID	ID of the Contact into which the lead will be merged (this contact must be associated with the specified `accountId`, and an `accountId` must be specified). Required only when updating an existing contact. **Important:** If you are converting a lead into a person account, do not specify the `contactId` or an error will result. Specify only the `accountId` of the person account. If no `contactID` is specified, then the API creates a new contact that is implicitly associated with the Account. To create a new contact, the client application must be logged in with sufficient access rights. To merge a lead into an existing contact, the client application must be logged in with read/write access to the specified contact. The contact name and other existing data are not overwritten (unless `overwriteLeadSource` is set to `true`, in which case only the `LeadSource` field is overwritten).

`convertedStatus`	string	Valid LeadStatus value for a converted lead. Required. To obtain the list of possible values, the client application queries the LeadStatus object. For example:
		
		`SELECT Id, MasterLabel`
		`FROM LeadStatus WHERE IsConverted=true`

`doNotCreateOpportunity`	boolean	Specifies whether to create an Opportunity during lead conversion (`false`, the default) or not (`true`). Set this flag to `true` only if you do not want to create an opportunity from the lead. An opportunity is created by default.
`leadId`	ID	ID of the Lead to convert. Required. For information on IDs, see ID Field Type.
`opportunityName`	string	Name of the opportunity to create. If no name is specified, then this value defaults to the company name of the lead. The maximum length of this field is 80 characters. If `doNotCreateOpportunity` argument is `true`, then no Opportunity is created and this field must be left blank; otherwise, an error is returned.
`overwriteLeadSource`	boolean	Specifies whether to overwrite the `LeadSource` field on the target Contact object with the contents of the `LeadSource` field in the source Lead object (`true`), or not (`false`, the default). To set this field to `true`, the client application must specify a `contactId` for the target contact.
`ownerId`	ID	Specifies the ID of the person to own any newly created account, contact, and opportunity. If the client application does not specify this value, then the owner of the new object will be the owner of the lead. Not applicable when merging with existing objects—if an `ownerId` is specified, the API does not overwrite the `ownerId` field in an existing account or contact. For information on IDs, see ID Field Type.

Name	Type	Description
sendNotificationEmail	boolean	Specifies whether to send a notification email to the owner specified in the ownerId (true) or not (false, the default).

Response

LeadConvertResult[]

Fault

UnexpectedErrorFault

See Also:

API Call Basics

https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

LeadConvertResult

This call returns an array of LeadConvertResult objects. Each element in the LeadConvertResult array corresponds to the LeadConvert[] array passed as the leadConverts parameter in the convertLead() call. For example, the object returned in the first index in the LeadConvertResult array matches the object specified in the first index of the LeadConvert[] array. A LeadConvertResult object has the following properties:

Name	Type	Description
accountId	ID	ID of the new Account (if a new account was specified) or the ID of the account specified when convertLead() was invoked.
contactId	ID	ID of the new Contact (if a new contact was specified) or the ID of the contact specified when convertLead() was invoked. For information on IDs, see ID Field Type.
leadId	ID	ID of the converted Lead. For information on IDs, see ID Field Type.
opportunityId	ID	ID of the new Opportunity, if one was created when convertLead() was invoked. For information on IDs, see ID Field Type.
success	boolean	Indicates whether the convertLead() call succeeded (true) or not (false) for this object.
errors	Error[]	If an error occurred during the create() call, an array of one or more Error objects providing the error code and description.

create()

Adds one or more new records to your organization's data.

Syntax

```
SaveResult[] = connection.create(sObject[] sObjects);
```

Usage

Use `create()` to add one or more records, such as an Account or Contact record, to your organization's information. The `create()` call is analogous to the INSERT statement in SQL.

When creating objects, consider the following rules and guidelines.

Permissions

Your client application must be logged in with sufficient access rights to create records within the specified object. For more information, see Factors that Affect Data Access.

Special Handling

Certain objects—and certain fields within those objects—require special handling or permissions. For example, you might also need permissions to access the object's parent object. Before you attempt to `create()` a record for a particular object, be sure to read its description in the Standard Objects.

Createable Fields

Only objects where `createable` is `true` can be created via the `create()` call. To determine whether a given object can be created, your client application can invoke the `describeSObjects()` call on the object and inspect its `createable` property.

Automatically Maintained Fields

The API generates unique values for ID fields automatically. For `create()`, you cannot explicitly specify an ID value in the sObject. The SaveResult[] object contains the ID of each record that was successfully created. For information on IDs, see ID Field Type.

The API populates certain fields automatically, such as `CreatedDate`, `CreatedById`, `LastModifiedDate`, `LastModifiedById`, and `SystemModstamp`. You cannot explicitly specify these values.

Required Fields

For required fields that do not have a preconfigured default value, you must supply a value. For more information, see Required Fields.

Default Values

For some objects, some fields have a default value, such as `OwnerID`. If you do not specify a value for such fields, the API populates the fields with the default value. For example, if you do not override `OwnerID`, then the API populates this field with the user ID associated with the user as whom your client application is logged in.

- For required fields that do not have a preconfigured default value, you must supply a value.
- For all other fields in the object, if you do not explicitly specify a value, then its value is `null (VT_EMPTY)`.

Referential Integrity

Your client application must conform to the rules of referential integrity. For example, if you are creating a record for an object that is the child of a parent object, you must supply the foreign key information that links the child to the parent. For example, when creating a CaseComment, you must supply the valid case ID for the parent Case, and that parent Case must exist in the database.

Valid Data Values

You must supply values that are valid for the field's data type, such as integers (not alphabetic characters) for integer fields. In your client application, follow the data formatting rules specified for your programming language and development tool (your development tool will handle the appropriate mapping of data types in SOAP messages).

String Values

When storing values in string fields, the API trims any leading and trailing whitespace. For example, if the value of a name field is entered as `" ABC Company "`, then the value is stored in the database as `"ABC Company"`.

Starting with API version 15.0, if you specify a value for a field that contains a string, and the value is too big for the field, the call fails and an error is returned. In previous versions of the API the value was truncated and the call succeeded. If you wish to keep the old behavior with versions 15.0 and later, use the AllowFieldTruncationHeader SOAP header.

Assignment Rules

When creating new Account (accounts fire Territory Management assignment rules), Case, or Lead records, your client application can set options in the AssignmentRuleHeader to have the case or lead automatically assigned to one or more users based on assignment rules configured in the Salesforce user interface.

Maximum Number of Records Created

Your client application can add up to 200 records in a single `create()` call. If a create request exceeds 200 objects, then the entire operation fails.

Rollback on Error

The AllOrNoneHeader header allows you to roll back all changes unless all records are processed successfully. This header is available in API version 20.0 and later. The default behavior is to allow partial success of a call: records without errors are committed, while records with errors are marked as failed in the call results.

Automatic Subscriptions for Chatter Feeds

To subscribe to records they create, users must enable the `Automatically follow records that I create` option in their personal settings. If users have automatic subscriptions enabled, they automatically follow the records they create and see changes to those records in their Chatter feed on the Home tab.

A user can subscribe to a record or to another user. Changes to the record and updates from the users are displayed in the Chatter feed on the user's home page, which is a useful way to stay up-to-date with other users and with changes made to records in Salesforce. Feeds are available in API version 18.0 and later. The EntitySubscription object represents a subscription of a user following a record or another user.

Disabling Feed Notifications

If you're processing a large number of records and don't want to track the changes in various feeds related to the records, use DisableFeedTrackingHeader. This is especially useful for bulk changes.

Creating Records for Different Object Types

You can create records for multiple object types, including custom objects, in one call with API version 20.0 and later. For example, you could create a contact and an account in one call. You can create records for up to 10 object types in one call.

Records are saved in the same order that they are entered in the `sObjects` input array. If you are entering new records that have a parent-child relationship, the parent record must precede the child record in the `sObjects` array. For example, if you are creating a contact that references an account that is also being created in the same call, the account must have a smaller index in the `sObjects` array than the contact does. The contact references the account by using an `External ID` field.

You can't add a record that references another record of the same object type in the same call. For example, the Contact object has a `Reports To` field that is a reference to another contact. You can't create two contacts in one call if one contact uses the `Reports To` field to reference a second contact in the `sObjects` array. You can create a contact that references another contact that has been previously created.

Records for different object types are broken into multiple chunks by Salesforce. A chunk is a subset of the `sObjects` input array and each chunk contains records of one object type. Data is committed on a chunk-by-chunk basis. Any Apex triggers related to the records in a chunk are invoked once per chunk. Consider an `sObjects` input array containing the following set of records:

```
account1, account2, contact1, contact2, contact3, case1, account3, account4, contact4
```

Salesforce splits the records into five chunks:

1. `account1, account2`
2. `contact1, contact2, contact3`
3. `case1`
4. `account3, account4`
5. `contact4`

Each call can process up to 10 chunks. If the `sObjects` array contains more than 10 chunks, you must process the records in more than one call.

Warning: You can't create records for multiple object types in one call if one of those types is related to a feature in the Setup area in Salesforce. The only exceptions are the following objects:

- Custom settings objects, which are similar to custom objects. For more information, see "Custom Settings Overview" in the Salesforce online help.
- GroupMember
- Group
- User if the `UserRoleId` field is not being set.

`create()` and Foreign Keys

You can use external ID fields as a foreign key, which allows you to create a record and relate it to another existing record in a single step instead of querying the parent record ID first. To do this, set the foreign key field to an instance of the parent sObject that only has the external ID field specified. This external ID should match the external ID value on the parent record.

The following Java and C# examples show you how to create an opportunity and relate it to an existing account using a custom external ID field named `MyExtId__c`. Each example creates an opportunity, sets the required fields, and then sets the opportunity external ID field to the account object that has only the external ID field specified. The code then creates the opportunity. Once the opportunity is created, the account will be its parent.

Java Example

```
public void createForeignKeySample() {
   try {
      Opportunity newOpportunity = new Opportunity();
      newOpportunity.setName("OpportunityWithFK");
      newOpportunity.setStageName("Prospecting");
      Calendar dt = connection.getServerTimestamp().getTimestamp();
      dt.add(Calendar.DAY_OF_MONTH, 7);
      newOpportunity.setCloseDate(dt);

      Account parentAccountRef = new Account();
      parentAccountRef.setMyExtId__c("SAP1111111");
      newOpportunity.setAccount(parentAccountRef);

      SaveResult[] results = connection
            .create(new SObject[] { newOpportunity });
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

C# Example

```
public void createForeignKeySample()
{
   try
   {
      Opportunity newOpportunity = new Opportunity();
      newOpportunity.Name = "OpportunityWithFK";
      newOpportunity.StageName = "Prospecting";
      DateTime dt = (DateTime)binding.getServerTimestamp().timestamp;
      newOpportunity.CloseDate = dt.AddDays(7);
      newOpportunity.CloseDateSpecified = true;

      // Create the parent reference.
      // Used only for foreign key reference
      //  and doesn't contain any other fields
      Account accountReference = new Account();
      accountReference.MyExtId__c = "SAP1111111";
      newOpportunity.Account = accountReference;

      // Create the account and the opportunity
      SaveResult[] results = binding.create(new sObject[] {
                              newOpportunity });
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                              e.Message + "\n" + e.StackTrace);
   }
}
```

Creating Parent and Child Records in a Single Call Using Foreign Keys

You can use external ID fields as foreign keys to create parent and child records of different sObject types in a single call instead of creating the parent record first, querying its ID, and then creating the child record. To do this:

- Create the child sObject and populate its required fields, and optionally other fields.
- Create the parent reference sObject used only for setting the parent foreign key reference on the child sObject. This sObject has only the external ID field defined and no other fields set.
- Set the foreign key field of the child sObject to the parent reference sObject you just created.
- Create another parent sObject to be passed to the `create()` call. This sObject must have the required fields (and optionally other fields) set in addition to the external ID field.
- Call `create()` by passing it an array of sObjects to create. The parent sObject must precede the child sObject in the array, that is, the array index of the parent must be lower than the child's index.

The parent and child records are records related through a predefined relationship, such as a master-detail or lookup relationship. You can create related records that are up to 10 levels deep. Also, the related records created in a single call must have different sObject types. For more information, see Creating Records for Different Object Types.

The following Java and C# examples show you how to create an opportunity with a parent account in the same `create()` call. Each example creates an Opportunity sObject and populates some of its fields, then creates two Account objects. The first account is only for the foreign key relationship, and the second is for the account creation and has the account fields set. Both accounts have the external ID field, `MyExtID__c`, set. Next, the sample calls `create()` by passing it an array of sObjects. The first element in the array is the parent sObject and the second is the opportunity sObject. The `create()` call creates the opportunity with its parent account in a single call. Finally, the sample checks the results of the call and writes the IDs of the created records to the console, or the first error if record creation fails.

Java Example

```
public void createForeignKeySample() {
   try {
      Opportunity newOpportunity = new Opportunity();
```

```
        newOpportunity.setName("OpportunityWithAccountInsert");
        newOpportunity.setStageName("Prospecting");
        Calendar dt = connection.getServerTimestamp().getTimestamp();
        dt.add(Calendar.DAY_OF_MONTH, 7);
        newOpportunity.setCloseDate(dt);

        // Create the parent reference.
        // Used only for foreign key reference
        // and doesn't contain any other fields.
        Account accountReference = new Account();
        accountReference.setMyExtID__c("SAP111111");
        newOpportunity.setAccount(accountReference);

        // Create the Account object to insert.
        // Same as above but has Name field.
        // Used for the create call.
        Account parentAccount = new Account();
        parentAccount.setName("Hallie");
        parentAccount.setMyExtID__c("SAP111111");

        // Create the account and the opportunity.
        SaveResult[] results = connection.create(new SObject[] {
              parentAccount, newOpportunity });

        // Check results.
        for (int i = 0; i < results.length; i++) {
           if (results[i].isSuccess()) {
              System.out.println("Successfully created ID: "
                    + results[i].getId());
           } else {
              System.out.println("Error: could not create sobject "
                    + "for array element " + i + ".");
              System.out.println("   The error reported was: "
                    + results[i].getErrors()[0].getMessage() + "\n");
           }
        }
     } catch (ConnectionException ce) {
        ce.printStackTrace();
     }
}
```

C# Example

```
public void createForeignKeySample()
{
   try
   {
      Opportunity newOpportunity = new Opportunity();
      newOpportunity.Name = "OpportunityWithAccountInsert";
      newOpportunity.StageName = "Prospecting";
      DateTime dt = (DateTime)binding.getServerTimestamp().timestamp;
      newOpportunity.CloseDate = dt.AddDays(7);
      newOpportunity.CloseDateSpecified = true;

      // Create the parent reference.
      // Used only for foreign key reference
      //  and doesn't contain any other fields.
      Account accountReference = new Account();
      accountReference.MyExtID__c = "SAP111111";
      newOpportunity.Account = accountReference;

      // Create the Account object to insert.
      // Same as above but has Name field.
      // Used for the create call.
      Account parentAccount = new Account();
      parentAccount.Name = "Hallie";
      parentAccount.MyExtID__c = "SAP111111";
```

```
      // Create the account and the opportunity.
      SaveResult[] results = binding.create(new sObject[] {
      parentAccount, newOpportunity });

      // Check results.
      for (int i = 0; i < results.Length; i++)
      {
         if (results[i].success)
         {
            Console.WriteLine("Successfully created ID: "
                  + results[i].id);
         }
         else
         {
            Console.WriteLine("Error: could not create sobject "
                  + "for array element " + i + ".");
            Console.WriteLine("   The error reported was: "
                  + results[i].errors[0].message + "\n");
         }
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                              e.Message + "\n" + e.StackTrace);
   }
}
```

Basic Steps for Creating Records

Creating records involves the following basic steps:

1. Create an sObject for one or more objects. For each record, populate its fields with the data that you want to add.
2. Construct an sObject[] array and populate that array with the objects that you want to create.
3. Call `create()`, passing in the sObject[] array.
4. Process the results in the `SaveResult`[] object to verify whether the records have been successfully created.

Sample Code—Java

This sample shows how to create records. It creates two Account objects and sets their fields. The Name of the second account isn't set so that an error occurs on creation, since Name is a required field. After making the `create()` call by passing the array containing the two accounts, the sample iterates over the results and writes the ID of the new account or an error message if the account creation fails. Finally, the sample returns an array of the new account IDs, which in this case contains only one ID.

```
public String[] createRecords() {
   // Create two accounts
   String[] result = new String[2];
   Account account1 = new Account();
   Account account2 = new Account();

   // Set some fields on the account object
   account1.setName("The Brick Hut");
   account1.setBillingStreet("403 McAdoo St");
   account1.setBillingCity("Truth or Consequences");
   account1.setBillingState("NM");
   account1.setBillingPostalCode("87901");
   account1.setBillingCountry("US");
   // Required Name field is not being set on account2,
   // so this record should fail during create.
   // account2.setName("Camp One Creations");
   account2.setBillingStreet("25800 Arnold Dr");
   account2.setBillingCity("Sonoma");
   account2.setBillingState("CA");
   account2.setBillingPostalCode("95476");
```

```
    account2.setBillingCountry("US");
    Account[] accounts = { account1, account2 };

    try {
        // Call create() to add the accounts
        SaveResult[] saveResults = connection.create(accounts);
        // Iterate through the results.
        // There should be one successful creation
        // and one failed creation.
        for (int i = 0; i < saveResults.length; i++) {
            if (saveResults[i].isSuccess()) {
                System.out.println("Successfully created Account ID: "
                      + saveResults[i].getId());
                result[i] = saveResults[i].getId();
            } else {
                System.out.println("Error: could not create Account "
                      + "for array element " + i + ".");
                System.out.println("   The error reported was: "
                      + saveResults[i].getErrors()[0].getMessage() + "\n");
                result[i] = saveResults[i].getId();
            }
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
    return result;
}
```

Sample Code—C#

This sample shows how to create records. It creates two Account objects and sets their fields. The Name of the second account isn't set so that an error occurs on creation, since Name is a required field. After making the `create()` call by passing the array containing the two accounts, the sample iterates over the results and writes the ID of the new account or an error message if the account creation fails. Finally, the sample returns an array of the new account IDs, which in this case contains only one ID.

```
public String[] createRecords()
{
    // Create two accounts
    String[] result = new String[2];
    Account account1 = new Account();
    Account account2 = new Account();

    // Set some fields on the account object
    account1.Name = "The Brick Hut";
    account1.BillingStreet = "403 McAdoo St";
    account1.BillingCity = "Truth or Consequences";
    account1.BillingState = "NM";
    account1.BillingPostalCode = "87901";
    account1.BillingCountry = "US";
    // Required Name field is not being set on account2,
    // so this record should fail during create.
    // account2.Name = "Camp One Creations";
    account2.BillingStreet = "25800 Arnold Dr";
    account2.BillingCity = "Sonoma";
    account2.BillingState = "CA";
    account2.BillingPostalCode = "95476";
    account2.BillingCountry = "US";
    Account[] accounts = { account1, account2 };

    try
    {
        // Call create() to add the accounts
        SaveResult[] saveResults = binding.create(accounts);
        // Iterate through the results.
        // There should be one successful creation
        //   and one failed creation.
```

```
        for (int i = 0; i < saveResults.Length; i++)
        {
           if (saveResults[i].success)
           {
              Console.WriteLine("Successfully created Account ID: " +
                    saveResults[i].id);
              result[i] = saveResults[i].id;
           }
           else
           {
              Console.WriteLine("Error: could not create Account " +
                    "for array element " + i + "."
              );
              Console.WriteLine("   The error reported was: " +
              saveResults[i].errors[0].message + "\n");
              result[i] = saveResults[i].id;
           }
        }
     }
     catch (SoapException e)
     {
        Console.WriteLine("An unexpected error has occurred: " +
                                 e.Message + "\n" + e.StackTrace);
     }

     return result;
}
```

Arguments

Name	Type	Description
sObjects	sObject[]	Array of one or more sObject objects to create(). Limit: 200 sObject values.

Response

SaveResult[]

Faults

InvalidSObjectFault

UnexpectedErrorFault

See Also:

upsert()
API Call Basics
https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

SaveResult

The create() call returns an array of SaveResult objects. Each element in the SaveResult array corresponds to the sObject[] array passed as the sObjects parameter in the create() call. For example, the object returned in the first index in the SaveResult array matches the object specified in the first index of the sObject[] array. A SaveResult object has the following properties:

Name	Type	Description
`id`	ID	ID of the sObject that you attempted to `create()`. If this field contains a value, then the object was created successfully. If this field is empty, then the object was not created and the API returned error information instead.
`success`	boolean	Indicates whether the `create()` call succeeded (`true`) or not (`false`) for this object.
`errors`	Error[]	If an error occurred during the `create()` call, an array of one or more Error objects providing the error code and description.

`delete()`

Deletes one or more records from your organization's data.

Syntax

```
DeleteResult[] = connection.delete(ID[] ids);
```

Usage

Use `delete()` to delete one or more existing records, such as individual accounts or contacts, in your organization's data. The `delete()` call is analogous to the `DELETE` statement in SQL.

Rules and Guidelines

When deleting objects, consider the following rules and guidelines:

- Your client application must be logged in with sufficient access rights to delete individual objects within the specified object. For more information, see Factors that Affect Data Access.
- In addition, you might also need permission to access this object's parent object. For special access requirements, see the object's description in Standard Objects.
- To ensure referential integrity, the `delete()` call supports cascading deletions. If you delete a parent object, you delete its children automatically, as long as each child object can be deleted. For example, if you delete a Case, the API automatically deletes any CaseComment, CaseHistory, and CaseSolution objects associated with that case. However, if a CaseComment is not deletable or is currently being used, then the `delete()` call on the parent Case will fail.
- Certain objects cannot be deleted via the API. To delete an object via the `delete()` call, its object must be configured as deletable (`deletable` is `true`). To determine whether a given object can be deleted, your client application can invoke the `describeSObjects()` call on the object and inspect its `deletable` property.
- You can't delete records for multiple object types in one call if one of those types is related to a feature in the Setup area in Salesforce. The only exceptions are the following objects:
  - ◊ Custom settings objects, which are similar to custom objects. For more information, see “Custom Settings Overview” in the Salesforce online help.
  - ◊ GroupMember
  - ◊ Group
  - ◊ User

Rollback on Error

The AllOrNoneHeader header allows you to roll back all changes unless all records are processed successfully. This header is available in API version 20.0 and later. The default behavior is to allow partial success of a call: records without errors are committed, while records with errors are marked as failed in the call results.

Basic Steps for Deleting Records

Deleting records involves the following basic steps:

1. Determine the ID of each record that you want to delete. For example, you might call `query()` to retrieve a set of records that you want to delete based on specific criteria.
2. Construct an ID[] array and populate it with the IDs of each record that you want to delete. You can specify the IDs of different types of objects in the same call. For example, you could specify the ID for an individual Account and an individual Contact in the same array. For information on IDs, see ID Field Type.
3. Call `delete()`, passing in the ID[] array.
4. Process the results in the `DeleteResult`[] to verify whether the records have been successfully deleted.

Sample Code—Java

This sample shows how to delete records based on record IDs. The method in this sample accepts an array of IDs, which it passes to the `delete()` call and makes the call. It then parses the results and writes the IDs of the deleted records to the console or the first returned error if the deletion failed.

```
public void deleteRecords(String[] ids) {
   try {
      DeleteResult[] deleteResults = connection.delete(ids);
      for (int i = 0; i < deleteResults.length; i++) {
         DeleteResult deleteResult = deleteResults[i];
         if (deleteResult.isSuccess()) {
            System.out
                  .println("Deleted Record ID: " + deleteResult.getId());
         } else {
            // Handle the errors.
            // We just print the first error out for sample purposes.
            Error[] errors = deleteResult.getErrors();
            if (errors.length > 0) {
               System.out.println("Error: could not delete " + "Record ID "
                     + deleteResult.getId() + ".");
               System.out.println("   The error reported was: ("
                     + errors[0].getStatusCode() + ") "
                     + errors[0].getMessage() + "\n");
            }
         }
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample shows how to delete records based on record IDs. The method in this sample accepts an array of IDs, which it passes to the `delete()` call and makes the call. It then parses the results and writes the IDs of the deleted records to the console or the first returned error if the deletion failed.

```
public void deleteRecords(String[] ids)
{
   try
   {
      DeleteResult[] deleteResults = binding.delete(ids);
      for (int i = 0; i < deleteResults.Length; i++)
```

```
        {
           DeleteResult deleteResult = deleteResults[i];
           if (deleteResult.success)
           {
              Console.WriteLine("Deleted Record ID: " + deleteResult.id);
           }
           else
           {
              // Handle the errors.
              // We just print the first error out for sample purposes.
              Error[] errors = deleteResult.errors;
              if (errors.Length > 0)
              {
                 Console.WriteLine("Error: could not delete " + "Record ID "
                       + deleteResult.id + ".");
                 Console.WriteLine("   The error reported was: ("
                       + errors[0].statusCode + ") "
                       + errors[0].message + "\n");
              }
           }
        }
     }
     catch (SoapException e)
     {
        Console.WriteLine("An unexpected error has occurred: " +
                                   e.Message + "\n" + e.StackTrace);
     }
  }
```

Arguments

Name	Type	Description
ids	ID[]	Array of one or more IDs associated with the objects to delete. In version 7.0 and later, you can pass a maximum of 200 object IDs to the `delete()` call. In version 6.0 and earlier, the limit is 2,000.

Response

DeleteResult[]

Faults

`InvalidSObjectFault`

`UnexpectedErrorFault`

See Also:

API Call Basics
https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

DeleteResult

The `delete()` call returns an array of `DeleteResult` objects. Each element in the `DeleteResult` array corresponds to the ID[] array passed as the `ids` parameter in the `delete()` call. For example, the object returned in the first index in the `DeleteResult` array matches the object specified in the first index of the ID[] array.

A `DeleteResult` object has the following properties:

Name	Type	Description
id	ID	ID of an sObject that you attempted to delete. For information on IDs, see ID Field Type.
success	boolean	Indicates whether the `delete()` call succeeded (`true`) or not (`false`) for this object.
errors	Error[]	If an error occurred during the `delete()` call, an array of one or more Error objects providing the error information.

emptyRecycleBin()

Delete records from the recycle bin immediately.

Syntax

```
EmptyRecycleBinResult[] = connection.emptyRecycleBin(ID[] ids);
```

Usage

The Recycle Bin lets you view and restore recently deleted records for 15 days before they are permanently deleted. Your organization can have up to 5,000 records per license in the Recycle Bin at any one time. For example, if your organization has five user licenses, 25,000 records can be stored in the Recycle Bin. If your organization reaches its Recycle Bin limit, Salesforce automatically removes the oldest records, as long as they have been in the recycle bin for at least two hours.

If you know you will be adding a great number of records to the Recycle Bin and you know you won't need to `undelete()` them, you may wish to remove them before the Salesforce process deletes records. For example, you can use this call if you are loading a large number of records for testing, or if you are doing a large number of `create()` calls followed by `delete()` calls.

Rules and Guidelines

When emptying recycle bins, consider the following rules and guidelines:

- The logged in user can delete any record that he or she can query in their Recycle Bin, or the recycle bins of any subordinates. If the logged in user has Modify All Data permission, he or she can query and delete records from any Recycle Bin in the organization.
- Available in version 10.0 and later.
- Maximum number of records is 200.
- Do not include the IDs of any records that will be cascade deleted, or an error will occur.
- Once records are deleted using this call, they cannot be `undelete()`d.
- After records are deleted from the Recycle Bin using this call, they can be queried using `queryAll()` for some time. Typically this time is 24 hours, but may be shorter or longer.

Sample Code—Java

This sample shows how to empty the Recycle Bin. It accepts an array containing the IDs of the records to remove from the Recycle Bin. It calls `emptyRecycleBin()` and passes it the array of IDs. Next, it iterates over the results and writes the IDs of the removed records or the first error of the failed records to the console.

```
public void emptyRecycleBin(String[] ids) {
   try {
      EmptyRecycleBinResult[] emptyRecycleBinResults = connection
```

```
            .emptyRecycleBin(ids);
      for (int i = 0; i < emptyRecycleBinResults.length; i++) {
         EmptyRecycleBinResult emptyRecycleBinResult = emptyRecycleBinResults[i];
         if (emptyRecycleBinResult.isSuccess()) {
            System.out.println("Recycled ID: "
                  + emptyRecycleBinResult.getId());
         } else {
            Error[] errors = emptyRecycleBinResult.getErrors();
            if (errors.length > 0) {
               System.out
                     .println("Error code: " + errors[0].getStatusCode());
               System.out
                     .println("Error message: " + errors[0].getMessage());
            }
         }
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample shows how to empty the Recycle Bin. It accepts an array containing the IDs of the records to remove from the Recycle Bin. It calls `emptyRecycleBin()` and passes it the array of IDs. Next, it iterates over the results and writes the IDs of the removed records or the first error of the failed records to the console.

```
public void emptyRecycleBin(String[] ids)
{
   try
   {
      EmptyRecycleBinResult[] emptyRecycleBinResults =
         binding.emptyRecycleBin(ids);
      for (int i = 0; i < emptyRecycleBinResults.Length; i++)
      {
         EmptyRecycleBinResult emptyRecycleBinResult = emptyRecycleBinResults[i];
         if (emptyRecycleBinResult.success)
         {
            Console.WriteLine("Recycled ID: "
                  + emptyRecycleBinResult.id);
         }
         else
         {
            Error[] errors = emptyRecycleBinResult.errors;
            if (errors.Length > 0)
            {
               Console.WriteLine("Error code: " + errors[0].statusCode);
               Console.WriteLine("Error message: " + errors[0].message);
            }
         }
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                                   e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

Name	Type	Description
`ids`	ID[]	Array of one or more IDs associated with the records to delete from the Recycle Bin. Maximum number of records is 200.

Response

EmptyRecycleBinResult

Faults

`InvalidSObjectFault`

`UnexpectedErrorFault`

See Also:

delete()
undelete()

EmptyRecycleBinResult

The `emptyRecycleBin()` call returns an array of `EmptyRecycleBinResult` objects. Each element in the array corresponds to an element in the ID[] array passed as the parameter in the `emptyRecycleBin()` call. For example, the object returned in the first index in the `EmptyRecycleBinResult` array matches the object specified in the first index of the ID[] array.

A `EmptyRecycleBinResult` object has the following properties:

Name	Type	Description
`id`	ID	ID of an sObject that you attempted to delete from the Recycle Bin. For information about IDs, see ID Field Type.
`isSuccess`	boolean	Indicates whether the call succeeded (`true`) or not (`false`) for this record.
`errors`	Error[]	If an error occurred during the call, an array of one or more Error objects providing the error information.

`getDeleted()`

Retrieves the list of individual records that have been deleted within the given timespan for the specified object.

Syntax

```
GetDeletedResult = connection.getDeleted(string sObjectType, dateTime startDate, dateTime
 EndDate);
```

Usage

Use `getDeleted()` for data replication applications to retrieve a list of records that have been deleted from your organization's data within the specified timespan. The `getDeleted()` call retrieves a GetDeletedResult object that contains an array of `DeletedRecord` objects containing the ID of each deleted record and the date/time (Coordinated Universal Time (UTC) time zone) on which it was deleted. Be sure to read Data Replication before using `getDeleted()` in your client applications. (For information on IDs, see ID Field Type.)

As of release 8.0, the `getDeleted()` call respects the user's sharing model.

Rules and Guidelines

When replicating deleted records, consider the following rules and guidelines:

- The specified `startDate` must chronologically precede the specified `endDate` value. The specified `startDate` cannot be the same value as, or later than, the specified `endDate` value. Otherwise, the API returns an `INVALID_REPLICATION_DATE` error.
- Records are returned only if the user has access to them.
- Results are returned for no more than 15 days previous to the day the call is executed (or earlier if an administrator has purged the Recycle Bin). If the purge has been performed before your `getDeleted()` call is executed, an `INVALID_REPLICATION_DATE` error is returned.
- If latestDateCovered is less than `endDate`, the call will fail, returning an `INVALID_REPLICATION_DATE` error with the value of latestDateCovered.
- Deleted records are written to a delete log, which `getDeleted()` accesses. A background process that runs every two hours purges records that have been in an organization's delete log for more than two hours if the number of records is above a certain limit. Starting with the oldest records, the process purges delete log entries until the delete log is back below the limit. This is done to protect Salesforce from performance issues related to massive delete logs. The limit is calculated using this formula:

  ```
  5000 * number of licenses in the organization
  ```

  For example, an organization with 1,000 licenses could have up to 5,000,000 (five million) records in the delete log before any purging took place. If purging has been performed before your `getDeleted()` call is executed, an `INVALID_REPLICATION_DATE` error is returned. If you get this exception, you should do a full pull of the table.

- If you delete a large numbers of records, your data replication should run more frequently than every two hours to ensure all records are returned by `getDeleted()`.
- Client applications typically poll for changed data periodically. For important polling considerations, see Polling for Changes.
- Records for certain objects cannot be replicated via the API. To replicate a record via the `getDeleted()` call, its object must be configured as replicateable (`rReplicateable` is `true`). To determine whether a given object can be replicated, your client application can invoke the `describeSObjects()` call on the object and inspect its `replicateable` property.
- Development tools differ in the way that they handle time data. Some development tools report the local time, while others report only the Coordinated Universal Time (UTC) time. To determine how your development tool handles time values, refer to its documentation.
- If you call `getDeleted()` for a history object, the call returns the records deleted during the given date range for all history objects, not only the history object you specified. For example, if you call `getDeleted()` for AccountHistory, you'll get records deleted during the given date range for AccountHistory, ContactHistory, and so on.

Basic Steps for Replicating Deleted Records

You can replicate deleted records using the following basic steps for each object:

1. Optionally, determine whether the structure of the object has changed since the last replication request, as described in Checking for Structural Changes in the Object.
2. Call `getDeleted()`, passing in the object and the relevant time span for deleted records.
3. In the DeleteResult object, iterate through the returned array of `DeletedRecord` objects containing the ID of each deleted record and the date on which it was deleted (Coordinated Universal Time (UTC) time zone).
4. Take the appropriate action on the local data to remove the deleted records or flag as deleted.
5. Optionally, save the request time span for future reference. You should save the value of latestDateCovered.

A client application likely performs other tasks associated with data replication operations. For example, if an opportunity is closed, a client application might run a new revenue report. Similarly, if a task is completed, the process might log this in another system.

Sample Code—Java

This sample calls `getDeleted()` to get all accounts that were deleted in the last 60 minutes. It then writes the ID and the deleted date of each returned account to the console.

```
public void getDeletedRecords() {
   try {
      GregorianCalendar endTime = (GregorianCalendar)
         connection.getServerTimestamp().getTimestamp();
      GregorianCalendar startTime = (GregorianCalendar) endTime.clone();
      // Subtract 60 minutes from the server time so that we have
      // a valid time frame.
      startTime.add(GregorianCalendar.MINUTE, -60);
      System.out.println("Checking deletes at or after: "
            + startTime.getTime().toString());

      // Get records deleted during the specified time frame.
      GetDeletedResult gdResult = connection.getDeleted("Account",
            startTime, endTime);

      // Check the number of records contained in the results,
      // to check if something was deleted in the 60 minute span.
      DeletedRecord[] deletedRecords = gdResult.getDeletedRecords();
      if (deletedRecords != null && deletedRecords.length > 0) {
         for (int i = 0; i < deletedRecords.length; i++) {
            DeletedRecord dr = deletedRecords[i];
            System.out.println(dr.getId() + " was deleted on "
                  + dr.getDeletedDate().getTime().toString());
         }
      } else {
         System.out.println("No deletions of Account records in "
               + "the last 60 minutes.");
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample calls `getDeleted()` to get all accounts that were deleted in the last 60 minutes. It then writes the ID and the deleted date of each returned account to the console.

```
public void getDeletedRecords()
{
   try
   {
      DateTime endTime = binding.getServerTimestamp().timestamp;
      // Subtract 60 minutes from the server time so that we have
      // a valid time frame.
      DateTime startTime = endTime.AddMinutes(-60);
      Console.WriteLine("Checking deletes at or after: "
            + startTime.ToLocalTime().ToString());

      // Get records deleted during the specified time frame.
      GetDeletedResult gdResult = binding.getDeleted("Account",
            startTime, endTime);

      // Check the number of records contained in the results,
      // to check if something was deleted in the 60 minute span.
      DeletedRecord[] deletedRecords = gdResult.deletedRecords;
      if (deletedRecords != null && deletedRecords.Length > 0)
      {
         for (int i = 0; i < deletedRecords.Length; i++)
         {
            DeletedRecord dr = deletedRecords[i];
```

```
            Console.WriteLine(dr.id + " was deleted on "
                  + dr.deletedDate.ToLocalTime().ToString());
         }
      }
      else
      {
         Console.WriteLine("No deletions of Account records in "
               + "the last 60 minutes.");
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                                 e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

Name	Type	Description
sObjectTypeEntityType	string	Object type. The specified value must be a valid object for your organization. See sObject.
startDate	dateTime	Starting date/time (Coordinated Universal Time (UTC)—not local— timezone) of the timespan for which to retrieve the data. The API ignores the seconds portion of the specified dateTime value (for example, 12:30:15 is interpreted as 12:30:00 UTC).
endDate	dateTime	Ending date/time (Coordinated Universal Time (UTC)—not local— timezone) of the timespan for which to retrieve the data. The API ignores the seconds portion of the specified dateTime value (for example, 12:35:15 is interpreted as 12:35:00 UTC).

Limits

There are record limits on the result GetDeletedResult:

- If your `getDeleted()` call returns more than 600,000 records and the user is a system administrator, an exception EXCEEDED_ID_LIMIT is returned.
- If your `getDeleted()` call returns more than 20,000 records and the user is not a system administrator, an exception OPERATION_TOO_LARGE is returned. Note that this error is returned when more than 20,000 records across the organization have been deleted, not just the records viewable by the user.

You can correct the error by choosing start and end dates that are closer together.

Response

GetDeletedResult

Faults

`InvalidSObjectFault`

`UnexpectedErrorFault`

See Also:

Data Replication
API Call Basics
https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

GetDeletedResult

The `getDeleted()` call returns a `GetDeletedResult` object that contains an array of `DeletedRecord` records and two properties:

Name	Type	Description
`earliestDateAvailable`	dateTime	For the object type of the `getDeleted()` call, the timestamp (Coordinated Universal Time (UTC)—not local— timezone) of the last physically deleted object. If this value is less than `endDate`, the call will fail, and you should resynch your data before performing another replication.
`deletedRecords[]`	deletedRecords	Array of the deleted records which satisfy the start and end dates specified in the `getDeleted()` call.
`latestDateCovered`	dateTime	The timestamp (Coordinated Universal Time (UTC)—not local— time zone) of the last date covered in the `getDeleted()` call. If there is a value, it is less than or equal to `endDate`. A value here indicates that, for safety, you should use this value for the `startDate` of your next call to capture the changes that started after this date but did not complete before `endDate` and were, therefore, not returned in the previous call.

deletedRecords

The GetDeletedResult contains and array of `deletedRecords`, which contain the following properties:

Name	Type	Description
`deletedDate`	dateTime	Date and time (Coordinated Universal Time (UTC)—not local—timezone) when this record was deleted.
`id`	ID	ID of an sObject that has been deleted.

`getUpdated()`

Retrieves the list of individual records that have been updated (added or changed) within the given timespan for the specified object.

Syntax

```
GetUpdatedResult[] = connection.getUpdated(string sObjectType, dateTime startDate, dateTime
 EndDate);
```

Usage

Use `getUpdated()` for data replication applications to retrieve a set of IDs for objects of the specified object that have been created or updated within the specified timespan. The `getUpdated()` call retrieves an array of `GetUpdatedResult` objects containing the ID of each created or updated object and the date/time (Coordinated Universal Time (UTC) time zone) on which it was created or updated, respectively. Be sure to read Data Replication before using `getUpdated()` in your client application.

Note: The `getUpdated()` call retrieves the IDs only for objects to which the logged-in user has access.

Rules and Guidelines

When replicating created and updated objects, consider the following rules and guidelines:

- The specified `startDate` must chronologically precede the specified `endDate` value. The specified `startDate` cannot be the same value as, or later than, the specified `endDate` value. Otherwise, the API returns an `INVALID_REPLICATION_DATE` error.
- Results are returned for no more than 30 days previous to the day the call is executed.
- Client applications typically poll for changed data periodically. For important polling considerations, see Polling for Changes.
- Your client application can replicate any objects to which it has sufficient permissions. For example, to replicate all data for your organization, your client application must be logged in with “View All Data” access rights to the specified object. Similarly, the objects must be within your sharing rules. For more information, see Factors that Affect Data Access.
- Certain objects cannot be replicated via the API. To replicate an object via the `getUpdated()` call, its object must be configured as replicateable (`replicateable` is `true`). To determine whether a given object can be replicated, your client application can invoke the `describeSObjects()` call on the object and inspect its `replicateable` property.
- Certain objects cannot be deleted, such as Group, User, Contract, or Product2 objects. However, if instances of these objects are no longer visible in the Salesforce user interface, they may have been rendered inactive so that only users with administrative access can see them. To determine whether a missing object instance has been made inactive, your client application can call `getUpdated()` and check the object’s active flag.
- Development tools differ in the way that they handle time data. Some development tools report the local time, while others report only the Coordinated Universal Time (UTC) time. To determine how your development tool handles time values, refer to its documentation.

Basic Steps for Replicating Updated Objects

Replicating objects involves the following basic steps for each object that you want to replicate:

1. Optionally, the client application determines whether the structure of the object has changed since the last replication request, as described in Checking for Structural Changes in the Object.
2. Call `getUpdated()`, passing in the object and timespan for which to retrieve data.
3. Iterate through the returned array of IDs. For each ID element in the array, call `retrieve()` to obtain the latest information you want from the associated object. Your client application must then take the appropriate action on the local data, such as inserting new rows or updating existing ones with the latest information.
4. Optionally, the client application saves the request timestamp for future reference.

A client application likely performs other tasks associated with data replication operations. For example, if an opportunity were to become closed, a client application might run a new revenue report. Similarly, if a task were completed, the process might log this somehow in another system.

Sample Code—Java

This sample gets the accounts that were updated in the last 60 minutes and writes their IDs to the console.

```
public void getUpdatedRecords() {
   try {
      GregorianCalendar endTime = (GregorianCalendar) connection
            .getServerTimestamp().getTimestamp();
      GregorianCalendar startTime = (GregorianCalendar) endTime.clone();
      // Subtract 60 minutes from the server time so that we have
      // a valid time frame.
      startTime.add(GregorianCalendar.MINUTE, -60);
      System.out.println("Checking updates as of: "
            + startTime.getTime().toString());

      // Get the updated accounts within the specified time frame
      GetUpdatedResult ur = connection.getUpdated("Account", startTime,
            endTime);
      System.out.println("GetUpdateResult: " + ur.getIds().length);

      // Write the results
      if (ur.getIds() != null && ur.getIds().length > 0) {
         for (int i = 0; i < ur.getIds().length; i++) {
            System.out.println(ur.getIds()[i] + " was updated between "
                  + startTime.getTime().toString() + " and "
                  + endTime.getTime().toString());
         }
      } else {
         System.out.println("No updates to accounts in "
               + "the last 60 minutes.");
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample gets the accounts that were updated in the last 60 minutes and writes their IDs to the console.

```
public void getUpdatedRecords()
{
   try
   {
      DateTime endTime = binding.getServerTimestamp().timestamp;
      // Subtract 60 minutes from the server time so that we have
      // a valid time frame.
      DateTime startTime = endTime.AddMinutes(-60);
      Console.WriteLine("Checking updates as of: "
            + startTime.ToLocalTime().ToString());

      // Get the updated accounts within the specified time frame
      GetUpdatedResult ur = binding.getUpdated("Account", startTime,
            endTime);
      Console.WriteLine("GetUpdateResult: " + ur.ids.Length);

      // Write the results
      if (ur.ids != null && ur.ids.Length > 0)
      {
         for (int i = 0; i < ur.ids.Length; i++)
         {
            Console.WriteLine(ur.ids[i] + " was updated between "
                  + startTime.ToLocalTime().ToString() + " and "
                  + endTime.ToLocalTime().ToString());
         }
      }
      else
```

```
        {
           Console.WriteLine("No updates to accounts in "
                 + "the last 60 minutes.");
        }
     }
     catch (SoapException e)
     {
        Console.WriteLine("An unexpected error has occurred: " +
                                   e.Message + "\n" + e.StackTrace);
     }
}
```

Arguments

Name	Type	Description
sObjectTypeEntityType	string	Object type. The specified value must be a valid object for your organization. For a list of standard objects, see Standard Objects.
startDate	dateTime	Starting date/time (Coordinated Universal Time (UTC) time zone—not local—timezone) of the timespan for which to retrieve the data. The API ignores the seconds portion of the specified dateTime value (for example, 12:30:15 is interpreted as 12:30:00 UTC).
endDate	dateTime	Ending date/time (Coordinated Universal Time (UTC) time zone—not local—timezone) of the timespan for which to retrieve the data. The API ignores the seconds portion of the specified dateTime value (for example, 12:35:15 is interpreted as 12:35:00 UTC).

Important: There is a limit of 200,000 IDs in the result GetUpdatedResult[]. If your `getUpdated()` call returns more than 200,000 IDs, an exception EXCEEDED_ID_LIMIT is returned. You can correct the error by choosing start and end dates that are closer together.

Response

GetUpdatedResult[]

Faults

`InvalidSObjectFault`

`UnexpectedErrorFault`

See Also:

Data Replication
API Call Basics
https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

GetUpdatedResult

The `getUpdated()` call returns a `GetUpdatedResult` object that contains information about each record that was inserted or updated within the given timespan. An `GetUpdatedResult` object has the following properties:

Name	Type	Description
id[]	ID	Array of IDs of each object that has been updated.
latestDateCovered	dateTime	The timestamp (Coordinated Universal Time (UTC)—not local— time zone) of the last date covered in the getUpdated() call. If there is a value, it is less than or equal to endDate. A value here indicates that, for safety, you should use this value for the startDate of your next call to capture the changes that started after this date but did not complete before the endDate and were, therefore, not returned in the previous call. **Note:** If Salesforce executes a long-running transaction on your instance, the value in this field is the start time of that long-running transaction until it completes. This is because a long-running transaction might affect your user data (for example, batch processing).

invalidateSessions()

Ends one or more sessions specified by a sessionId.

Syntax

```
InvalidateSessionsResult = connection.invalidateSessions(string[] sessionIds);
```

Usage

Use this call to end one or more sessions.

You can also use logout() to end just one session, the session of the logged-in user.

Sample Code—Java

This sample invalidates a set of sessions. The method in this sample takes an array of session IDs passed in as String values. The method then calls invalidateSessions() with this array and then checks the results for any errors.

```
public void invalidateSessionsSample(String[] sessionIds) {
   try {
      InvalidateSessionsResult[] results;
      results = connection.invalidateSessions(sessionIds);
      for (InvalidateSessionsResult result : results) {
         // Check results for errors
         if (!result.isSuccess()) {
            if (result.getErrors().length > 0) {
               System.out.println("Status code: "
                     + result.getErrors()[0].getStatusCode());
               System.out.println("Error message: "
                     + result.getErrors()[0].getMessage());
            }
         } else {
            System.out.println("Success.");
         }
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample invalidates a set of sessions. The method in this sample takes an array of session IDs passed in as String values. The method then calls `invalidateSessions()` with this array and then checks the results for any errors.

```
public void invalidateSessionsSample(string[] sessionIds)
{
   try
   {
      InvalidateSessionsResult[] results;
      results = binding.invalidateSessions(sessionIds);
      foreach (InvalidateSessionsResult result in results)
      {
         // Check results for errors
         if (!result.success)
         {
            if (result.errors.Length > 0)
            {
               Console.WriteLine("Status code: " +
                  result.errors[0].statusCode);
               Console.WriteLine("Error message: " +
                  result.errors[0].message);
            }
         }
         else
         {
            Console.WriteLine("Success.");
         }
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                                 e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

Name	Type	Description
`sessionIds`	string[]	One or more `sessionId` strings. Limit 200. You can obtain your `sessionId` from the SessionHeader.

Response

InvalidateSessionsResult[]

Faults

`UnexpectedErrorFault`

InvalidateSessionsResult

The `invalidateSessions()` call returns an array of `LogoutResult` objects. Each object has the following properties:

Name	Type	Description
`success`	boolean	Indicates whether the session was successfully terminated (`true`) or not (`false`).

Name	Type	Description
errors	Error[]	If an error occurred during the call, an array of one or more Error objects. Each object contains an error code and description.

login()

Logs in to the login server and starts a client session.

Syntax

```
LoginResult = connection.login(string username, string password);
```

Usage

Use the `login()` call to log in to the login server and start a client session. A client application must log in and obtain a `sessionId` and server URL before making any other API calls.

When a client application invokes the `login()` call, it passes in a username and password as user credentials. Upon invocation, the API authenticates the credentials and returns the `sessionId` for the session, the user ID associated with the logged-in username, and a URL that points to the Force.com API to use in all subsequent API calls.

Salesforce checks the IP address from which the client application is logging in, and blocks logins from unknown IP addresses. For a blocked login via the API, Salesforce returns a login fault. Then, the user must add their security token to the end of their password in order to log in. A security token is an automatically-generated key from Salesforce. For example, if a user's password is `mypassword`, and their security token is `XXXXXXXXXX`, then the user must enter `mypasswordXXXXXXXXXX` to log in. Users can obtain their security token by changing their password or resetting their security token via the Salesforce user interface. When a user changes their password or resets their security token, Salesforce sends a new security token to the email address on the user's Salesforce record. The security token is valid until a user resets their security token, changes their password, or has their password reset. When the security token is invalid, the user must repeat the login process to log in. To avoid this, the administrator can make sure the client's IP address is added to the organization's list of trusted IP addresses. For more information, see Security Token.

After logging in, a client application needs to perform these tasks:

- Set the session ID in the SOAP header so that the API can validate subsequent requests for this session.
- Specify the server URL as the target for subsequent service requests. You must change to the server URL, the login server only supports login calls.

Development tools differ in the way you specify session headers and server URLs. For more information, see the documentation for your particular development tool.

Note: Multiple client applications can log in using the same `username` argument. However, this increases your risk of getting errors due to query limits. A user can have up to 10 query cursors open at a time. If 10 `QueryLocator` cursors are open when a client application, logged in as the same user, attempts to open a new one, then the oldest of the 10 cursors is released. If the client application attempts to open the released query cursor, an error results.

There is a limit of 3600 calls to `login()` per user per hour. Exceeding this limit will result in a “Login Rate Exceeded” error.

Enterprise and Partner Endpoints

In version 11.1 of the API and earlier, client applications built with the partner WSDL can send requests to the enterprise endpoint and enterprise WSDL applications can send requests to the partner endpoint. Beginning with version 12.0, this is not supported.

Logging In When Using a Proxy

If you log into Salesforce, via a proxy, set the proxy host and port on the instance of the ConnectorConfig class that you use to log in. Optionally, you may need to set the username and password, if you must authenticate on the proxy.

```
ConnectorConfig config = new ConnectorConfig();
config.setUsername(userId);
config.setPassword(passwd);
config.setAuthEndpoint(authEndPoint);
config.setProxy(proxyHost, proxyPort);
// Set the username and password if your proxy must be authenticated
config.setProxyUsername(proxyUsername);
config.setProxyPassword(proxyPassword);
try {
   EnterpriseConnection connection = new EnterpriseConnection(config);
   // etc.
} catch (ConnectionException ce) {
  ce.printStackTrace();
}
```

Session Expiration

Client applications do not need to explicitly log out to end a session. Sessions expire automatically after a predetermined length of inactivity, which can be configured in Salesforce from Setup by clicking **Security Controls**. The default is 120 minutes (two hours). If you make an API call, the inactivity timer is reset to zero.

Authenticating Active Self-Service Users

Note: Starting with Spring '12, the Self-Service portal isn't available for new organizations. Existing organizations continue to have access to the Self-Service portal.

To authenticate active Self-Service users, use the LoginScopeHeader to specify the Organization ID against which Self-Service users are authenticated. A Self-Service user must exist and be active before being authenticated (see SelfServiceUser).

Authenticating Customer Service Portal and Customer Community Users in Salesforce Communities

To authenticate an active Customer Community or Customer Portal user, use the LoginScopeHeader to specify the Organization ID of the org with communities. Customer Community and Customer Portal users must exist, be active, and belong to communities in the organization before being authenticated.

Logging Out

Salesforce recommends that you always call `logout()` to end a session when it is no longer needed. This ends any child sessions as well as the session being logged out. Logging out instead of waiting for the configured session expiration provides the most protection.

Sample Code—Java

This sample logs a user in using the specified username, password, and authentication endpoint URL. The sample writes user and session information to the console after a successful login. Before running this sample, replace the values for user name, password, and authentication endpoint with valid values.

To learn how to generate and import the Web service WSDL that you need to make API calls, see Step 2: Generate or Obtain the Web Service WSDL in the Quick Start.

```
public boolean loginSample() {
   boolean success = false;
   String username = "username";
   String password = "password";
```

```
    String authEndPoint = "https://login.salesforce.com/services/Soap/c/24.0/";

    try {
       ConnectorConfig config = new ConnectorConfig();
       config.setUsername(username);
       config.setPassword(password);

       System.out.println("AuthEndPoint: " + authEndPoint);
       config.setAuthEndpoint(authEndPoint);

       connection = new EnterpriseConnection(config);

       // Print user and session info
       GetUserInfoResult userInfo = connection.getUserInfo();
       System.out.println("UserID: " + userInfo.getUserId());
       System.out.println("User Full Name: " + userInfo.getUserFullName());
       System.out.println("User Email: " + userInfo.getUserEmail());
       System.out.println();
       System.out.println("SessionID: " + config.getSessionId());
       System.out.println("Auth End Point: " + config.getAuthEndpoint());
       System.out
             .println("Service End Point: " + config.getServiceEndpoint());
       System.out.println();

       success = true;
    } catch (ConnectionException ce) {
       ce.printStackTrace();
    }

    return success;
 }
```

Sample Code—C#

This sample logs a user in using the specified username and password. The result of the login call contains the service endpoint URL, which is the virtual server instance that's servicing your organization, and a unique session ID. The sample sets these returned values on the binding. It sets the binding URL to the returned service endpoint. It also sets the session ID on the session header that is used on all API calls. Next, the sample writes user and session information to the console after a successful login. Before running this sample, replace the values for user name and password with valid values.

To learn how to generate and import the Web service WSDL that you need to make API calls, see Step 2: Generate or Obtain the Web Service WSDL in the Quick Start.

```
public bool loginSample()
{
   Boolean success = false;
   string username = "username";
   string password = "password";

   // Create a service object
   binding = new SforceService();

   LoginResult lr;
   try
   {
      Console.WriteLine("\nLogging in...\n");
      lr = binding.login(username, password);

      /**
         * The login results contain the endpoint of the virtual server instance
         * that is servicing your organization. Set the URL of the binding
         * to this endpoint.
         */
      // Save old authentication end point URL
      String authEndPoint = binding.Url;
      // Set returned service endpoint URL
```

```
        binding.Url = lr.serverUrl;

        /** Get the session ID from the login result and set it for the
           * session header that will be used for all subsequent calls.
           */
        binding.SessionHeaderValue = new SessionHeader();
        binding.SessionHeaderValue.sessionId = lr.sessionId;

        // Print user and session info
        GetUserInfoResult userInfo = lr.userInfo;
        Console.WriteLine("UserID: " + userInfo.userId);
        Console.WriteLine("User Full Name: " +
              userInfo.userFullName);
        Console.WriteLine("User Email: " +
              userInfo.userEmail);
        Console.WriteLine();
        Console.WriteLine("SessionID: " +
              lr.sessionId);
        Console.WriteLine("Auth End Point: " +
              authEndPoint);
        Console.WriteLine("Service End Point: " +
              lr.serverUrl);
        Console.WriteLine();

        // Return true to indicate that we are logged in, pointed
        // at the right URL and have our security token in place.
        success = true;
    }
    catch (SoapException e)
    {
        Console.WriteLine("An unexpected error has occurred: " +
                                   e.Message + "\n" + e.StackTrace);
    }
    return success;
}
```

Arguments

Name	Type	Description
`username`	string	Login `username`.
`password`	string	Login password associated with the specified `username`.

The login request size is limited to 10 KB or less.

Response

LoginResult

Faults

`LoginFault`

`UnexpectedErrorFault`

See Also:

API Call Basics

https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

LoginResult

The `login()` call returns a `LoginResult` object, which has the following properties:

Name	Type	Description
`metadataServerUrl`	string	URL of the endpoint that will process subsequent metadata API calls. Your client application needs to set the endpoint.
`passwordExpired`	boolean	Indicates whether the password used during the login attempt is expired (`true`) or not (`false`). If the password has expired, then the API returns a valid `sessionId`, but the only allowable operation is the `setPassword()` call.
`serverUrl`	string	URL of the endpoint that will process subsequent API calls. Your client application needs to set the endpoint.
`sessionId`	string	Unique ID associated with this session. Your client application needs to set this value in the session header.
`userId`	ID	ID of the user associated with the specified username and password.
`userInfo`	getUserInfoResult	User information fields. For a list of these fields, see getUserInfoResult.

`logout()`

Ends the session of the logged-in user.

Syntax

```
connection.logout();
```

Usage

This call ends the session for the logged-in user issuing the call. No arguments are needed.

To end one or more sessions started by someone other than the logged-in user, see `invalidateSessions()`.

Sample Code—Java

This sample calls `logout()` to log the current user out and writes a message to the console.

```
public void logoutSample() {
   try {
      connection.logout();
      System.out.println("Logged out.");
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample calls `logout()` to log the current user out and writes a message to the console.

```
public void logoutSample()
{
   try
   {
      binding.logout();
      Console.WriteLine("Logged out.");
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                               e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

This call uses no arguments. It ends the session for the logged-in user issuing the call, so no arguments are needed. The logged-in user is identified by the `sessionId` specified in the SessionHeader for this call.

Response

Void is returned. Because failure of the call means that the session has already been logged out, no results are needed. Any unexpected error, such as system unavailability, throws an error that should be handled by your client application.

Faults

`UnexpectedErrorFault`

`merge()`

Merge up to three records into one.

Syntax

```
MergeResult[]= connection.merge(MergeRequest[] mergeRequests);
```

Usage

Use this call to merge records of the same object type into one of the records, deleting the others, and re-parenting any related records. Each merge operation is within one transaction. A batch merge has multiple transactions, one for each element in the batch.

The only supported object types are Lead, Contact and Account.

The `masterRecord` field indicates which of the records is the master record that the others are merged into. You can use `queryAll()` to view records that have been deleted during a merge.

This call requires that you decide prior to the merge call if there are any field values from the non-master record(s) that should supersede the values in the master record. If so, the field names and their new values should be set in the `masterRecord` of the MergeRequest, similar to a call to update.

The following limits apply to any merge request:

- Up to 200 merge requests can be made in a single SOAP call.

- Up to three records can be merged in a single request, including the master record. This is the same limit as the Salesforce user interface. If you have more than three records to merge, use the same master record in each request to avoid errors.
- External ID fields cannot be used with `merge()`.

To find all records that have been merged since a given point in time, you can use `queryAll()` with a SELECT statement similar to the following:

```
SELECT Id FROM Contact WHERE isDeleted=true and masterRecordId != null
   AND SystemModstamp > 2006-01-01T23:01:01+01:00
```

It is a recommended best practice to narrow your result set to the most relevant records by filtering on `SystemModstamp`.

Note:

Starting with API version 15.0, if you specify a value for a field that contains a string, and the value is too big for the field, the call fails and an error is returned. In previous versions of the API the value was truncated and the call succeeded. If you wish to keep the old behavior with versions 15.0 and later, use the AllowFieldTruncationHeader SOAP header.

Sample Code—Java

This sample merges one account with a master account. It creates two accounts and attaches a note to the account to merge with the master. After the merge, it writes the ID of the account that was merged and the number of child records updated, which in this case is one because the note of the merged account is moved to the master account.

```
public Boolean mergeRecords() {
   Boolean success = false;
   // Array to hold the results
   String[] accountIds = new String[2];
   try {
      // Create two accounts to merge
      Account[] accounts = new Account[2];
      Account masterAccount = new Account();
      masterAccount.setName("MasterAccount");
      masterAccount.setDescription("The Account record to merge with.");
      accounts[0] = masterAccount;
      Account accountToMerge = new Account();
      accountToMerge.setName("AccountToMerge");
      accountToMerge
            .setDescription("The Account record that will be merged.");
      accounts[1] = accountToMerge;
      SaveResult[] saveResults = connection.create(accounts);

      if (saveResults.length > 0) {
         for (int i = 0; i < saveResults.length; i++) {
            if (saveResults[i].isSuccess()) {
               accountIds[i] = saveResults[i].getId();
               System.out.println("Created Account ID: "
                     + accountIds[i]);
            } else {
               // If any account is not created,
               // print the error returned and exit
               System.out
                     .println("An error occurred while creating account."
                           + " Error message: "
                           + saveResults[i].getErrors()[0].getMessage());
               return success;
            }
         }
      }

      // Set the Ids of the accounts
      masterAccount.setId(accountIds[0]);
```

```
        accountToMerge.setId(accountIds[1]);

        // Attach a note to the account to be merged with the master,
        // which will get re-parented after the merge
        Note note = new Note();
        System.out.println("Attaching note to record " +
              accountIds[1]);
        note.setParentId(accountIds[1]);
        note.setTitle("Merged Notes");
        note.setBody("This note will be moved to the "
              + "MasterAccount during merge");
        SaveResult[] sRes = connection.create(new SObject[] { note });
        if (sRes[0].isSuccess()) {
           System.out.println("Created Note record.");
        } else {
           Error[] errors = sRes[0].getErrors();
           System.out.println("Could not create Note record: "
                 + errors[0].getMessage());
        }

        // Perform the merge
        MergeRequest mReq = new MergeRequest();
        masterAccount.setDescription("Was merged");
        mReq.setMasterRecord(masterAccount);
        mReq.setRecordToMergeIds(new String[] { saveResults[1].getId() });
        MergeResult mRes = connection.merge(new MergeRequest[] { mReq })[0];

        if (mRes.isSuccess())
        {
           System.out.println("Merge successful.");
           // Write the IDs of merged records
           for(String mergedId : mRes.getMergedRecordIds()) {
              System.out.println("Merged Record ID: " + mergedId);

           }
           // Write the updated child records. (In this case the note.)
           System.out.println(
                 "Child records updated: " + mRes.getUpdatedRelatedIds().length);
           success = true;
        } else {
           System.out.println("Failed to merge records. Error message: " +
                 mRes.getErrors()[0].getMessage());
        }
     } catch (ConnectionException ce) {
        ce.printStackTrace();
     }
     return success;
}
```

Sample Code—C#

This sample merges one account with a master account. It creates two accounts and attaches a note to the account to merge with the master. After the merge, it writes the ID of the account that was merged and the number of child records updated, which in this case is one because the note of the merged account is moved to the master account.

```
public Boolean mergeRecords()
{
   Boolean success = false;
   // Array to hold the results
   String[] accountIds = new String[2];
   try
   {
      // Create two accounts to merge
      Account[] accounts = new Account[2];
      Account masterAccount = new Account();
      masterAccount.Name = "MasterAccount";
      masterAccount.Description = "The Account record to merge with.";
```

```
accounts[0] = masterAccount;
Account accountToMerge = new Account();
accountToMerge.Name = "AccountToMerge";
accountToMerge
      .Description = "The Account record that will be merged.";
accounts[1] = accountToMerge;
SaveResult[] saveResults = binding.create(accounts);

if (saveResults.Length > 0)
{
   for (int i = 0; i < saveResults.Length; i++)
   {
      if (saveResults[i].success)
      {
         accountIds[i] = saveResults[i].id;
         Console.WriteLine("Created Account ID: "
               + accountIds[i]);
      }
      else
      {
         // If any account is not created,
         // print the error returned and exit
         Console.WriteLine("An error occurred while creating account."
                     + " Error message: "
                     + saveResults[i].errors[0].message);
         return success;
      }
   }
}

// Set the Ids of the accounts
masterAccount.Id = accountIds[0];
accountToMerge.Id = accountIds[1];

// Attach a note to the account to be merged with the master,
// which will get re-parented after the merge
Note note = new Note();
Console.WriteLine("Attaching note to record " +
      accountIds[1]);
note.ParentId = accountIds[1];
note.Title = "Merged Notes";
note.Body = "This note will be moved to the "
      + "MasterAccount during merge";
SaveResult[] sRes = binding.create(new sObject[] { note });
if (sRes[0].success)
{
   Console.WriteLine("Created Note record.");
}
else
{
   Error[] errors = sRes[0].errors;
   Console.WriteLine("Could not create Note record: "
         + errors[0].message);
}

// Perform the merge
MergeRequest mReq = new MergeRequest();
masterAccount.Description = "Was merged";
mReq.masterRecord = masterAccount;
mReq.recordToMergeIds = new String[] { saveResults[1].id };
MergeResult mRes = binding.merge(new MergeRequest[] { mReq })[0];

if (mRes.success)
{
   Console.WriteLine("Merge successful.");
   // Write the IDs of merged records
   foreach (String mergedId in mRes.mergedRecordIds)
   {
      Console.WriteLine("Merged Record ID: " + mergedId);
```

```
            }
            // Write the updated child records. (In this case the note.)
            Console.WriteLine(
                  "Child records updated: " + mRes.updatedRelatedIds.Length);
            success = true;
         }
         else
         {
            Console.WriteLine("Failed to merge records. Error message: " +
                  mRes.errors[0].message);
         }
      }
      catch (SoapException e)
      {
         Console.WriteLine("An unexpected error has occurred: " +
                                    e.Message + "\n" + e.StackTrace);
      }
      return success;
}
```

Arguments

This call accepts an array of MergeRequest objects. A MergeRequest object contains the following properties.

Name	Type	Description
masterRecord	sObject	Required. Must provide the ID of the object that other records will be merged into. Optionally, provide the fields to be updated and their values.
recordToMergeIds	ID[]	Required. Minimum of one, maximum of two. The other record or records to be merged into the master record.

Response

MergeResult[]

Faults

InvalidSObjectFault

UnexpectedErrorFault

InvalidIdFault

See Also:

API Call Basics

MergeResult

The merge() call returns a MergeResult object, which has the following properties:

Name	Type	Description
errors	Error[]	If an error occurred during the merge() call, an array of one or more Error objects providing the error code and description.
id	ID	ID of the master record, the record into which the other records were merged.

Name	Type	Description
mergedRecordIds	ID[]	ID of the records that were merged into the master record. If successful, the values will match mergeRequest.recordToMergeIds.
success	boolean	Indicates whether the merge was successful (true) or not (false).
updatedRelatedIds	ID[]	ID of all related records that were moved (re-parented) as a result of the merge, and that are viewable by the user sending the merge call.

performQuickActions()

Executes quick actions of type create or update.

Note: In the application, QuickActions are referred to as actions or publisher actions.

Syntax

```
PerformQuickActionResult[] = connection.performQuickActions(PerformQuickActionRequest
PerformQuickActionRequest[]);
```

Usage

Use the performQuickActions() call to perform a specific quick action. Returns an array of PerformQuickActionResult objects.

Note: If you're accessing the API using a custom community URL and you use the performQuickActions() call to create a group, the group will only be available within that community.

Sample—Java

This sample uses a quick action to create a new contact.

```
public void example() throws Exception {

    PerformQuickActionRequest req = new PerformQuickActionRequest();

    Contact con = new Contact();
    con.setLastName("Smith");

    req.setQuickActionName("Account.QuickCreateContact");
    req.setParentId("001D000000JSaHa");
/* For version 29.0 and greater, use setContextId */
    req.setRecords(new SObject[] { con });  //you can only save one record here
    PerformQuickActionResult[] pResult =
        conn.performQuickActions(new PerformQuickActionRequest[] { req } );
    for(PerformQuickActionResult pr : pResult) {
        assert pr.getSuccess();
        assert pr.getCreated();
        assert pr.getErrors().length == 0;
        System.out.println("Id of the record created: " + pr.getIds()[0]);
        System.out.println("Id of the feeditem for action performed: " +
            pr.getFeedItemIds()[0]);
    }
}
```

Arguments

Name	Type	Description
`quickActions`	PerformQuickActionRequest	The action request to perform.

PerformQuickActionRequest

Name	Type	Description
`parentOrContextId`	ID	• In API version 28.0 `parentId` is the ID of the sObject on which to create a record for the request. • In API version 29.0 and greater, `contextId` is the ID of the context on which to create a record for the request.
`quickActionName`	string	The parent or context sObject and action name—for example, Opportunity.QuickCreateOpp.
`records`	SObject[]	The record to be created. Only one record can be saved at a time.

Response

PerformQuickActionResult

PerformQuickActionResult

The `performQuickActions()` call returns an array of PerformQuickActionResult objects.

Note: In the application, QuickActions are referred to as actions or publisher actions.

Name	Type	Description
`created`	boolean	If `true`, the record was created successfully and if `false`, no record was created.
`errors`	Error[]	If an error occurred during the call, an array of one or more Error objects providing the error information.
`feedItemIds`	ID[]	Returns an array of unique identifiers of a feed item in the form of a string with IDs; in partner portals, a type with an ID is returned.
`ids`	ID[]	An array of IDs.
`success`	boolean	If `true`, the action executed successfully and if `false`, the action failed.

`process()`

Submits an array of approval process instances for approval, or processes an array of approval process instances to be approved, rejected, or removed. For more information, see “Getting Started with Approval Processes” in the Salesforce online help.

Syntax

```
ProcessResult = connection.process( processType processRequest[])
```

`processType` can be either ProcessSubmitRequest or ProcessWorkitemRequest

Usage

Use the `process()` call to perform either of the following two tasks:

- Submit an array of objects to the approval process. Objects cannot already be in an approval process when submitted. Use the ProcessSubmitRequest signature.
- Process an object that has been submitted to the approval process by performing an approval action (Approve or Reject). Use the ProcessWorkitemRequest signature.

Requests are processed and a ProcessResult is returned with the same process instances as sent in the request.

The failure of a particular record will not cause failure of the entire request.

Note:

Because you can fire Apex triggers with this call, you may be updating fields that contain strings.

Starting with API version 15.0, if you specify a value for a field that contains a string, and the value is too big for the field, the call fails and an error is returned. In previous versions of the API the value was truncated and the call succeeded. If you wish to keep the old behavior with versions 15.0 and later, use the AllowFieldTruncationHeader SOAP header.

Sample Code—Java

This sample accepts the ID of the sObject to process the approval for and an array containing the IDs of the next approvers. It creates a process approval request and submits it for approval. Finally, it parses the results of the `process()` call.

```
public void processRecords(String id, String[] approverIds) {
   ProcessSubmitRequest request = new ProcessSubmitRequest();
   request.setComments("A comment about this approval.");
   request.setObjectId(id);
   request.setNextApproverIds(approverIds);
   try {
      ProcessResult[] processResults = connection
            .process(new ProcessSubmitRequest[] { request });
      for (ProcessResult processResult : processResults) {
         if (processResult.isSuccess()) {
            System.out.println("Approval submitted for: " + id + ":");
            for (int i = 0; i < approverIds.length; i++) {
               System.out
                     .println("\tBy: " + approverIds[i] + " successful.");
            }
            System.out.println("Process Instance Status: "
                  + processResult.getInstanceStatus());
         } else {
            System.out.println("Approval submitted for: " + id
                  + ", approverIds: " + approverIds.toString() + " FAILED.");
            System.out.println("Error: "
                  + processResult.getErrors().toString());
         }
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample accepts the ID of the sObject to process the approval for and an array containing the IDs of the next approvers. It creates a process approval request and submits it for approval. Finally, it parses the results of the `process()` call.

```
public void processRecords(String id, String[] approverIds)
{
   ProcessSubmitRequest request = new ProcessSubmitRequest();
   request.comments = "A comment about this approval.";
   request.objectId = id;
   request.nextApproverIds = approverIds;
   try
   {
      ProcessResult[] processResults = binding.process(
                     new ProcessSubmitRequest[] { request });
      foreach (ProcessResult processResult in processResults)
      {
         if (processResult.success)
         {
            Console.WriteLine("Approval submitted for: " + id + ":");
            for (int i = 0; i < approverIds.Length; i++)
            {
               Console.WriteLine("\tBy: " + approverIds[i] + " successful.");
            }
            Console.WriteLine("Process Instance Status: "
                  + processResult.instanceStatus);
         }
         else
         {
            Console.WriteLine("Approval submitted for: " + id
                  + ", approverIds: " + approverIds.ToString() + " FAILED.");
            Console.WriteLine("Error: "
                  + processResult.errors.ToString());
         }
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                                 e.Message + "\n" + e.StackTrace);
   }
}
```

ProcessSubmitRequest Arguments

Name	Type	Description
`objectId`	ID	The object to submit for approval, for example, an Account, Contact, or custom object.
`nextApproverIds`	ID[]	If the process requires specification of the next approval, the ID of the user to be assigned the next request.
`comments`	string	The comment to add to the history step associated with this request.

ProcessWorkitemRequest Arguments

Name	Type	Description
`action`	string	For processing an item after being submitted for approval, a string representing the kind of action to take: Approve, Reject, or Remove. Only system administrators can specify Remove. If the **Allow submitters to recall approval requests** option is selected for the approval process, the submitter can also specify Remove.

Name	Type	Description
nextApproverIds	ID[]	If the process requires specification of the next approval, the ID of the user to be assigned the next request.
comments	string	The comment to add to the history step associated with this request.
workitemId	ID	The ID of the ProcessInstanceWorkitem that is being approved, rejected, or removed.

Response

ProcessResult[]

Faults

ALREADY_IN_PROCESS

NO_APPLICABLE_PROCESS

See Also:

API Call Basics

ProcessResult

The process() call returns a ProcessResult object, which has the following properties, depending on the type of call (submit for approval or process object already submitted to for approval):

Name	Type	Description
actorIds	ID[]	IDs of the users who are currently assigned to this approval step.
entityId	ID	The object being processed.
errors	Error[]	The set of errors returned if the request failed.
instanceId	ID	The ID of the ProcessInstance associated with the object submitted for processing.
instanceStatus	string	The status of the current process instance (not an individual object but the entire process instance). The valid values are “Approved,” “Rejected,” “Removed,” or “Pending.”
newWorkItemIds	ID[]	Case-insensitive IDs that point to ProcessInstanceWorkitem items (the set of pending approval requests).
success	boolean	true if processing or approval completed successfully.

query()

Executes a query against the specified object and returns data that matches the specified criteria.

Syntax

```
QueryResult = connection.query(string queryString);
```

Usage

Use the `query()` call to retrieve data from an object. When a client application invokes the `query()` call, it passes in a query expression that specifies the object to query, the fields to retrieve, and any conditions that determine whether a given object qualifies. For an extensive discussion about the syntax and rules used for queries, see the *Salesforce SOQL and SOSL Reference Guide*.

Upon invocation, the API executes the query against the specified object, caches the results of the query on the API, and returns a query response object to the client application. The client application can then use methods on the query response object to iterate through rows in the query response and retrieve information.

Your client application must be logged in with sufficient access rights to query individual objects within the specified object and to query the fields in the specified field list. For more information, see Factors that Affect Data Access.

Certain objects cannot be queried via the API. To query an object via the `query()` call, its object must be configured as queryable. To determine whether an object can be queried, your client application can invoke the `describeSObjects()` call on the object and inspect its `queryable` property.

Tip: If you use the enterprise WSDL, you should not use `describe` to populate a select list. For example, if a system administrator adds a field to the SObject after you consume it, the `describe` call will pull down the field but your toolkit won't know how to serialize it, and your integration may fail.

You can use `queryAll()` to query on all Task and Event records, archived or not. You can also filter on the `isArchived` field to find only the archived objects. You cannot use `query()`, it automatically filters out all records where `isArchived` is set to `true`. You can insert, update, or delete archived records.

The query result object contains up to 500 rows of data by default. If the query results exceed 500 rows, then the client application uses the `queryMore()` call and a server-side cursor to retrieve additional rows in 500-row chunks. You can increase the default size up to 2,000 in the QueryOptions header. For more details see Changing the Batch Size in Queries in the *Force.com SOQL and SOSL Reference*.

Queries that take longer than two minutes to process will be timed out. For timed out queries, the API returns an API fault element of `InvalidQueryLocatorFault`. If a timeout occurs, refactor your query to return or scan a smaller amount of data.

When querying for fields of type Base64 (see base64), the query response object returns only one record at a time. You cannot alter this by changing the batch size of the `query()` call.

Note: For multicurrency organizations, special handling is required when querying currency fields containing values in different currencies. For example, if a client application is querying PricebookEntry objects based on values in the `UnitPrice` field, and if the `UnitPrice` amounts are expressed in different currencies, then the query logic must handle this case correctly. For example, if the query is trying to retrieve the product codes of all products with a unit price greater than or equal to $10 USD, the query expression might look something like this:

```
SELECT Product2Id,ProductCode,UnitPrice FROM PricebookEntry
WHERE (UnitPrice >= 10 and CurrencyIsoCode='USD')
OR (UnitPrice >= 5.47 and CurrencyIsoCode='GBP')
OR (UnitPrice >= 8.19 and CurrencyIsoCode='EUR')
```

Sample Code—Java

This sample executes a query that fetches the first names and last names of all contacts. It calls `query()` with the query string to get the first batch of records. It then calls `queryMore()` in a loop to get subsequent batches of records until no records are returned. It writes the first and last names of the contacts queried to the console.

```
public void queryRecords() {
   QueryResult qResult = null;
   try {
      String soqlQuery = "SELECT FirstName, LastName FROM Contact";
      qResult = connection.query(soqlQuery);
      boolean done = false;
      if (qResult.getSize() > 0) {
         System.out.println("Logged-in user can see a total of "
            + qResult.getSize() + " contact records.");
         while (!done) {
            SObject[] records = qResult.getRecords();
            for (int i = 0; i < records.length; ++i) {
               Contact con = (Contact) records[i];
               String fName = con.getFirstName();
               String lName = con.getLastName();
               if (fName == null) {
                  System.out.println("Contact " + (i + 1) + ": " + lName);
               } else {
                  System.out.println("Contact " + (i + 1) + ": " + fName
                        + " " + lName);
               }
            }
            if (qResult.isDone()) {
               done = true;
            } else {
               qResult = connection.queryMore(qResult.getQueryLocator());
            }
         }
      } else {
         System.out.println("No records found.");
      }
      System.out.println("\nQuery succesfully executed.");
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample executes a query that fetches the first names and last names of all contacts. It calls `query()` with the query string to get the first batch of records. It then calls `queryMore()` in a loop to get subsequent batches of records until no records are returned. It writes the first and last names of the contacts queried to the console.

```
public void queryRecords()
{
   QueryResult qResult = null;
   try
   {
      String soqlQuery = "SELECT FirstName, LastName FROM Contact";
      qResult = binding.query(soqlQuery);
      Boolean done = false;
      if (qResult.size > 0)
      {
         Console.WriteLine("Logged-in user can see a total of "
            + qResult.size + " contact records.");
         while (!done)
         {
            sObject[] records = qResult.records;
            for (int i = 0; i < records.Length; ++i)
            {
```

```
                Contact con = (Contact)records[i];
                String fName = con.FirstName;
                String lName = con.LastName;
                if (fName == null)
                {
                   Console.WriteLine("Contact " + (i + 1) + ": " + lName);
                }
                else
                {
                   Console.WriteLine("Contact " + (i + 1) + ": " + fName
                         + " " + lName);
                }
             }
             if (qResult.done)
             {
                done = true;
             }
             else
             {
                qResult = binding.queryMore(qResult.queryLocator);
             }
          }
       }
       else
       {
          Console.WriteLine("No records found.");
       }
       Console.WriteLine("\nQuery succesfully executed.");
    }
    catch (SoapException e)
    {
       Console.WriteLine("An unexpected error has occurred: " +
                                  e.Message + "\n" + e.StackTrace);
    }
}
```

Arguments

Name	Type	Description
`queryString`	string	Query string that specifies the object to query, the fields to return, and any conditions for including a specific object in the query. For more information, see the *Salesforce SOQL and SOSL Reference Guide*.

Response

QueryResult

Faults

`MalformedQueryFault`

`InvalidSObjectFault`

`InvalidFieldFault`

UnexpectedErrorFault

See Also:

queryAll()
queryMore()
API Call Basics
https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

QueryResult

The `query()` call returns a `QueryResult` object, which has the following properties:

Name	Type	Description
`queryLocator`	QueryLocator	A specialized string, similar to ID. Used in `queryMore()` for retrieving subsequent sets of objects from the query results, if applicable. Represents a server-side cursor. Each user can have up to ten query cursors open at a time.
`done`	boolean	Indicates whether additional rows need to be retrieved from the query results (`false`) using `queryMore()`, or not (`true`). Your client application can use this value as a loop condition while iterating through the query results.
`records`	sObject[]	Array of sObjects representing individual objects of the specified object and containing data defined in the field list specified in the queryString. For information on queries that use a `GROUP BY` clause, see AggregateResult.
`size`	int	Your client application can use this value to determine whether the query retrieved any rows (size > 0) or not (size = 0). Total number of rows retrieved in the query.

AggregateResult

This object contains the results returned by a `query()` if the query contains an aggregate function, such as `MAX()`. AggregateResult is an sObject, but unlike other sObject objects such as Contact, it is read-only and it is only used for query results.

The QueryResult object has a `records` field that is an array of sObject records matching your query. For example, the following query returns an array of Contact records in the `records` field.

```
SELECT Id, LastName
FROM Contact
WHERE FirstName = 'Bob'
```

When a SOQL query contains an aggregate function, the results are a set of aggregated data instead of an array of records for a standard object, such as Contact. Therefore, the `records` field returns an array of AggregateResult records.

For more information on aggregate functions, see “Aggregate Functions” in the *Salesforce SOQL and SOSL Reference Guide*.

Fields

Each AggregateResult object contains a separate field for each of the items in the `SELECT` list. For the enterprise WSDL, retrieve the result for each item by calling `getField()` on an AggregateResult object when using WSC client framework. For the partner WSDL, retrieve the result for each item by calling `getField()` on an sObject object.

See Sample Code—Java and Sample Code—C# for examples that work with the enterprise WSDL.

Sample Code—Java

```
public void queryAggregateResult() {
  try {
    String groupByQuery = "SELECT Account.Name n, " +
        "MAX(Amount) max, MIN(Amount) min " +
        "FROM Opportunity GROUP BY Account.Name";
    QueryResult qr = connection.query(groupByQuery);
    if (qr.getSize() > 0) {
      System.out.println("Query returned " +
          qr.getRecords().length +  " results."
      );
      for (SObject sObj : qr.getRecords()) {
        AggregateResult result = (AggregateResult) sObj;
        System.out.println("aggResult.Account.Name: " +
            result.getField("n")
        );
        System.out.println("aggResult.max: " +
            result.getField("max")
        );
        System.out.println("aggResult.min: " +
            result.getField("min")
        );
        System.out.println();
      }
    } else {
      System.out.println("No results found.");
    }
    System.out.println("\nQuery successfully executed.");
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

Sample Code—C#

```
private void testAggregateResult()
{
   try
   {
      QueryResult qr = null;

      binding.QueryOptionsValue = new QueryOptions();

      String soqlStr = "SELECT Name, " +
                       "MAX(Amount), " +
                       "MIN(Amount) " +
                       "FROM Opportunity " +
                       "GROUP BY Name";

      qr = binding.query(soqlStr);

      if (qr.size > 0)
      {

         for (int i = 0; i < qr.records.Length; i++)
         {

                sforce.AggregateResult ar = (AggregateResult)qr.records[i];

                foreach (XmlElement e in ar.Any)
                    Console.WriteLine(
                        "{0} - {1}",
                        e.LocalName,
```

```
                        e.InnerText
                    );

              }
           }
           else
           {
              Console.WriteLine("No records found");
           }
           Console.WriteLine("Query successfully executed.");
        }
        catch (Exception ex)
        {
           Console.WriteLine(
               "\nFailed to execute query successfully." +
               "error message was: \n" +
               ex.Message
           );

        }
}
```

QueryLocator

In the QueryResult object returned by the `query()` call, `queryLocator` contains a value that you will use in the subsequent `queryMore()` call. Note the following guidelines:

- Use a given `queryLocator` value only once. When you pass it in a `queryMore()` call, the API returns a new `queryLocator` in the QueryResult.
- `QueryLocator` objects expire automatically after 15 minutes of inactivity.
- A user can have up to 10 query cursors open at a time. If 10 `QueryLocator` cursors are open when a client application, logged in as the same user, attempts to open a new one, then the oldest of the 10 cursors is released. If the client application attempts to open the released query cursor, an error results.

Note: Cursor limits for different Force.com features are tracked separately. For example, you can have 10 query cursors open and 10 Metadata API cursors at the same time.

A `QueryLocator` represents a server-side cursor.

`queryAll()`

Retrieves data from specified objects, whether or not they have been deleted.

Syntax

```
QueryResult = connection.queryAll(string queryString);
```

Usage

Use `queryAll` to identify the records that have been deleted because of a merge or delete. `queryAll` has read-only access to the field `isDeleted`; otherwise it is the same as `query()`.

To find records that have been deleted (in preparation for undeleting them with the `undelete()` call), specify `isDeleted = true` in the query string, and for merged records, request the `masterRecord`. For example:

```
SELECT id, isDeleted, masterRecordId FROM Account WHERE masterRecordId='100000000000Abc'
```

You can use `queryAll()` to query on all Task and Event records, archived or not. You can also filter on the `isArchived` field to find only the archived objects. You cannot use `query()` as it automatically filters out all records where `isArchived` is set to `true`. You can update or delete archived records, though you cannot update the `isArchived` field. If you use the API to insert activities that meet the criteria listed below, the activities will be archived during the next run of the archival background process.

For additional information about using queryAll, see `query()`.

Sample Code—Java

This sample performs a query to get all the accounts, whether they're deleted or not. It sets a custom batch size of 250 records. It fetches all batches of records by calling `queryAll()` the first time and then `queryMore()`. The names and the value of the isDeleted fields of all returned accounts are written to the console.

```
public void queryAllRecords() {
   // Setting custom batch size
   connection.setQueryOptions(250);

   try {
      String soqlQuery = "SELECT Name, IsDeleted FROM Account";
      QueryResult qr = connection.queryAll(soqlQuery);
      boolean done = false;
      if (qr.getSize() > 0) {
         System.out.println("Logged-in user can see a total of "
               + qr.getSize()
               + " contact records (including deleted records).");
         while (!done) {
            SObject[] records = qr.getRecords();
            for (int i = 0; i < records.length; i++) {
               Account account = (Account) records[i];
               boolean isDel = account.getIsDeleted();
               System.out.println("Account " + (i + 1) + ": "
                     + account.getName() + " isDeleted = "
                     + account.getIsDeleted());
            }
            if (qr.isDone()) {
               done = true;
            } else {
               qr = connection.queryMore(qr.getQueryLocator());
            }
         }
      } else {
         System.out.println("No records found.");
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample performs a query to get all the accounts, whether they're deleted or not. It sets a custom batch size of 250 records. It fetches all batches of records by calling `queryAll()` the first time and then `queryMore()`. The names and the value of the isDeleted fields of all returned accounts are written to the console.

```
public void queryAllRecords()
{
   // Setting custom batch size
   QueryOptions qo = new QueryOptions();
   qo.batchSize = 250;
   qo.batchSizeSpecified = true;
   binding.QueryOptionsValue = qo;

   try
   {
      String soqlQuery = "SELECT Name, IsDeleted FROM Account";
      QueryResult qr = binding.queryAll(soqlQuery);
      Boolean done = false;
      if (qr.size > 0)
      {
         Console.WriteLine("Logged-in user can see a total of "
               + qr.size
               + " contact records (including deleted records).");
         while (!done)
         {
            sObject[] records = qr.records;
            for (int i = 0; i < records.Length; i++)
            {
               Account account = (Account)records[i];
               Boolean isDel = (Boolean)account.IsDeleted;
               Console.WriteLine("Account " + (i + 1) + ": "
                     + account.Name + " isDeleted = "
                     + account.IsDeleted);
            }
            if (qr.done)
            {
               done = true;
            }
            else
            {
               qr = binding.queryMore(qr.queryLocator);
            }
         }
      }
      else
      {
         Console.WriteLine("No records found.");
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                                e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

Name	Type	Description
`queryString`	string	Query string that specifies the object to query, the fields to return, and any conditions for including a specific object in the query. For more information, see the *Salesforce SOQL and SOSL Reference Guide*.

Response

QueryResult

Faults

`MalformedQueryFault`

`InvalidSObjectFault`

`InvalidFieldFault`

`UnexpectedErrorFault`

See Also:

API Call Basics
queryMore()

queryMore()

Retrieves the next batch of objects from a `query()`.

Syntax

```
QueryResult = connection.queryMore( QueryLocator  QueryLocator);
```

Usage

Use this call to process `query()` calls that retrieve a large number of records (by default, more than 500) in the result set. The `query()` call retrieves the first 500 records and creates a server-side cursor that is represented in the queryLocator object. The `queryMore()` call processes subsequent records in up to 500-record chunks, resets the server-side cursor, and returns a newly generated QueryLocator. To iterate through records in the result set, you generally call `queryMore()` repeatedly until all records in the result set have been processed (the `Done` flag is `true`). You can change the maximum number of records returned to up to 2,000. See Changing the Batch Size in Queries in the *Salesforce SOQL and SOSL Reference Guide* for more information.

Note: A `queryMore()` call on a parent object invalidates all child cursors in the previous result set. If you need the results from the child, you must use `queryMore()` on those results before using `queryMore()` on the parent results.

You can't use `queryMore()` if a query includes a `GROUP BY` clause. See `GROUP BY` and `queryMore()` in the *Salesforce SOQL and SOSL Reference Guide* for more information.

Sample Code—Java

This sample executes a query that fetches the first names and last names of all contacts. It calls `query()` with the query string to get the first batch of records. It then calls `queryMore()` in a loop to get subsequent batches of records until no records are returned. It writes the first and last names of the contacts queried to the console.

```
public void queryRecords() {
   QueryResult qResult = null;
   try {
      String soqlQuery = "SELECT FirstName, LastName FROM Contact";
      qResult = connection.query(soqlQuery);
      boolean done = false;
      if (qResult.getSize() > 0) {
         System.out.println("Logged-in user can see a total of "
```

```
            + qResult.getSize() + " contact records.");
         while (!done) {
            SObject[] records = qResult.getRecords();
            for (int i = 0; i < records.length; ++i) {
               Contact con = (Contact) records[i];
               String fName = con.getFirstName();
               String lName = con.getLastName();
               if (fName == null) {
                  System.out.println("Contact " + (i + 1) + ": " + lName);
               } else {
                  System.out.println("Contact " + (i + 1) + ": " + fName
                        + " " + lName);
               }
            }
            if (qResult.isDone()) {
               done = true;
            } else {
               qResult = connection.queryMore(qResult.getQueryLocator());
            }
         }
      } else {
         System.out.println("No records found.");
      }
      System.out.println("\nQuery succesfully executed.");
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample executes a query that fetches the first names and last names of all contacts. It calls `query()` with the query string to get the first batch of records. It then calls `queryMore()` in a loop to get subsequent batches of records until no records are returned. It writes the first and last names of the contacts queried to the console.

```
public void queryRecords()
{
   QueryResult qResult = null;
   try
   {
      String soqlQuery = "SELECT FirstName, LastName FROM Contact";
      qResult = binding.query(soqlQuery);
      Boolean done = false;
      if (qResult.size > 0)
      {
         Console.WriteLine("Logged-in user can see a total of "
            + qResult.size + " contact records.");
         while (!done)
         {
            sObject[] records = qResult.records;
            for (int i = 0; i < records.Length; ++i)
            {
               Contact con = (Contact)records[i];
               String fName = con.FirstName;
               String lName = con.LastName;
               if (fName == null)
               {
                  Console.WriteLine("Contact " + (i + 1) + ": " + lName);
               }
               else
               {
                  Console.WriteLine("Contact " + (i + 1) + ": " + fName
                        + " " + lName);
               }
            }
            if (qResult.done)
            {
```

```
                done = true;
             }
             else
             {
                qResult = binding.queryMore(qResult.queryLocator);
             }
          }
       }
       else
       {
          Console.WriteLine("No records found.");
       }
       Console.WriteLine("\nQuery succesfully executed.");
    }
    catch (SoapException e)
    {
       Console.WriteLine("An unexpected error has occurred: " +
                                     e.Message + "\n" + e.StackTrace);
    }
}
```

Arguments

Name	Type	Description
queryLocator	QueryLocator	Represents the server-side cursor that tracks the current processing location in the query result set.

Response

QueryResult

Faults

InvalidQueryLocatorFault

UnexpectedErrorFault

See Also:

query()
API Call Basics
https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

QueryResult

The `queryMore()` call returns a `QueryResult` object, which has the following properties:

Name	Type	Description
queryLocator	QueryLocator	A specialized string, similar to ID. Used in the subsequent `queryMore()` call for retrieving sets of objects from the query results, if applicable.
done	boolean	Indicates whether additional rows need to be retrieved from the query results (`false`) using another `queryMore()` call, or not (`true`). Your client application can use this value as a loop condition while iterating through the query results.
records	sObject[]	Array of sObjects representing individual objects of the specified object and containing data defined in the field list specified in the queryString.

Name	Type	Description
size	int	Total number of rows retrieved in the query. Your client application can use this value to determine whether the query retrieved any rows (size > 0) or not (size = 0).

Note: A queryMore() call on a parent object invalidates all child cursors in the previous result set. If you need the results from the child, you must use queryMore() on those results before using queryMore() on the parent results.

QueryLocator

In the QueryResult object returned by the queryMore() call, queryLocator contains a value that you will use in the subsequent queryMore() call. Note the following guidelines for using this value:

- Use a queryLocator only once. When you pass it in a queryMore() call, the API returns a new queryLocator in the QueryResult.
- The queryLocator value expires automatically after 15 minutes of inactivity.
- A user can have up to ten query cursors open at a time. If ten QueryLocator cursors are opened when a client application with the same logged-in user attempts to open a new cursor, then the oldest of the ten cursors is released.

A QueryLocator represents a server-side cursor.

Note: A queryMore() call on a parent object invalidates all child cursors in the previous result set. If you need the results from the child, you must use queryMore() on those results before using queryMore() on the parent results.

retrieve()

Retrieves one or more records based on the specified IDs.

Syntax

```
sObject[] result = connection.retrieve(string fieldList, string sObjectType, ID ids[]);
```

Usage

Use the retrieve() call to retrieve individual records from an object. The client application passes the list of fields to retrieve, the object, and an array of record IDs to retrieve. The retrieve() call does not return records that have been deleted.

In general, you use retrieve() when you know in advance the IDs of the records to retrieve. Use query() instead to obtain records when you do not know the IDs or when you want to specify other selection criteria.

Client applications can use retrieve() to perform a client-side join. For example, a client application can run a query() to obtain a set of Opportunity records, iterate through the returned opportunity records, obtain the accountId for each opportunity, and then call retrieve() to obtain Account information for those accountIds.

Records for certain objects cannot be retrieved via the API. To retrieve a record via the retrieve() call, its object must be configured as retrieveable (retrieveable is true). To determine whether an object can be retrieved, your client application can invoke the describeSObjects() call on the object and inspect its retrievable property.

Your client application must be logged in with sufficient access rights to retrieve records within the specified object and to retrieve the fields in the specified field list. For more information, see Factors that Affect Data Access.

Sample Code—Java

This sample retrieves the Id, Name, and Website of the specified Account records. It writes the fields of the retrieved records to the console.

```
public void retrieveRecords(String[] ids) {
   try {
      SObject[] sObjects = connection.retrieve("ID, Name, Website",
            "Account", ids);
      // Verify that some objects were returned.
      // Even though we began with valid object IDs,
      // someone else might have deleted them in the meantime.
      if (sObjects != null) {
         for (int i = 0; i < sObjects.length; i++) {
            // Cast the SObject into an Account object
            Account retrievedAccount = (Account) sObjects[i];
            if (retrievedAccount != null) {
               System.out.println("Account ID: " + retrievedAccount.getId());
               System.out.println("Account Name: " + retrievedAccount.getName());
               System.out.println("Account Website: "
                     + retrievedAccount.getWebsite());
            }
         }
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample retrieves the Id, Name, and Website of the specified Account records. It writes the fields of the retrieved records to the console.

```
public void retrieveRecords(String[] ids)
{
   try
   {
      sObject[] sObjects = binding.retrieve("ID, Name, Website",
            "Account", ids);
      // Verify that some objects were returned.
      // Even though we began with valid object IDs,
      // someone else might have deleted them in the meantime.
      if (sObjects != null)
      {
         for (int i = 0; i < sObjects.Length; i++)
         {
            // Cast the SObject into an Account object
            Account retrievedAccount = (Account)sObjects[i];
            if (retrievedAccount != null)
            {
               Console.WriteLine("Account ID: " + retrievedAccount.Id);
               Console.WriteLine("Account Name: " + retrievedAccount.Name);
               Console.WriteLine("Account Website: "
                     + retrievedAccount.Website);
            }
         }
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                                   e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

Name	Type	Description
`fieldList`	string	List of one or more fields in the specified object, separated by commas. You must specify valid field names and must have read-level permissions to each specified field. The fieldList defines the ordering of fields in the `result`.
`sObjectType`	string	Object from which to retrieve data. The specified value must be a valid object for your organization. For a complete list of objects, see Standard Objects.
`ids`	ID[]	Array of one or more IDs of the objects to retrieve. You can pass a maximum of 2000 object IDs to the `retrieve()` call. For information on IDs, see ID Field Type.

Response

Name	Type	Description
`result`	sObject[]	Array of one or more sObjects representing individual records of the specified object. The number of sObjects returned in the array matches the number of IDs passed into the `retrieve()` call. If you do not have access to an object or if a passed ID is invalid, the array returns `null` for that object. For information on IDs, see ID Field Type.

Faults

`InvalidSObjectFault`

`InvalidFieldFault`

`UnexpectedErrorFault`

See Also:

API Call Basics
https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

search()

Executes a text search in your organization's data.

Syntax

```
SearchResult = connection.search(String searchString);
```

Usage

Use `search()` to search for records based on a search string. The `search()` call supports searching custom objects. For an extensive discussion about the syntax and rules used for text searches, see the *Salesforce SOQL and SOSL Reference Guide*.

Certain objects cannot be searched via the API, such as Attachment objects. To search an object via the `search()` call, the object must be configured as searchable (`isSearchable` is `true`). To determine whether an object can be searched, your client application can invoke the `describeSObjects()` call on the object and inspect its `searchable` property.

Sample Code—Java

This sample makes the `search()` call by passing it a SOSL query, which returns contacts, leads, and accounts whose phone fields contain a specified value. Next, it gets the sObject records from the results and stores the records in arrays depending on the record type. Finally, it writes the fields of the returned contacts, leads, and accounts to the console.

```
public void searchSample() {
   try {
      // Perform the search using the SOSL query.
      SearchResult sr = connection.search(
            "FIND {4159017000} IN Phone FIELDS RETURNING "
            + "Contact(Id, Phone, FirstName, LastName), "
            + "Lead(Id, Phone, FirstName, LastName), "
            + "Account(Id, Phone, Name)");

      // Get the records from the search results.
      SearchRecord[] records = sr.getSearchRecords();

      ArrayList<Contact> contacts = new ArrayList<Contact>();
      ArrayList<Lead> leads = new ArrayList<Lead>();
      ArrayList<Account> accounts = new ArrayList<Account>();

      // For each record returned, find out if it's a
      // contact, lead, or account and add it to the
      // appropriate array, then write the records
      // to the console.
      if (records.length > 0) {
         for (int i = 0; i < records.length; i++) {
            SObject record = records[i].getRecord();
            if (record instanceof Contact) {
               contacts.add((Contact) record);
            } else if (record instanceof Lead) {
               leads.add((Lead) record);
            } else if (record instanceof Account) {
               accounts.add((Account) record);
            }
         }

         System.out.println("Found " + contacts.size() + " contacts.");
         for (Contact c : contacts) {
            System.out.println(c.getId() + ", " + c.getFirstName() + ", "
                  + c.getLastName() + ", " + c.getPhone());
         }
         System.out.println("Found " + leads.size() + " leads.");
         for (Lead d : leads) {
            System.out.println(d.getId() + ", " + d.getFirstName() + ", "
                  + d.getLastName() + ", " + d.getPhone());
         }
         System.out.println("Found " + accounts.size() + " accounts.");
         for (Account a : accounts) {
            System.out.println(a.getId() + ", " + a.getName() + ", "
                  + a.getPhone());
         }
      } else {
         System.out.println("No records were found for the search.");
      }
   } catch (Exception ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample makes the `search()` call by passing it a SOSL query, which returns contacts, leads, and accounts whose phone fields contain a specified value. Next, it gets the sObject records from the results and stores the records in arrays depending on the record type. Finally, it writes the fields of the returned contacts, leads, and accounts to the console.

```
public void searchSample()
{
   try
   {
      // Perform the search using the SOSL query.
      SearchResult sr = binding.search(
         "FIND {4159017000} IN Phone FIELDS RETURNING "
         + "Contact(Id, Phone, FirstName, LastName), "
         + "Lead(Id, Phone, FirstName, LastName), "
         + "Account(Id, Phone, Name)");

      // Get the records from the search results.
      SearchRecord[] records = sr.searchRecords;

      List<Contact> contacts = new List<Contact>();
      List<Lead> leads = new List<Lead>();
      List<Account> accounts = new List<Account>();

      // For each record returned, find out if it's a
      // contact, lead, or account and add it to the
      // appropriate array, then write the records
      // to the console.
      if (records.Length > 0)
      {
         for (int i = 0; i < records.Length; i++)
         {
            sObject record = records[i].record;
            if (record is Contact)
            {
               contacts.Add((Contact)record);
            }
            else if (record is Lead)
            {
               leads.Add((Lead)record);
            }
            else if (record is Account)
            {
               accounts.Add((Account)record);
            }
         }

         Console.WriteLine("Found " + contacts.Count + " contacts.");
         foreach (Contact c in contacts)
         {
            Console.WriteLine(c.Id + ", " +
               c.FirstName + ", " +
               c.LastName + ", " +
               c.Phone);
         }
         Console.WriteLine("Found " + leads.Count + " leads.");
         foreach (Lead d in leads)
         {
            Console.WriteLine(d.Id + ", " +
               d.FirstName + ", " +
               d.LastName + ", " +
               d.Phone);
         }
         Console.WriteLine("Found " + accounts.Count + " accounts.");
         foreach (Account a in accounts)
         {
            Console.WriteLine(a.Id + ", " +
               a.Name + ", " +
```

```
                a.Phone);
         }
      }
      else
      {
         Console.WriteLine("No records were found for the search.");
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                                 e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

Name	Type	Description
search	string	Search string that specifies the text expression to search for, the scope of fields to search, the list of objects and fields to retrieve, and the maximum number of records to return. For more information, see the *Salesforce SOQL and SOSL Reference Guide*.

Response

SearchResult

Fault

InvalidFieldFault

InvalidSObjectFault

MalformedSearchFault

UnexpectedErrorFault

See Also:

API Call Basics
https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

SearchResult

The search() call returns a SearchResult object, which has the following properties:

Name	Type	Description
searchRecords	SearchRecord[]	Array of SearchRecord objects, each of which contains an sObject.

undelete()

Undeletes records from the Recycle Bin.

Syntax

```
UndeleteResult[] = connection.undelete(ID[] ids );
```

Usage

Use this call to restore any deleted record that is undeletable. Undeletable records include those in the Recycle Bin. Records can be put in the Recycle Bin as the result of a `merge()` or `delete()` call. You can identify deleted records, including records deleted as the result of a merge, using the `queryAll()` call.

You should verify that a record can be undeleted before attempting to delete it. Some records cannot be undeleted, for example, Account records can be undeleted, but not AccountTeamMember records. To verify that a record can be undeleted, check that the value of the `undeletable` flag in the DescribeSObjectResult for that object is set to `true`.

Since a delete call cascade-deletes child records, an undelete call will undelete the cascade-deleted records. For example, deleting an account will delete all the contacts associated with that account.

You can undelete records that were deleted as the result of a merge, but the child objects will have been re-parented, which cannot be undone.

Note:

Starting with API version 15.0, if you specify a value for a field that contains a string, and the value is too big for the field, the call fails and an error is returned. In previous versions of the API the value was truncated and the call succeeded. If you wish to keep the old behavior with versions 15.0 and later, use the AllowFieldTruncationHeader SOAP header.

This call supports the AllOrNoneHeader, AllowFieldTruncationHeader, and CallOptions headers.

Rollback on Error

The AllOrNoneHeader header allows you to roll back all changes unless all records are processed successfully. This header is available in API version 20.0 and later. The default behavior is to allow partial success of a call: records without errors are committed, while records with errors are marked as failed in the call results.

Sample Code—Java

This sample calls `queryAll()` to get the last five deleted accounts. It then passes the IDs of these accounts to `undelete()`, which restores these accounts. Finally, it checks the results of the call and writes the IDs of the restored accounts or any errors to the console.

```
public void undeleteRecords() {
   try {
      // Get the accounts that were last deleted
      // (up to 5 accounts)
      QueryResult qResult = connection
            .queryAll("SELECT Id, SystemModstamp FROM "
                  + "Account WHERE IsDeleted=true "
                  + "ORDER BY SystemModstamp DESC LIMIT 5");

      String[] Ids = new String[qResult.getSize()];
      // Get the IDs of the deleted records
      for (int i = 0; i < qResult.getSize(); i++) {
         Ids[i] = qResult.getRecords()[i].getId();
      }

      // Restore the records
      UndeleteResult[] undelResults = connection.undelete(Ids);

      // Check the results
```

```
      for (UndeleteResult result : undelResults) {
         if (result.isSuccess()) {
            System.out.println("Undeleted Account ID: " + result.getId());
         } else {
            if (result.getErrors().length > 0) {
               System.out.println("Error message: "
                     + result.getErrors()[0].getMessage());
            }
         }
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample calls `queryAll()` to get the last five deleted accounts. It then passes the IDs of these accounts to `undelete()`, which restores these accounts. Finally, it checks the results of the call and writes the IDs of the restored accounts or any errors to the console.

```
public void undeleteRecords()
{
   try
   {
      // Get the accounts that were last deleted
      //   (up to 5 accounts)
      QueryResult qResult = binding.queryAll(
            "SELECT Id, SystemModstamp FROM " +
            "Account WHERE IsDeleted=true " +
            "ORDER BY SystemModstamp DESC LIMIT 5");

      String[] Ids = new String[qResult.size];
      // Get the IDs of the deleted records
      for (int i = 0; i < qResult.size; i++)
      {
         Ids[i] = qResult.records[i].Id;
      }

      // Restore the records
      UndeleteResult[] undelResults = binding.undelete(Ids);

      // Check the results
      foreach (UndeleteResult result in undelResults)
      {
         if (result.success)
         {
            Console.WriteLine("Undeleted Account ID: " +
                  result.id);
         }
         else
         {
            if (result.errors.Length > 0)
            {
               Console.WriteLine("Error message: " +
                     result.errors[0].message);
            }
         }
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                           e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

Name	Type	Description
ids	ID[]	IDs of the records to be restored.

Response

UndeleteResult

Faults

`UnexpectedErrorFault`

See Also:

delete()

UndeleteResult

The `undelete()` call returns an undeleteResult object with the following properties:

Name	Type	Description
Id	ID	ID of the record being undeleted.
success	boolean	Indicates whether the undelete was successful (`true`) or not (`false`).
errors	Error[]	If an error occurred during the `undelete()` call, an array of one or more Error objects providing the error code and description.

`update()`

Updates one or more existing records in your organization's data.

Syntax

```
SaveResult[] = connection.update(sObject[] sObjects);
```

Usage

Use this call to update one or more existing records, such as accounts or contacts, in your organization's data. The `update()` call is analogous to the UPDATE statement in SQL.

Permissions

Your client application must be logged in with sufficient access rights to `update()` records objects for the specified object, as well as individual fields inside that object. For more information, see Factors that Affect Data Access.

Special Handling

Certain objects—and certain fields within those objects—require special handling or permissions. For example, you might also need permissions to access an object's parent object. Before you attempt to update a record for a particular object, be sure to read its description in the Standard Objects and in the Salesforce online help.

Updateable Objects

Certain records cannot be updated via the API. To update a record via the `update()` call, its object must be configured as updateable (`updateable` is `true`). To determine whether an object can be updated, your client application can invoke the `describeSObjects()` call on the object and inspect its `updateable` property.

Required Fields

When updating required fields, you must supply a value—you cannot set the value to `null`. For more information, see Required Fields.

ID Fields

Fields whose names contain "`Id`" are either that object's primary key (see ID Field Type) or a foreign key (see Reference Field Type). Client applications cannot update primary keys, but they can update foreign keys. For example, a client application can update the `OwnerId` of an Account, because `OwnerID` is a foreign key that refers to the user who owns the account record. Use `describeSObjects()` to confirm whether the field can be updated.

This call checks a batch for duplicate `Id` values, and if there are duplicates, the first six are processed. For additional duplicate `Id` values, the SaveResult for those entries is marked with an error similar to the following:

```
Maximum number of duplicate updates in one batch (6 allowed).
```

Automatically Updated Fields

The API updates certain fields automatically, such as `LastModifiedDate`, `LastModifiedById`, and `SystemModstamp`. You cannot explicitly specify these values in your `update()` call.

Resetting Values to null

To reset a field value to `null`, you add the field name to the `fieldsToNull` array in the sObject. You cannot set required fields (`nillable` is `false`) to `null`.

Valid Field Values

You must supply values that are valid for the field's data type, such as integers (not alphabetic characters) for integer fields. In your client application, follow the data formatting rules specified for your programming language and development tool (your development tool will handle the appropriate mapping of data types in SOAP messages).

String Values

When storing values in string fields, the API trims any leading and trailing white space. For example, if the value of a name field is entered as `" ABC Company "`, then the value is stored in the database as `"ABC Company"`.

Starting with API version 15.0, if you specify a value for a field that contains a string, and the value is too big for the field, the call fails and an error is returned. In previous versions of the API the value was truncated and the call succeeded. If you wish to keep the old behavior with versions 15.0 and later, use the AllowFieldTruncationHeader SOAP header.

Assignment Rules

When updating Case or Lead objects, your client application can set AssignmentRuleHeader options to have the case or lead automatically assigned to one or more users based on assignment rules configured in the Salesforce user interface. For more information, see Case or Lead.

Maximum Number of Objects Updated

Your client application can change up to 200 records in a single `update()` call. If an update request exceeds 200 records, then the entire operation fails.

Rollback on Error

The AllOrNoneHeader header allows you to roll back all changes unless all records are processed successfully. This header is available in API version 20.0 and later. The default behavior is to allow partial success of a call: records without errors are committed, while records with errors are marked as failed in the call results.

Automatic Subscriptions for Chatter Feeds

To subscribe to records they create, users must enable the `Automatically follow records that I create` option in their personal settings. If users have automatic subscriptions enabled, they automatically follow the records they create and see changes to those records in their Chatter feed on the Home tab.

When you update the owner of a record, the new owner is not automatically subscribed to the record, unless the new owner has automatic subscriptions for records enabled in his or her Chatter feed settings. The previous owner is not automatically unsubscribed. If the new owner has automatic subscriptions for records enabled, the new and previous owners both see any changes to the record in their news feed.

A user can subscribe to a record or to another user. Changes to the record and updates from the users are displayed in the Chatter feed on the user's home page, which is a useful way to stay up-to-date with other users and with changes made to records in Salesforce. Feeds are available in API version 18.0 and later.

Updating Records for Different Object Types

You can update records for multiple object types, including custom objects, in one call with API version 20.0 and later. For example, you could update a contact and an account in one call. You can update records for up to 10 objects types in one call.

Records are saved in the same order that they are entered in the `sObjects` input array.

Records for different object types are broken into multiple chunks by Salesforce. A chunk is a subset of the `sObjects` input array and each chunk contains records of one object type. Data is committed on a chunk-by-chunk basis. Any Apex triggers related to the records in a chunk are invoked once per chunk. Consider an `sObjects` input array containing the following set of records:

```
account1, account2, contact1, contact2, contact3, case1, account3, account4, contact4
```

Salesforce splits the records into five chunks:

1. `account1, account2`
2. `contact1, contact2, contact3`
3. `case1`
4. `account3, account4`
5. `contact4`

Each call can process up to 10 chunks. If the `sObjects` array contains more than 10 chunks, you must process the records in more than one call.

Warning: You can't update records for multiple object types in one call if one of those types is related to a feature in the Setup area in Salesforce. The only exceptions are the following objects:

- Custom settings objects, which are similar to custom objects. For more information, see "Custom Settings Overview" in the Salesforce online help.
- GroupMember
- Group
- User if the following fields are not being updated:
  - ◊ `UserRoleId`
  - ◊ `IsActive`
  - ◊ `ForecastEnabled`
  - ◊ `IsPortalEnabled`
  - ◊ `Username`
  - ◊ `ProfileId`

`update()` and Foreign Keys

You can use external ID fields as a foreign key, which allows you to update a record and relate it to another existing record in a single step instead of querying the parent record ID first. To do this, set the foreign key to an instance of the parent sObject that has only the external ID field specified. This external ID should match the external ID value on the parent record.

The following Java and C# examples show you how to update an opportunity and relate it to an existing account using a custom external ID field named `MyExtId__c`. Each example has a method that accepts the ID of the opportunity to update. It creates an opportunity sObject and sets its ID field so that the object points to an existing opportunity to be updated, sets a new value for the stage name field, and then sets the external ID field to the account object. It then updates the opportunity. Once the opportunity is updated, the account becomes its parent and the state name is updated.

Java Example

```
public void updateForeignKeySample(String oppId) {
   try {
      Opportunity updateOpportunity = new Opportunity();
      // Point to an existing opportunity to update
      updateOpportunity.setId(oppId);
      updateOpportunity.setStageName("Qualification");

      Account parentAccountRef = new Account();
      parentAccountRef.setMyExtId__c("SAP1111111");
      updateOpportunity.setAccount(parentAccountRef);

      SaveResult[] results = connection
            .update(new SObject[] { updateOpportunity });
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

C# Example

```
public void updateForeignKeySample(String oppId)
{
   try
   {
      Opportunity updateOpportunity = new Opportunity();
      // Point to an existing opportunity to update
      updateOpportunity.Id = oppId;
      updateOpportunity.StageName = "Prospecting";
```

```
      Account parentAccountRef = new Account();
      parentAccountRef.MyExtId__c = "SAP1111111";
      updateOpportunity.Account = parentAccountRef;

      SaveResult[] results = binding.update(
         new sObject[] { updateOpportunity });
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                                 e.Message + "\n" + e.StackTrace);
   }
}
```

Basic Steps for Updating Records

Use this process to update records:

1. Determine the ID of each record that you want to `update()`. For example, you might call `query()` to retrieve a set of records (with their IDs), based on specific criteria, that you would want to update. If you know the ID of the record that you want to update, you can call `retrieve()` instead. For information on IDs, see ID Field Type.
2. Create an sObject for each record, and populate its fields with the data that you want to update.
3. Construct an sObject[] array and populate that array with the records that you want to update.
4. Call `update()`, passing in the sObject[] array.
5. Process the results in the SaveResult[] object to verify whether the records have been successfully updated.

Sample Code—Java

This sample accepts the IDs of the accounts to update. It creates two account sObjects, sets each with one of the passed IDs so that the sObject points to an existing account, and sets other fields. It then makes the `update()` call and verifies the results.

```
public void updateRecords(String[] ids) {
   Account[] updates = new Account[2];

   Account account1 = new Account();
   account1.setId(ids[0]);
   account1.setShippingPostalCode("89044");
   updates[0] = account1;

   Account account2 = new Account();
   account2.setId(ids[1]);
   account2.setNumberOfEmployees(1000);
   updates[1] = account2;

   // Invoke the update call and save the results
   try {
      SaveResult[] saveResults = connection.update(updates);
      for (SaveResult saveResult : saveResults) {
         if (saveResult.isSuccess()) {
            System.out.println("Successfully updated Account ID: "
                  + saveResult.getId());
         } else {
            // Handle the errors.
            // We just print the first error out for sample purposes.
            Error[] errors = saveResult.getErrors();
            if (errors.length > 0) {
               System.out.println("Error: could not update " + "Account ID "
                     + saveResult.getId() + ".");
               System.out.println("\tThe error reported was: ("
                     + errors[0].getStatusCode() + ") "
                     + errors[0].getMessage() + ".");
            }
         }
```

```
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

Sample Code—C#

This sample accepts the IDs of the accounts to update. It creates two account sObjects, sets each with one of the passed IDs so that the sObject points to an existing account, and sets other fields. It then makes the `update()` call and verifies the results.

```
public void updateRecords(String[] ids)
{
   Account[] updates = new Account[2];

   Account account1 = new Account();
   account1.Id = ids[0];
   account1.ShippingPostalCode = "89044";
   updates[0] = account1;

   Account account2 = new Account();
   account2.Id = ids[1];
   account2.NumberOfEmployees = 1000;
   updates[1] = account2;

   // Invoke the update call and save the results
   try
   {
      SaveResult[] saveResults = binding.update(updates);
      foreach (SaveResult saveResult in saveResults)
      {
         if (saveResult.success)
         {
            Console.WriteLine("Successfully updated Account ID: " +
                  saveResult.id);
         }
         else
         {
            // Handle the errors.
            // We just print the first error out for sample purposes.
            Error[] errors = saveResult.errors;
            if (errors.Length > 0)
            {
               Console.WriteLine("Error: could not update " +
                     "Account ID " + saveResult.id + "."
               );
               Console.WriteLine("\tThe error reported was: (" +
                     errors[0].statusCode + ") " +
                     errors[0].message + "."
               );
            }
         }
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                                 e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

Name	Type	Description
`sObjects`	sObject[]	Array of one or more records (maximum of 200) to update.

Response

SaveResult[]

Faults

`InvalidSObjectFault`

`UnexpectedErrorFault`

See Also:

API Call Basics

https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

SaveResult

The `update()` call returns an array of SaveResult objects. Each element in the SaveResult array corresponds to the sObject[] array passed as the `sObjects` parameter in the `update()` call. For example, the object returned in the first index in the SaveResult array matches the object specified in the first index of the sObject[] array.

A SaveResult object has the following properties:

Name	Type	Description
`id`	ID	ID of an sObject that you successfully updated. If this field contains a value, then the object was updated successfully. If this field is empty, then the object was not updated and the API returned error information instead.
`success`	boolean	Indicates whether the `update()` call succeeded (`true`) or not (`false`) for this object.
`errors`	Error[]	If an error occurred during the `update()` call, an array of one or more Error objects providing the error code and description.

`upsert()`

Creates new records and updates existing records; uses a custom field to determine the presence of existing records. In most cases, we recommend that you use `upsert()` instead of `create()` to avoid creating unwanted duplicate records (idempotent). Available in the API version 7.0 and later. You can process records for one more than object type in an `create()` or `update()` call, but all records must have the same object type in an `upsert()` call.

Note:

Starting with API version 15.0, if you specify a value for a field that contains a string, and the value is too big for the field, the call fails and an error is returned. In previous versions of the API the value was truncated and the call succeeded. If you wish to keep the old behavior with versions 15.0 and later, use the AllowFieldTruncationHeader SOAP header.

Syntax

```
UpsertResult[] = connection.upsert(String externalIdFieldName, sObject[] sObjects);
```

Usage

Upsert is a merging of the words insert and update. This call is available for objects if the object has an external ID field or a field with the idLookup field property.

On custom objects, this call uses an indexed custom field called an external ID to determine whether to create a new record or update an existing record. On standard objects, this call can use the name of any field with the idLookup field property instead of the external ID.

Note: External ID fields cannot be used with `merge()`.

For more information about adding custom fields, including external ID fields, to objects, see the "Adding Fields" topic in the Salesforce online help.

Using this call can dramatically reduce how many calls you need to make, particularly when:

- You are integrating your organization's Salesforce data with ERP (enterprise resource planning) systems such as accounting and manufacturing.
- You are importing data and want to prevent the creation of duplicate objects.

If you are upserting a record for an object that has a custom field with both the `External ID` and `Unique` attributes selected (a unique index), you do not need any special permissions, because the `Unique` attribute prevents the creation of duplicates. If you are upserting a record for an object that has the `External ID` attribute selected but not the `Unique` attribute selected, (a non-unique index) your client application must have the permission "View All Data" to execute this call. Having this permission prevents the client application from using `upsert()` to insert an accidental duplicate record because it couldn't see that the record existed.

Note: Matching by external ID is case-insensitive only if the external ID field has the `Unique` attribute and the `Treat "ABC" and "abc" as duplicate values (case insensitive))` option selected. These options are selected in the Salesforce user interface during field creation. If this is the case, "ABC123" is matched with "abc123." Before performing an operation, if you have external ID fields without the case-insensitive option selected, review your external IDs for any values that would be matched if case was not considered. If such values exist, you may want to modify them to make them unique, or select the case-sensitive option for your external ID fields. For more information about field attributes, see "Custom Field Attributes" in the Salesforce online help.

How Upsert Chooses to `update()` or `create()`

Upsert uses the external ID to determine whether it should create a new record or update an existing one:

- If the external ID is not matched, then a new record is created.
- If the external ID is matched once, then the existing record is updated.
- If the external ID is matched multiple times, then an error is reported.
- When batch updating multiple records where the external ID is the same for two or more records in your batch call, those records will be marked as errors in the UpsertResult file. The records will be neither created or updated.

Rollback on Error

The AllOrNoneHeader header allows you to roll back all changes unless all records are processed successfully. This header is available in API version 20.0 and later. The default behavior is to allow partial success of a call: records without errors are committed, while records with errors are marked as failed in the call results.

Automatic Subscriptions for Chatter Feeds

To subscribe to records they create, users must enable the `Automatically follow records that I create` option in their personal settings. If users have automatic subscriptions enabled, they automatically follow the records they create and see changes to those records in their Chatter feed on the Home tab.

When you update the owner of a record, the new owner is not automatically subscribed to the record, unless the new owner has automatic subscriptions for records enabled in his or her Chatter feed settings. The previous owner is not automatically unsubscribed. If the new owner has automatic subscriptions for records enabled, the new and previous owners both see any changes to the record in their news feed.

A user can subscribe to a record or to another user. Changes to the record and updates from the users are displayed in the Chatter feed on the user's home page, which is a useful way to stay up-to-date with other users and with changes made to records in Salesforce. Feeds are available in API version 18.0 and later.

`upsert()` and Foreign Keys

You can use external ID fields as a foreign key, which allows you to create or update a record and relate it to another existing record in a single step instead of querying the parent record ID first. To do this, set the foreign key to an instance of the parent sObject that has only the external ID field specified. This external ID should match the external ID value on the parent record. Unlike `create()`, the parent record must already exist when using `upsert()` to create or update a child record related by a foreign key.

The following Java and C# examples upsert an opportunity. In this case, the opportunity doesn’t exist in the database, so the `upsert()` call will create it. The opportunity references an existing account. Rather than specify the account ID, which would require a separate query to obtain, we specify an external ID for the account, in this example the `MyExtId__c` custom field.

Java Example

```
public void upsertForeignKeySample() {
   try {
      Opportunity newOpportunity = new Opportunity();
      newOpportunity.setName("UpsertOpportunity");
      newOpportunity.setStageName("Prospecting");
      Calendar dt = connection.getServerTimestamp().getTimestamp();
      dt.add(Calendar.DAY_OF_MONTH, 7);
      newOpportunity.setCloseDate(dt);
      newOpportunity.setMyExtId__c("UPSERTID001");

      // Parent Account record must already exist
      Account parentAccountRef = new Account();
      parentAccountRef.setMyExtId__c("SAP111111");
      newOpportunity.setAccount(parentAccountRef);

      SaveResult[] results = connection
            .upsert("MyExtId__c", new SObject[] { newOpportunity });
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

C# Example

```
public void upsertForeignKeySample()
{
   try
   {
      Opportunity newOpportunity = new Opportunity();
      newOpportunity.Name = "UpsertOpportunity";
      newOpportunity.StageName = "Prospecting";
      DateTime dt = (DateTime)binding.getServerTimestamp().timestamp;
      newOpportunity.CloseDate = dt.AddDays(7);
      newOpportunity.CloseDateSpecified = true;
```

```
        newOpportunity.MyExtId__c = "UPSERTID001";

        // Parent Account record must already exist
        Account parentAccountRef = new Account();
        parentAccountRef.MyExtId__c = "SAP111111";
        newOpportunity.Account = parentAccountRef;

        SaveResult[] results = binding
              .upsert("MyExtId", new sObject[] { newOpportunity });
     }
     catch (SoapException e)
     {
        Console.WriteLine("An unexpected error has occurred: " +
                                     e.Message + "\n" + e.StackTrace);
     }
}
```

`upsert()` and Polymorphic Foreign Keys

A polymorphic key is an ID that can refer to more than one type of object as a parent. For example, the `OwnerId` field on the Case object can reference either a User or a Group. Similarly, the `WhoID` field on the Task object can reference either a Contact or a Lead.

With the Enterprise WSDL, such polymorphic foreign key fields are defined as a Name sObject type, so it is not possible to pass in the correct sObject in an `upsert()` call. However, with the Partner WSDL all foreign key fields—including polymorphic ones—are of type sObject , which allows `upsert()` calls on such fields.

Important: If your organization has SOQL Polymorphism enabled, polymorphic relationship fields point to sObjects and not Names in the Enterprise WSDL; therefore, you can pass the correct sObject in a call. SOQL Polymorphism is currently available as a Developer Preview. For more information on enabling SOQL Polymorphism for your organization, contact salesforce.com, inc.

Sample Code—Java

This sample upserts two accounts using a custom external ID field called `MyExtId__c`. The `upsert()` call matches the accounts based on the `MyExtId__c` field in order to determine whether to create or update the accounts. Before running this sample, change the `MyExtId__c` field name to an existing custom ID field name in your org.

```
public void upsertRecords() {
   SObject[] upserts = new Account[2];

   Account upsertAccount1 = new Account();
   upsertAccount1.setName("Begonia");
   upsertAccount1.setIndustry("Education");
   upsertAccount1.setMyExtId__c("1111111111");
   upserts[0] = upsertAccount1;

   Account upsertAccount2 = new Account();
   upsertAccount2 = new Account();
   upsertAccount2.setName("Bluebell");
   upsertAccount2.setIndustry("Technology");
   upsertAccount2.setMyExtId__c("2222222222");
   upserts[1] = upsertAccount2;

   try {
      // Invoke the upsert call and save the results.
      // Use External_Id custom field for matching records.
      UpsertResult[] upsertResults = connection.upsert(
                                  "MyExtId__c", upserts);
      for (UpsertResult result : upsertResults) {
         if (result.isSuccess()) {
            System.out.println("\nUpsert succeeded.");
            System.out.println((result.isCreated() ? "Insert" : "Update")
                  + " was performed.");
```

```
            System.out.println("Account ID: " + result.getId());
         } else {
            System.out.println("The Upsert failed because: "
                  + result.getErrors()[0].getMessage());
         }
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample upserts two accounts using a custom external ID field called `MyExtId__c`. The `upsert()` call matches the accounts based on the `MyExtId__c` field in order to determine whether to create or update the accounts. Before running this sample, change the `MyExtId__c` field name to an existing custom ID field name in your org.

```
public void upsertRecords()
{
   sObject[] upserts = new Account[2];

   Account upsertAccount1 = new Account();
   upsertAccount1.Name = "Begonia";
   upsertAccount1.Industry = "Education";
   upsertAccount1.MyExtId__c = "1111111111";
   upserts[0] = upsertAccount1;

   Account upsertAccount2 = new Account();
   upsertAccount2 = new Account();
   upsertAccount2.Name = "Bluebell";
   upsertAccount2.Industry = "Technology";
   upsertAccount2.MyExtId__c = "2222222222";
   upserts[1] = upsertAccount2;

   try
   {
      // Invoke the upsert call and save the results.
      // Use External_Id custom field for matching records.
      UpsertResult[] upsertResults =
            binding.upsert("MyExtId__c", upserts);
      foreach (UpsertResult result in upsertResults)
      {
         if (result.success)
         {
            Console.WriteLine("\nUpsert succeeded.");
            Console.WriteLine(
                  (result.created ? "Insert" : "Update") +
                  " was performed."
            );
            Console.WriteLine("Account ID: " + result.id);
         }
         else
         {
            Console.WriteLine("The Upsert failed because: " +
                  result.errors[0].message);
         }
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                              e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

Name	Type	Description
ExternalIDFieldName	string	Contains the name of the field on this object with the external ID field attribute for custom objects or the idLookup field property for standard objects. The idLookup field property is usually on a field that is the object's ID field or name field, but there are exceptions, so check for the presence of the property in the object you wish to upsert().
sObjects	sObject[]	Array of one or more records (maximum of 200) to create or update.

Response

UpsertResult[]

Faults

InvalidSObjectFault

UnexpectedErrorFault

See Also:

create()

update()

API Call Basics

UpsertResult

The upsert call returns an array of UpsertResult objects. Each element in the array corresponds to the sObject[] array passed as the sObjects parameter in the upsert() call. For example, the object returned in the first index in the UpsertResult array matches the object specified in the first index of the sObject[] array.

An UpsertResult object has the following properties:

Name	Type	Description
created	boolean	Indicates whether the record was created (true) or updated (false).
errors	Error[]	If errors occurred during the call, an array Error objects, providing the error code and description, is returned.
id	ID	If the call succeeded, the field contains the ID of the record that was either updated or created. If there was an error, the field is null. For more information, see ID Field Type.
success	boolean	Indicates whether the call succeeded (true) or not (false) for this object.

Chapter 10

Describe Calls

The following table lists supported describe calls in the API in alphabetical order, and provides a brief description for each. Click a call name to see syntax, usage, and more information for that call.

Note: For a list of API utility calls, see Utility Calls, and for a list of general calls (calls that query, retrieve, or modify data), see Core Calls.

Call	Description
`describeAllTabs()`	Returns information about all the tabs—including Flexible Page tabs—available to the logged-in user, regardless of whether the user has chosen to hide tabs in his own user interface via the All Tabs (+) tab customization feature.
`describeAppMenu()`	Retrieves metadata about items either in the Salesforce1 navigation menu or the Salesforce drop-down app menu.
`describeApprovalLayout()`	Retrieves metadata about approval layouts for the specified object type.
`describeAvailableQuickActions()`	In API version 28.0, describes details about actions available for a specified parent. In API version 29.0 and greater, describes details about actions available for a specified context.
`describeCompactLayouts()`	Retrieves metadata about compact layouts for the specified object type.
`describeDataCategoryGroups()`	Retrieves available category groups for entities specified in the request.
`describeDataCategoryGroupStructures()`	Retrieves available category groups along with their data category structure for entities specified in the request.
`describeFlexiPages()`	Retrieves metadata details about a set of Flexible Pages. A Flexible Page is the home page for a mobile app that appears as a menu item in the Salesforce1 navigation menu.
`describeGlobal()`	Retrieves a list of available objects for your organization's data.
`describeGlobalTheme()`	Returns information about both objects and themes available to the current logged-in user.
`describeLayout()`	Retrieves metadata about page layouts for the specified object type.
`describePrimaryCompactLayouts()`	Retrieves metadata about the primary compact layout for each of the specified object types.
`describeQuickActions()`	Retrieves details about specified actions.
`describeSearchScopeOrder()`	Retrieves an ordered list of objects in the logged-in user's default global search scope, including any pinned objects in the user's search results page.

Call	Description
describeSObject()	Retrieves metadata (field list and object properties) for the specified object type. Superseded by describeSObjects().
describeSObjects()	An array-based version of describeSObject.
describeSoftphoneLayout()	Describes the softPhone layout(s) created for an organization.
describeTabs()	Returns information about the standard and custom apps available to the logged-in user, as listed in the Force.com app menu at the top of the page.
describeTheme()	Returns information about themes available to the current logged-in user.

Samples

The samples in this section are based on the enterprise WSDL file. They assume that you have already imported the WSDL file and created a connection. To learn how to do so, see the Quick Start tutorial.

describeAllTabs()

Returns information about all the tabs—including Flexible Page tabs—available to the logged-in user, regardless of whether the user has chosen to hide tabs in his own user interface via the All Tabs (+) tab customization feature.

Syntax

```
DescribeTab [] = connection.describeAllTabs();
```

Usage

Use the `describeAllTabs()` call to obtain information about all the tabs that are available to the logged-in user.

Alternately, use `describeTabs()` if you want information only about the tabs that display in the Salesforce user interface for the logged-in user.

Sample Code—Java

This sample calls `describeAllTabs()`, which returns an array of DescribeTab results.

```
public void describeAllTabsSample() {
  try {
    // Describe tabs
    DescribeTab[] tabs = connection.describeAllTabs();
    System.out.println("There are " + tabs.length +
        " tabs available to you.");

      // Iterate through the returned tabs
      for (int j = 0; j < tabs.length; j++) {
        DescribeTab tab = tabs[j];
        System.out.println("\tTab " + (j + 1) + ":");
        System.out.println("\t\tName: " + tab.getName());
        System.out.println("\t\t\Associated SObject" + tab.getSobjectName());
        System.out.println("\t\tLabel: " + tab.getLabel());
        System.out.println("\t\tURL: " + tab.getUrl());
        DescribeColor[] tabColors = tab.getColors();
        // Iterate through tab colors as needed
        DescribeIcon[] tabIcons = tab.getIcons();
        // Iterate through tab icons as needed
      }
  } catch (ConnectionException ce) {
```

```
    ce.printStackTrace();
  }
}
```

Arguments

None.

Response

DescribeTab

describeAppMenu()

Retrieves metadata about items either in the Salesforce1 navigation menu or the Salesforce drop-down app menu. This call is available in API version 29.0 and later.

If you're accessing the API using a custom community URL, the `describeAppMenu()` call retrieves the tab set associated with the community.

Syntax

```
DescribeAppMenuResult describeResult = connection.describeAppMenu(String appMenuType);
```

Code Sample—Java

This code sample shows how to get the menu items from the Salesforce1 navigation menu.

```
public void describeAppMenu() {
    try {
        //The following two lines are equivalent
        DescribeAppMenuResult describe = connection.describeAppMenu("Salesforce1");
       DescribeAppMenuResult appMenu = getClient().describeAppMenu(AppMenuType.Salesforce1);


        for (DescribeAppMenuItem menuItem : appMenu.getAppMenuItems()) {

            if (menuItem.getType() == "Tab.apexPage") {

                String visualforceUrl = menuItem.getContent();

                System.out.println("URL to Visualforce page: " + visualforceUrl);

            }

        }

    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

Arguments

Name	Type	Description
appMenuType	string	Restricts the menu data returned to the specified menu type. Valid values are: • AppSwitcher—to retrieve the data from the Salesforce drop-down app menu • Salesforce1—to retrieve the data from the Salesforce1 navigation menu

Response

DescribeAppMenuResult

Faults

InvalidOrNullForRestrictedPicklist

DescribeAppMenuResult

The describeAppMenu() call returns a list of menu items contained in the specified menu type. The following types are available in API version 29.0 and later.

Name	Type	Description
appMenuItems	DescribeAppMenuItem[]	Array of one or more menu items in the selected menu type.

DescribeAppMenuItem

Each DescribeAppMenuItem object has these fields:

Name	Type	Description
colors	DescribeColor[]	Array of color information used for the tab associated with the menu item.
content	string	Information that helps build the menu item. Each menu item has a different type of content for this field. For example, the Salesforce1 app menu type could contain: • FlexiPage—the ID of the Flexible Page • Visualforce tab—the URL to the page, such as /apex/myApexPage. Menu items of types other than these don't use this field.
icons	DescribeIcon[]	Array of icon information used for the tab associated with the menu item.
label	string	The display label of the menu item.
name	string	API name of the menu item.
type	string	The type of menu item, and its subtype, if any. Possible values for the Salesforce1 menu type are: • Standard.Dashboards—Dashboards menu item • Standard.Feed—Chatter feed menu item

Name	Type	Description
		• Standard.Today—the Today menu item • Standard.Tasks—Tasks menu item • Tab.apexPage—a Visualforce tab menu item • Tab.flexipage—a Flexible Page tab menu item
url	string	The Salesforce URL the menu item should point to. For the Salesforce1 menu type, this field is `null` for the Dashboards, Feed, Today, Tasks, and Flexible Page menu items.

describeApprovalLayout()

Retrieves metadata about approval layouts for the specified object type.

Syntax

```
DescribeApprovalLayoutResult approvalLayoutResult = connection.describeApprovalLayout(string
 sObjectType, string[] approvalProcessNames);
```

Usage

Use this call to retrieve information about the approval layout for a given object type. Each approval process has one approval layout.

If you supply a null value for `approvalProcessNames`, all the approval layouts for the object are returned, instead of the approval layout of each specified approval process.

Sample Code—Java

This sample shows how to get the approval layouts of an Account sObject. It calls `describeApprovalLayout()` with the name of the sObject type to describe. After getting the approval layouts, the sample prints the name and fields found for each approval layout.

```
public void describeApprovalLayoutSample() {
  try {
    String objectToDescribe = "Account";
    DescribeApprovalLayoutResult approvalLayoutResult =
      connection.describeApprovalLayout(objectToDescribe, null);
    System.out.print("There are " + approvalLayoutResult.getApprovalLayouts().length);
    System.out.println(" approval layouts for the " + objectToDescribe + " object.");

    // Get all the approval layouts for the sObject
    for (int i = 0; i < approvalLayoutResult.getApprovalLayouts().length; i++) {
      DescribeApprovalLayout aLayout = approvalLayoutResult.getApprovalLayouts()[i];
      System.out.println(" There is an approval layout with name: " + aLayout.getName());
      DescribeLayoutItem[] layoutItems = aLayout.getLayoutItems();
      System.out.print(" There are " + layoutItems.length);
      System.out.println(" fields in this approval layout.");
      for (int j = 0; j < layoutItems.length; j++) {
        System.out.print("This approval layout has a field with name: ");
        System.out.println(layoutItems[j].getLabel());
      }
    }
  } catch (ConnectionException ce) {
    ce.printStackTrace();
```

```
   }
}
```

Arguments

Name	Type	Description
sObjectType	string	The specified value must be a valid object for your organization. If the object is a person account, specify Account, or if it is a person contact, specify Contact.
approvalProcessNames	string[]	Optional array of the approval process API names to return approval layout metadata for.

Response

DescribeApprovalLayoutResult

Faults

InvalidSObjectFault

UnexpectedErrorFault

DescribeApprovalLayoutResult

The describeApprovalLayout() call returns a DescribeApprovalLayoutResult object containing top-level record type information about the passed-in sObjectType. Your client application can traverse this object to retrieve detailed metadata about the approval layout.

Name	Type	Description
approvalLayouts	DescribeApprovalLayout[]	List of all the approval layouts in use by the object.

DescribeApprovalLayout

Represents an individual item in the DescribeApprovalLayout list.

Name	Type	Description
id	ID	Unique ID of this ApprovalLayout. For information on IDs, see ID Field Type.
label	string	Label of the approval layout.
layoutItems	DescribeLayoutItem[]	Array of one or more fields assigned to the approval layout.
name	string	API name of the approval layout.

describeAvailableQuickActions()

In API version 28.0, describes details about actions available for a specified parent. In API version 29.0 and greater, describes details about actions available for a specified context.

Note: In the application, QuickActions are referred to as actions or publisher actions.

Syntax

```
DescribeAvailableQuickActionResult [] = connection.describeAvailableQuickActions(string
parentOrContextType );
```

Usage

Use `describeAvailableQuickActions()` to get the list of actions whose parent (API version 28.0) or context (API version 29.0 and greater) entity name is supplied as well as standard and global actions. The `describeAvailableQuickActions()` call uses the parent entity name, such as "Account", or "null" for global actions, or in API version 29.0 and greater, the context, to return an array of DescribeAvailableQuickActionResult.

Sample—Java

This sample retrieves and displays the available action information for the Account object.

```
public void example() throws Exception {
    DescribeAvailableQuickActionResult[] aResult =
        conn.describeAvailableQuickActions("Account");
    for(DescribeAvailableQuickActionResult ar : aResult) {
        System.out.println("Action label: " + ar.getLabel());
        System.out.println("Action name: " + ar.getName());
        System.out.println("Action type: " + ar.getType());
    }
}
```

Arguments

Name	Type	Description
`parentOrContextType`	string	Either a standard or custom object. • The `parentType` applies only to API version 28.0. • The `contextType` applies to API version 29.0 and greater.

Response

An array of DescribeAvailableQuickActionResult objects.

Faults

`connection.exception` errors

DescribeAvailableQuickActionResult

The `describeAvailableQuickActions()` call returns an array of DescribeAvailableQuickActionResult objects. In API version 28.0, each DescribeAvailableQuickActionResult object represents details about actions available for a specified parent. In API version 29.0 and greater, each DescribeAvailableQuickActionResult object represents details about actions available for a specified context.

Note: In the application, QuickActions are referred to as actions or publisher actions.

Name	Type	Description
label	string	The action label.
name	string	The action name.
type	string	• LogACall • SocialPost • Canvas • Create • VisualforcePage • Update

describeCompactLayouts()

Retrieves metadata about compact layouts for the specified object type.

Syntax

```
DescribeCompactLayoutsResult compactLayoutResult = connection.describeCompactLayouts(string
 sObjectType, ID[] recordTypeId);
```

Usage

Use this call to retrieve information about the compact layout for a given object type. This call returns metadata about a given compact layout, including the record type mappings. For more information about compact layouts, see the Salesforce online help.

Sample Code—Java

This sample shows how to get the compact layouts of an Account sObject. It calls `describeCompactLayouts()` with the name of the sObject type to describe. After getting the compact layouts, the sample prints the images, fields, and action buttons found for each compact layout. Next, it prints the system default compact layout for the object, then the mapping information of record types to compact layouts.

```
public void testDescribeCompactLayoutsSample() {
    try {
        String objectToDescribe = "Account";
        DescribeCompactLayoutsResult compactLayoutResult = connection
                .describeCompactLayouts(objectToDescribe, null);
        System.out.println("There are " + compactLayoutResult.getCompactLayouts().length
                + " compact layouts for the " + objectToDescribe + " object.");

        // Get all the compact layouts for the sObject
        for (int i = 0; i < compactLayoutResult.getCompactLayouts().length; i++) {
            DescribeCompactLayout cLayout = compactLayoutResult.getCompactLayouts()[i];
          System.out.println(" There is a compact layout with name: " + cLayout.getName());

            DescribeLayoutItem[] fieldItems = cLayout.getFieldItems();
           System.out.println(" There are " + fieldItems.length + " fields in this compact
 layout.");
```

```
            // Write field items
            for (int j = 0; j < fieldItems.length; j++) {
                System.out.println(j + " This compact layout has a field with name: " +
fieldItems[j].getLabel());
            }

            DescribeLayoutItem[] imageItems = cLayout.getImageItems();
            System.out.println(" There are " + imageItems.length + " image fields in this
 compact layout.");

            // Write the image items
            for (int j = 0; j < imageItems.length; j++) {
                System.out.println(j + " This compact layout has an image field with name:
 " + imageItems[j].getLabel());
            }

            DescribeLayoutButton[] actions = cLayout.getActions();
            System.out.println(" There are " + actions.length + " buttons in this compact
 layout.");

            // Write the action buttons
            for (int j = 0; j < actions.length; j++) {
                System.out.println(j + " This compact layout has a button with name: " +
actions[j].getLabel());
            }

            System.out.println("This object's default compact layout is: "
                    + compactLayoutResult.getDefaultCompactLayoutId());

            RecordTypeCompactLayoutMapping[] mappings =
compactLayoutResult.getRecordTypeCompactLayoutMappings();
            System.out.println("There are " + mappings.length + " record type to compact
layout mapping for the "
                    + objectToDescribe + " object.");
            for (int j = 0; j < mappings.length; j++) {
                System.out.println(j + " Record type " + mappings[j].getRecordTypeId()
                        + " is mapped to compact layout " +
mappings[j].getCompactLayoutId());
            }
        }

    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }

}
```

Arguments

Name	Type	Description
sObjectType	string	The specified value must be a valid object for your organization. If the object is a person account, specify Account, or if it is a person contact, specify Contact.
recordTypeId	ID[]	Optional parameter that restricts the compact layout data returned to the specified record types.

Response

DescribeCompactLayoutsResult

Faults

InvalidSObjectFault

UnexpectedErrorFault

DescribeCompactLayoutsResult

The `describeCompactLayouts()` call returns a DescribeCompactLayoutsResult object containing top-level record type information about the passed-in `sObjectType`, as well as a mapping of record types to compact layouts. Your client application can traverse this object to retrieve detailed metadata about the compact layout.

Name	Type	Description
`compactLayouts`	DescribeCompactLayout[]	List of all the compact layouts in use by the object.
`defaultCompactLayoutId`	ID	ID of the primary compact layout assigned to the object.
`recordTypeCompactLayoutMappings`	RecordTypeCompactLayoutMapping[]	Record type mapping(s) for the object. The compact layouts associated with the object may be mapped to more than one record type.

DescribeCompactLayout

Represents an individual item in the DescribeCompactLayout list.

Name	Type	Description
`actions`	DescribeLayoutButton[]	Array of one or more DescribeLayoutButton items assigned to the compact layout. This list is set by salesforce.com and is read-only.
`fieldItems`	DescribeLayoutItem[]	Array of one or more fields assigned to the compact layout.
`id`	ID	Unique ID of this CompactLayout. For information on IDs, see ID Field Type.
`imageItems`	DescribeLayoutItem[]	Array of one or more images assigned to the compact layout. This list is set by salesforce.com and is read-only.
`label`	string	Label of the compact layout.
`name`	string	API name of the compact layout.
`objectType`	string	The name of the object to which the compact layout is assigned.

RecordTypeCompactLayoutMapping

Represents a single record type mapping in the `recordTypeCompactLayoutMappings` field in a DescribeCompactLayoutsResult object. This object is a map of valid `recordTypeId` to `compactLayoutId`.

Name	Type	Description
`available`	boolean	Indicates whether this record type is available (`true`) or not (`false`). Availability is used to display a list of available record types to the user when they are creating a new record.
`compactLayoutId`	ID	ID of the compact layout associated with this record type.

Name	Type	Description
compactLayoutName	string	API name of the compact layout.
recordTypeName	string	API name of the record type.
recordTypeId	ID	ID of the record type.

describeDataCategoryGroups()

Retrieves available category groups for objects specified in the request.

Syntax

```
DescribeDataCategoryGroupResult[] = connection.describeDataCategoryGroups()(string[]
sObjectTypes);
```

Usage

Use this call to describe the available category groups for the objects specified in the request. This call can be used with the describeDataCategoryGroupStructures() call to describe all the categories available for a specific object. For additional information about data categories, see “What are Data Categories?” in the Salesforce online help.

Sample Code—Java

This sample shows how to retrieve the data category groups associated with:

- Salesforce Knowledge articles
- Questions from the Answers feature

It returns the name, label and description of a category group and the name of the associated sobject (article or question). It also returns the number of data categories in the data category group.

```
public void describeDataCategoryGroupsSample() {
    try {
        // Make the describe call for data category groups
        DescribeDataCategoryGroupResult[] results =
            connection.describeDataCategoryGroups(new String[] {
                "KnowledgeArticleVersion", "Question"});

        // Get the properties of each data category group
        for (int i = 0; i < results.length; i++) {
          System.out.println("sObject: " +
              results[i].getSobject());
          System.out.println("Group name: " +
              results[i].getName());
          System.out.println("Group label: " +
              results[i].getLabel());
          System.out.println("Group description: " +
              (results[i].getDescription()==null? "" :
                  results[i].getDescription()));
          System.out.println("Number of categories: " +
              results[i].getCategoryCount());
        }
      } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

Sample Code—C#

This sample shows how to retrieve the data category groups associated with:

- Salesforce Knowledge articles
- Questions from the Answers feature

It returns the name, label and description of a category group and the name of the associated `sobject` (article or question). It also returns the number of data categories in the data category group.

```
public void describeDataCategoryGroups() {
    try {
        // Make the describe call for data category groups
        DescribeDataCategoryGroupResult[] results =
            binding.describeDataCategoryGroups(new String[] {
                "KnowledgeArticleVersion", "Question"});

        // Get the properties of each data category group
        for (int i = 0; i < results.Length; i++) {
            Console.WriteLine("sObject: " +
                results[i].sobject);
            Console.WriteLine("Group name: " +
                results[i].name);
            Console.WriteLine("Group label: " +
                results[i].label);
            Console.WriteLine("Group description: " +
                (results[i].description==null? "" :
                    results[i].description));
            Console.WriteLine("Number of categories: " +
                results[i].categoryCount);
        }
        } catch (SoapException e) {
            Console.WriteLine("An unexpected error has occurred: " +
                    e.Message + "\n" + e.StackTrace);
    }
}
```

Arguments

Name	Type	Description
`sObjectTypes`	string[]	The specified value can be: • `KnowledgeArticleVersion`—to retrieve category groups associated with article types. • `Question`—to retrieve category groups associated with questions. For additional information about articles and questions, see “Managing Articles and Translations” and “Answers Overview” in the Salesforce online help.

Response

`DescribeDataCategoryGroupResult`

Faults

`InvalidSObjectFault`

`UnexpectedErrorFault`

DescribeDataCategoryGroupResult

The `describeDataCategoryGroups()` call returns a `DescribeDataCategoryGroupResult` object containing the list of the category groups associated with the specified objects.

Name	Type	Description
categoryCount	int	The number of visible data categories in the data category group.
description	string	The description of the data category group.
label	string	Label for the data category group in the Salesforce user interface.
name	string	The unique name used for API access to the data category group .
sobject	string	The object associated with the data category group.

describeDataCategoryGroupStructures()

Retrieves available category groups along with their data category structure for objects specified in the request.

Syntax

```
DescribeDataCategoryGroupStructureResult[] = connection.
describeDataCategoryGroupStructures()(DataCategoryGroupSObjectTypePair[] pairs, boolean
topCategoriesOnly)
```

Usage

Use this call to return the visible data category structure for the given object category group pairs. First use `describeDataCategoryGroups()` to find the available category groups for the objects specified. From the returned list, choose the object category group pairs to pass as the input in `describeDataCategoryGroupStructures()`. This call returns all the visible categories and data category structure as output. For additional information about data categories and data category visibility, see “What are Data Categories?” and “About Category Group Visibility” in the Salesforce online help.

Sample Code—Java

This sample shows how to use sObject and data category group pairs to retrieve data categories for each pair. It calls `describeDataCategoryGroupStructures()` with two pairs, KnowledgeArticleVersion/Regions and Question/Regions, and iterates through the results of this call. It gets the top categories for each result, which is “All”, and then gets the first-level child categories. The sample requires that you set up a data category group called `Regions` with some child categories and associate it with a knowledge article and questions. Alternatively, you can replace the data category group name in the sample if you want to use an existing data category group in your org that has a different name.

```
public void describeDataCateogryGroupStructuresSample() {
    try {
        // Create the data category pairs
        DataCategoryGroupSobjectTypePair pair1 =
            new DataCategoryGroupSobjectTypePair();
        DataCategoryGroupSobjectTypePair pair2 =
            new DataCategoryGroupSobjectTypePair();
        pair1.setSobject("KnowledgeArticleVersion");
        pair1.setDataCategoryGroupName("Regions");
        pair2.setSobject("Question");
```

```
        pair2.setDataCategoryGroupName("Regions");

        DataCategoryGroupSobjectTypePair[] pairs =
            new DataCategoryGroupSobjectTypePair[] {
                pair1,
                pair2
            };

        // Get the list of top level categories using the describe call
        DescribeDataCategoryGroupStructureResult[] results =
            connection.describeDataCategoryGroupStructures(
                pairs,
                false
        );

        // Iterate through each result and get some properties
        // including top categories and child categories
        for (int i = 0; i < results.length; i++) {
          DescribeDataCategoryGroupStructureResult result =
              results[i];
          String sObject = result.getSobject();
          System.out.println("sObject: " + sObject);
          System.out.println("Group name: " + result.getName());
          System.out.println("Group label: " + result.getLabel());
          System.out.println("Group description: " +
              result.getDescription());

          // Get the top-level categories
          DataCategory[] topCategories = result.getTopCategories();

          // Iterate through the top level categories and retrieve
          // some information
          for (int j = 0; j < topCategories.length; j++) {
            DataCategory topCategory = topCategories[j];
            System.out.println("Category name: " +
                topCategory.getName());
            System.out.println("Category label: " +
                topCategory.getLabel());
            DataCategory [] childCategories =
                topCategory.getChildCategories();
            System.out.println("Child categories: ");
            for (int k = 0; k < childCategories.length; k++) {
              System.out.println("\t" + k + ". Category name: " +
                  childCategories[k].getName());
              System.out.println("\t" + k + ". Category label: " +
                  childCategories[k].getLabel());
            }
          }
        }
      } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

Sample Code—C#

This sample shows how to use sObject and data category group pairs to retrieve data categories for each pair. It calls `describeDataCategoryGroupStructures()` with two pairs, KnowledgeArticleVersion/Regions and Question/Regions, and iterates through the results of this call. It gets the top categories for each result, which is “All”, and then gets the first-level child categories. The sample requires that you set up a data category group called `Regions` with some child categories and associate it with a knowledge article and questions. Alternatively, you can replace the data category group name in the sample if you want to use an existing data category group in your org that has a different name.

```
public void describeDataCateogryGroupStructuresSample() {
    try {
        // Create the data category pairs
```

```
        DataCategoryGroupSobjectTypePair pair1 =
            new DataCategoryGroupSobjectTypePair();
        DataCategoryGroupSobjectTypePair pair2 =
            new DataCategoryGroupSobjectTypePair();
        pair1.sobject = "KnowledgeArticleVersion";
        //pair1.setDataCategoryGroupName("Regions");
        pair1.dataCategoryGroupName = "KBArticleCategories";
        pair2.sobject = "Question";
        //pair2.setDataCategoryGroupName("Regions");
        pair2.dataCategoryGroupName = "KBArticleCategories";

        DataCategoryGroupSobjectTypePair[] pairs =
            new DataCategoryGroupSobjectTypePair[] {
                pair1,
                pair2
            };

        // Get the list of top level categories using the describe call
        DescribeDataCategoryGroupStructureResult[] results =
            binding.describeDataCategoryGroupStructures(
                pairs,
                false
        );

        // Iterate through each result and get some properties
        // including top categories and child categories
        for (int i = 0; i < results.Length; i++) {
            DescribeDataCategoryGroupStructureResult result =
                results[i];
            String sObject = result.sobject;
            Console.WriteLine("sObject: " + sObject);
            Console.WriteLine("Group name: " + result.name);
            Console.WriteLine("Group label: " + result.label);
            Console.WriteLine("Group description: " +
                result.description);

            // Get the top-level categories
            DataCategory[] topCategories = result.topCategories;

            // Iterate through the top level categories and retrieve
            // some information
            for (int j = 0; j < topCategories.Length; j++) {
            DataCategory topCategory = topCategories[j];
            Console.WriteLine("Category name: " +
                topCategory.name);
            Console.WriteLine("Category label: " +
                topCategory.label);
            DataCategory [] childCategories =
                topCategory.childCategories;
            Console.WriteLine("Child categories: ");
            for (int k = 0; k < childCategories.Length; k++) {
                Console.WriteLine("\t" + k + ". Category name: " +
                    childCategories[k].name);
                Console.WriteLine("\t" + k + ". Category label: " +
                    childCategories[k].label);
            }
            }
        }
    }
    catch (SoapException e)
    {
        Console.WriteLine("An unexpected error has occurred: " +
            e.Message + "\n" + e.StackTrace);
    }
}
```

Arguments

Name	Type	Description
`pairs`	DataCategoryGroupSObjectTypePair[]	Specifies a category group and an object to query. Visible data categories are retrieved for that object.
`topCategoriesOnly`	boolean	Indicates whether the call returns only the top (`true`) or all the categories (`false`) visible depending on the user's data category group visibility settings. For more information on data category group visibility, see About Category Group Visibility in the Salesforce online help.

DataCategoryGroupSObjectTypePair contains the following fields:

Name	Type	Description
`dataCategoryGroupName`	string	The unique name used for API access to the data category group.
`sobject`	string	The object associated with the data category group

Response

`DescribeDataCategoryGroupStructureResult`[]

Faults

`InvalidSObjectFault`

`UnexpectedErrorFault`

DescribeDataCategoryGroupStructureResult

The `describeDataCategoryGroupStructures()` call returns an array of `DescribeDataCategoryGroupStructureResult` objects containing the category groups and categories associated with the specified objects.

Name	Type	Description
description	string	The description of the data category group.
label	string	The label for the data category group in the Salesforce user interface.
name	string	The unique name used for API access to the data category group.
subject	string	The object associated with the data category group.
topCategories	DataCategory[]	A list of top level categories visible depending on the user's data category group visibility settings. For more information on data category group visibility, see “About Category Group Visibility” in the Salesforce online help.

DataCategory

Name	Type	Description
childDataCategories	DataCategory[]	A recursive list of visible sub categories in the data category.
label	string	The label for the data category in the Salesforce user interface.

Name	Type	Description
name	string	The unique name used for API access to the data category.

describeFlexiPages()

Retrieves metadata details about a set of Flexible Pages. A Flexible Page is the home page for a mobile app that appears as a menu item in the Salesforce1 navigation menu.

Syntax

```
DescribeFlexiPageResult[] = connection.describeFlexiPages(string flexiPageNames[]);
```

Usage

Use this call to retrieve information about the specified Flexible Pages. This call returns metadata about a set of Flexible Pages, such as each Flexible Page's layout and associated QuickActions.

If you supply an empty list for `flexiPageNames`, no Flexible Pages are returned.

Note:

This call is an advanced API call that is typically used by partners who have written custom page rendering code for generating output on a mobile device and need to examine the layout details of a Flexible Page before rendering the output.

Sample Code—Java

This sample shows how to get a Flexible Page. It calls `describeFlexiPage()` with the name of a Flexible Page to describe. After getting the Flexible Page, it prints out the regions, components and properties.

```
public void describeFlexiPageSample() {
  try {
        // Retrieve a single FlexiPage
        String flexiPageName = "MyFlexiPage";
        DescribeFlexiPageResult[] result = null;
        result = connection.describeFlexiPages(new String[]{flexiPageName});
        String msg = String.format("There are %s FlexiPages described in the response",
result.length);
        System.out.println(msg);
        DescribeFlexiPageResult page = result[0];

        // Iterate over the regions of the FlexiPage
        for (DescribeFlexiPageRegion region : page.getRegions()) {
            msg = String.format("Region: %s", region.getName());

            // Iterate over the components in each region
            for (DescribeComponentInstance cmp : region.getComponents()) {
              String fullComponentName = cmp.getTypeNamespace() + ":" + cmp.getTypeName();

                System.out.println("Component: " + fullComponentName);

                // Iterate over the properties of each component
                for (DescribeComponentInstanceProperty prop : cmp.getProperties()) {
                    msg = String.format("Property [%s] has value [%s]", prop.getName(),
prop.getValue());
                    System.out.println(msg);
                }
            }
```

```
        }
    } catch (ConnectionException ce) {
      ce.printStackTrace();
    }
}
```

Faults

UnexpectedErrorFault

Arguments

Name	Type	Description
flexiPages	string[]	An array of Flexible Pages to be retrieved.

Response

DescribeFlexiPageResult

DescribeFlexiPageResult

The `describeFlexiPages()` call returns an array of `DescribeFlexiPageResult` objects containing information about the passed in `flexiPages`, including lists of regions and actions associated with each Flexible Page.

Name	Type	Description
id	ID	The ID of the Flexible Page.
label	string	The label of the Flexible Page.
name	string	The unique name of the Flexible Page.
quickActionList	DescribeQuickActionListResult	List of actions associated with the specified Flexible Page.
regions	DescribeFlexiPageRegion[]	List of regions on the specified Flexible Page.

DescribeFlexiPageRegion

Represents a region of a FlexiPage. There is one region per FlexiPage. A region can contain a record list component or a recent items component that can be scoped to a set of entities.

Name	Type	Description
components	DescribeComponentInstance[]	List of components on the specified Flexible Page.
name	string	Unique name of the FlexiPage region.

DescribeComponentInstance

Represents an instance of a component in a DescribeFlexiPageRegion, such as a filter list.

Name	Type	Description
properties	DescribeComponentInstanceProperty[]	Properties of the component instance.

Name	Type	Description
typeName	string	The first piece of the name of the component. For example, the “flexipage” in `flexipage:filterListCard`.
typeNamespace	string	The second piece of the name of the component. For example, the “filterListCard” in `flexipage:filterListCard`.

DescribeComponentInstanceProperty

Represents a single property in a DescribeComponentInstance.

Name	Type	Description
name	string	Name of the component instance property. For example, the `filterListCard` component has a property whose name is "entityNames".
value	string	Value of the component instance property. For example, the `filterListCard` component has an `entityNames` property whose value is a comma separated list of entity names like "Account, Contact, Opportunity"

describeGlobal()

Retrieves a list of available objects for your organization’s data.

Syntax

```
DescribeGlobalResult = connection.describeGlobal();
```

Usage

Use `describeGlobal()` to obtain a list of available objects for your organization. You can then iterate through this list and use `describeSObjects()` to obtain metadata about individual objects.

Your client application must be logged in with sufficient access rights to retrieve metadata about your organization’s data. For more information, see Factors that Affect Data Access.

Sample Code—Java

This sample shows how to perform a global describe. It then retrieves the sObjects from the global describe result and writes their names to the console.

```
public void describeGlobalSample() {
    try {
        // Make the describeGlobal() call
        DescribeGlobalResult describeGlobalResult =
            connection.describeGlobal();

        // Get the sObjects from the describe global result
        DescribeGlobalSObjectResult[] sobjectResults =
            describeGlobalResult.getSobjects();
```

```
        // Write the name of each sObject to the console
        for (int i = 0; i < sobjectResults.length; i++) {
          System.out.println(sobjectResults[i].getName());
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

Sample Code—C#

This sample shows how to perform a global describe. It then retrieves the sObjects from the global describe result and writes their names to the console.

```
public void describeGlobalSample()
{
    try
    {
        // Make the describeGlobal() call
        DescribeGlobalResult dgr = binding.describeGlobal();

        // Get the sObjects from the describe global result
        DescribeGlobalSObjectResult[] sObjResults = dgr.sobjects;

        // Write the name of each sObject to the console
        for (int i = 0; i < sObjResults.Length; i++)
        {
            Console.WriteLine(sObjResults[i].name);
        }
    }
    catch (SoapException e)
    {
        Console.WriteLine("An unexpected error has occurred: " +
            e.Message + "\n" + e.StackTrace);
    }
}
```

Arguments

None.

Response

DescribeGlobalResult

Fault

`UnexpectedErrorFault`

See Also:

describeSObjects()
API Call Basics
Using the Partner WSDL
https://developer.salesforce.com/page/Sample_SOAP_Messages

DescribeGlobalResult

The `describeGlobal()` call returns a `DescribeGlobalResult` object, which has the following properties.

Name	Type	Description
encoding	string	Specifies how an organization's data is encoded, such as UTF-8 or ISO-8859-1.
maxBatchSize	int	Maximum number of records allowed in a create(), update(), or delete() call.
sobjects	DescribeGlobalSObjectResult[]	List of result objects that returns information about the available objects for your organization. Available in API version 17.0 and later. This property enhances the information that was previously available in the types property.
types	string[]	List of available objects for your organization. You iterate through this list to retrieve the object string that you pass to describeSObjects(). Beginning with API version 17.0, this property is no longer supported. Use the name property in DescribeGlobalSObjectResult instead.

DescribeGlobalSObjectResult

Represents the properties for one of the objects available for your organization. Each object has the following properties:

Name	Type	Description
activateable	boolean	Reserved for future use.
createable	boolean	Indicates whether the object can be created via the create() call (true) or not (false).
custom	boolean	Indicates whether the object is a custom object (true) or not (false).
customSetting	boolean	Indicates whether the object is a custom setting object (true) or not (false).
deletable	boolean	Indicates whether the object can be deleted via the delete() call (true) or not (false).
deprecatedAndHidden	boolean	Reserved for future use.
feedEnabled	boolean	Indicates whether Chatter feeds are enabled for the object (true) or not (false). This property is available in API version 19.0 and later.
keyPrefix	string	Three-character prefix code in the object ID. Object IDs are prefixed with three-character codes that specify the type of the object. For example, Account objects have a prefix of 001 and Opportunity objects have a prefix of 006. Note that a key prefix can sometimes be shared by multiple objects so it does not always uniquely identify an object. Use the value of this field to determine the object type of a parent in those cases where the child may have more than one object type as parent (polymorphic). For example, you may need to obtain the keyPrefix value for the parent of a Task or Event.
label	string	Label text for a tab or field renamed in the user interface, if applicable, or the object name, if not. For example, an organization representing a medical vertical might rename Account to Patient. Tabs and fields can be

Name	Type	Description
		renamed in the Salesforce user interface. See the Salesforce online help for more information.
labelPlural	string	Label text for an object that represents the plural version of an object name, for example, “Accounts.”
layoutable	boolean	Indicates whether the object supports the describeLayout() call (true) or not (false).
mergeable	boolean	Indicates whether the object can be merged with other objects of its type (true) or not (false). true for leads, contacts, and accounts.
name	string	Name of the object. This name is equivalent to an entry in the types list that is no longer supported, beginning with API version 17.0.
queryable	boolean	Indicates whether the object can be queried via the query() call (true) or not (false).
replicateable	boolean	Indicates whether the object can be replicated via the getUpdated() and getDeleted() calls (true) or not (false).
retrieveable	boolean	Indicates whether the object can be retrieved via the retrieve() call (true) or not (false).
searchable	boolean	Indicates whether the object can be searched via the search() call (true) or not (false).
triggerable	boolean	Indicates whether the object supports Apex triggers.
undeletable	boolean	Indicates whether an object can be undeleted using the undelete() call (true) or not (false).
updateable	boolean	Indicates whether the object can be updated via the update() call (true) or not (false).

describeGlobalTheme()

Returns information about both objects and themes available to the current logged-in user.

Syntax

```
DescribeGlobalTheme = connection.describeGlobalTheme();
```

Usage

Use describeGlobalTheme() to get both a list of available objects and theme information about those objects for your organization. describeGlobalTheme() is a combination of describeGlobal() and describeTheme() combined into a single call.

Your client application must be logged in with sufficient access rights to retrieve theme and object information about your organization’s data. For more information, see Factors that Affect Data Access.

describeGlobalTheme() is available in API version 29.0 and later.

Sample

This Java sample calls `describeGlobalTheme()` and then iterates over the retrieved object and theme information.

```
public static void describeGlobalThemeExample() {
    try {
        // Get current theme and object information
        DescribeGlobalTheme globalThemeResult = connection.describeGlobalTheme();
        DescribeGlobalResult globalResult = globalThemeResult.getGlobal();
        DescribeThemeResult globalTheme = globalThemeResult.getTheme();
        // For the themes, get the array of theme items, one per object
        DescribeThemeItem[] themeItems = globalTheme.getThemeItems();
        for (int i = 0; i < themeItems.length; i++) {
            DescribeThemeItem themeItem = themeItems[i];
            System.out.println("Theme information for object " + themeItem.getName());
            // Get color and icon info for each themeItem
            DescribeColor colors[] = themeItem.getColors();
            System.out.println("    Number of colors: " + colors.length);
            int k;
            for (k = 0; k < colors.length; k++) {
                DescribeColor color = colors[k];
                System.out.println("    For Color #" + k + ":");
                System.out.println("      Web RGB Color: " + color.getColor());
                System.out.println("      Context: " + color.getContext());
                System.out.println("      Theme: " + color.getTheme());
            }
            DescribeIcon icons[] = themeItem.getIcons();
            System.out.println("    Number of icons: " + icons.length);
            for (k = 0; k < icons.length; k++) {
                DescribeIcon icon = icons[k];
                System.out.println("    For Icon #" + k + ":");
                System.out.println("      ContentType: " + icon.getContentType());
                System.out.println("      Height: " + icon.getHeight());
                System.out.println("      Theme: " + icon.getTheme());
                System.out.println("      URL: " + icon.getUrl());
                System.out.println("      Width: " + icon.getWidth());
            }
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
    }
}
```

Response

DescribeGlobalTheme

Faults

`UnexpectedErrorFault`

See Also:

- *DescribeGlobalTheme*
- *DescribeThemeResult*
- *DescribeThemeItem*
- *DescribeColor*
- *DescribeIcon*

DescribeGlobalTheme

The `describeGlobalTheme()` call returns DescribeGlobalTheme, which contains a DescribeThemeResult and a DescribeGlobalResult.

Name	Type	Description
`global`	DescribeGlobalResult	Object information.
`theme`	DescribeThemeResult	Theme information.

`describeLayout()`

Retrieves metadata about page layouts for the specified object type.

Syntax

```
DescribeLayoutResult = connection.describeLayout(string sObjectType, string layoutName, ID
 recordTypeID[]);
```

Usage

Use this call to retrieve information about the layout (presentation of data to users) for a given object type. This call returns metadata about a given page layout, such as the detail page layout, the edit page layout, and the record type mappings. For additional information about page layouts, see "Customizing Page Layouts" in the Salesforce online help.

Generally, user profiles have one layout associated with each object. In Enterprise, Unlimited, and Performance Editions, user profiles can have multiple layouts per object, where each layout is specific to a given record type. This call returns metadata for multiple layouts, if applicable.

Layouts may be further customized in standard objects that have defined named layouts, which are separate from the primary layout for both the profile and the record type. One example of named layouts is the UserAlt layout defined on the User object, which is consumed in Salesforce1 instead of the primary User layout. New layout names may only be defined by Salesforce.com, but customization of named layouts is controlled by administrators in the same way as primary layouts.

If you supply a null value for `recordTypeIds`, all the layouts for that user are returned, instead of just the layouts for each specified record type. The same layout may be associated with multiple record types for the user's profile, in which case there would only be one layout returned.

Note: This call is an advanced API call that is typically used only by partners who have written custom page rendering code for generating output on a specialized device (for example, on PDAs) and need to examine the layout details of an object before rendering the page output.

Use the following procedure to describe layouts:

1. To display a detail page or edit page for a record that already exists, a client application first gets the `recordTypeIds` from the record, then it finds the layoutId associated with that `recordTypeIds` (through recordTypeMapping), and finally it uses that layout information to render the page.
2. To display the create version of an edit page, a client application first determines whether more than one record type is available and, if so, presents the user with a choice. Once a record type has been chosen, then the client application uses the layout information to render the page. It uses the picklist values from the RecordTypeMapping to display valid picklist values for picklist fields.
3. A client application can access the labels for the layout, using the DescribeLayoutResult.

The following restrictions apply to person account record types:

- `describeLayout()` for version 7.0 and below will return the default business account record type as the default record type even if the tab default is a person account record type. In version 8.0 and after, it will always be the tab default.
- `describeLayout()` for version 7.0 and below will not return any person account record types.

For more information about person account record types, see Person Account Record Types.

Sample Code—Java

This sample shows how to get the layouts of an Account sObject. It calls `describeLayout()` with the name of the sObject type to describe. It doesn't specify record type IDs as a third argument, which means that layouts for all record types will be returned if record types are defined in your org for the specified sObject. After getting the layout, the sample writes the number of detail and edit sections found and their headings. Next, it iterates through each edit layout section and retrieves its components.

```
public void describeLayoutSample(){
  try {
      String objectToDescribe = "Account";
      DescribeLayoutResult dlr =
          connection.describeLayout(objectToDescribe, null, null);
      System.out.println("There are " + dlr.getLayouts().length +
          " layouts for the " + objectToDescribe + " object."
      );

      // Get all the layouts for the sObject
      for(int i = 0; i < dlr.getLayouts().length; i++) {
        DescribeLayout layout = dlr.getLayouts()[i];
        DescribeLayoutSection[] detailLayoutSectionList =
          layout.getDetailLayoutSections();
        System.out.println(" There are " +
          detailLayoutSectionList.length +
            " detail layout sections");
        DescribeLayoutSection[] editLayoutSectionList =
          layout.getEditLayoutSections();
        System.out.println(" There are " +
          editLayoutSectionList.length +
          " edit layout sections");

        // Write the headings of the detail layout sections
        for(int j = 0; j < detailLayoutSectionList.length; j++) {
          System.out.println(j +
              " This detail layout section has a heading of " +
              detailLayoutSectionList[j].getHeading());
        }

        // Write the headings of the edit layout sections
        for(int x = 0; x < editLayoutSectionList.length; x++) {
          System.out.println(x +
              " This edit layout section has a heading of " +
              editLayoutSectionList[x].getHeading());
        }

        // For each edit layout section, get its details.
        for(int k = 0; k < editLayoutSectionList.length; k++) {
          DescribeLayoutSection els =
            editLayoutSectionList[k];
          System.out.println("Edit layout section heading: " +
              els.getHeading());
          DescribeLayoutRow[] dlrList = els.getLayoutRows();
            System.out.println("This edit layout section has " +
                dlrList.length + " layout rows.");
            for(int m = 0; m < dlrList.length; m++) {
              DescribeLayoutRow lr = dlrList[m];
              System.out.println(" This row has " +
                  lr.getNumItems() + " layout items.");
              DescribeLayoutItem[] dliList = lr.getLayoutItems();
              for(int n = 0; n < dliList.length; n++) {
                DescribeLayoutItem li = dliList[n];
```

```
                if ((li.getLayoutComponents() != null) &&
                  (li.getLayoutComponents().length > 0)) {
                  System.out.println("\tLayout item " + n +
                      ", layout component: " +
                      li.getLayoutComponents()[0].getValue());
                }
                else {
                System.out.println("\tLayout item " + n +
                      ", no layout component");
                }
              }
            }
          }
        }

        // Get record type mappings
        if (dlr.getRecordTypeMappings() != null) {
          System.out.println("There are " +
              dlr.getRecordTypeMappings().length +
              " record type mappings for the " +
              objectToDescribe + " object"
          );
        } else {
          System.out.println(
              "There are no record type mappings for the " +
              objectToDescribe + " object."
          );
        }
      } catch (ConnectionException ce) {
        ce.printStackTrace();
      }
}
```

Sample Code—C#

This sample shows how to get the layouts of an Account sObject. It calls `describeLayout()` with the name of the sObject type to describe. It doesn't specify record type IDs as a third argument, which means that layouts for all record types will be returned if record types are defined in your org for the specified sObject. After getting the layout, the sample writes the number of detail and edit sections found and their headings. Next, it iterates through each edit layout section and retrieves its components.

```
public void describeLayoutSample()
{
    try
    {
        String objectToDescribe = "Account";
        DescribeLayoutResult dlr =
            binding.describeLayout(objectToDescribe, null, null);
        Console.WriteLine("There are " + dlr.layouts.Length +
            " layouts for the " + objectToDescribe + " object."
        );

        // Get all the layouts for the sObject
        for (int i = 0; i < dlr.layouts.Length; i++)
        {
            DescribeLayout layout = dlr.layouts[i];
            DescribeLayoutSection[] detailLayoutSectionList =
                layout.detailLayoutSections;
            Console.WriteLine(" There are " +
                detailLayoutSectionList.Length +
                " detail layout sections");
            DescribeLayoutSection[] editLayoutSectionList =
                layout.editLayoutSections;
            Console.WriteLine(" There are " +
                editLayoutSectionList.Length +
                " edit layout sections");

            // Write the headings of the detail layout sections
```

```
            for (int j = 0; j < detailLayoutSectionList.Length; j++)
            {
                Console.WriteLine(j +
                    " This detail layout section has a heading of " +
                    detailLayoutSectionList[j].heading);
            }

            // Write the headings of the edit layout sections
            for (int x = 0; x < editLayoutSectionList.Length; x++)
            {
                Console.WriteLine(x +
                    " This edit layout section has a heading of " +
                    editLayoutSectionList[x].heading);
            }

            // For each edit layout, get its details.
            for (int k = 0; k < editLayoutSectionList.Length; k++)
            {
                DescribeLayoutSection els =
                    editLayoutSectionList[k];
                Console.WriteLine("Edit layout section heading: " +
                        els.heading);
                DescribeLayoutRow[] dlrList = els.layoutRows;
                Console.WriteLine("This edit layout section has " +
                        dlrList.Length + " layout rows.");
                for (int m = 0; m < dlrList.Length; m++)
                {
                    DescribeLayoutRow lr = dlrList[m];
                    Console.WriteLine(" This row has " +
                        lr.numItems + " layout items.");
                    DescribeLayoutItem[] dliList = lr.layoutItems;
                    for (int n = 0; n < dliList.Length; n++)
                    {
                        DescribeLayoutItem li = dliList[n];
                        if ((li.layoutComponents != null) &&
                                (li.layoutComponents.Length > 0))
                        {
                            Console.WriteLine("\tLayout item " + n +
                                    ", layout component: " +
                                li.layoutComponents[0].value);
                        }
                        else
                        {
                            Console.WriteLine("\tLayout item " + n +
                                    ", no layout component");
                        }
                    }
                }
            }

            // Get record type mappings
            if (dlr.recordTypeMappings != null)
            {
                Console.WriteLine("There are " +
                    dlr.recordTypeMappings.Length +
                    " record type mappings for the " +
                    objectToDescribe + " object");
            }
            else
            {
                Console.WriteLine(
                    "There are no record type mappings for the " +
                    objectToDescribe + " object.");
            }
        }
    }
    catch (SoapException e)
    {
        Console.WriteLine("An unexpected error has occurred: " +
```

```
            e.Message + "\n" + e.StackTrace);
    }
}
```

Arguments

Name	Type	Description
sObjectType	string	The specified value must be a valid object for your organization. If the object is a person account, specify Account, or if it is a person contact, specify Contact.
layoutName	string	The specified value must be a valid named layout for this object. Layout names are obtained from namedLayoutInfos in DescribeSObjectResult. The entity name is not valid because the primary layout is not considered "named."
recordTypeIds	ID[]	Optional parameter restricts the layout data returned to the specified record types. To retrieve the layout for the master record type, specify the value 012000000000000AAA for the recordTypeIds regardless of the object. This value is returned in the recordTypeInfos for the master record type in the DescribeSObjectResult. Note that a SOQL query returns a null value, not 012000000000000AAA. For information on IDs, see ID Field Type.

Response

DescribeLayoutResult

Faults

InvalidSObjectFault

UnexpectedErrorFault

See Also:

API Call Basics
https://developer.salesforce.com/page/Sample_SOAP_Messages

DescribeLayoutResult

The describeLayout() call returns a DescribeLayoutResult object containing top-level record type information about the passed-in sObjectType, as well as a mapping of record types to layouts. Your client application can traverse this object to retrieve detailed metadata about the layout.

Tip: If you have Publisher Actions enabled in your organization, you can retrieve the layout definition for a global publisher layout by using Global as the sObjectType and null as the recordTypeId.

Name	Type	Description
layouts	DescribeLayout[]	Layout(s) associated with the specified sObjectType. In general, there is a one-to-one correspondence between layouts and objects. However, in some cases,

Name	Type	Description
		an object will have multiple layouts in the context of a given user profile.
recordTypeMappings	RecordTypeMapping[]	Record type mapping(s) available for the user. The objects on a user profile can have multiple record types. All record types are returned, not just those available to the calling user. This allows the client application to display a layout appropriate for a given user profile. For example, suppose User A owns a record, and this record has record type X set. If User B tries to view this record, then the client application can display the record using the layout associated with this record type for User B's profile (even if the record type is not a available for the user).
recordTypeSelectorRequired	boolean	If `true`, a record type selector page is required, if `false`, use the default record type.

DescribeLayout

Represents a specific layout for the specified `sObjectType`. Each `DescribeLayout` is referenced by its unique layout ID and consists of two types of views (represented in this object as arrays of DescribeLayoutSection):

- **Detail view**—Read-only display of the object. In a detail layout, certain pieces of information (such as address details) might be aggregated into a single DescribeLayoutItem.
- **Edit view**—Editable display of the object. In an edit layout, individual pieces of information (such as an address) will be broken up into separate fields.

An individual DescribeLayout consists of these fields:

Name	Type	Description
buttonLayoutSection	DescribeLayoutButtonSection	Standard and custom button sections associated with the specified layout.
detailLayoutSections	DescribeLayoutSection[]	Layout section(s) for the detail view.
editLayoutSections	DescribeLayoutSection[]	Layout section(s) for the edit view.
highlightsPanelLayoutSection	DescribeLayoutSection[]	Layout section(s) for the highlights panel view.
id	ID	Unique ID of this layout. For information on IDs, see ID Field Type.
quickActionList	DescribeQuickActionListResult	List of actions associated with the specified layout. This field is available in API version 28.0 and later.
relatedContent	RelatedContent	Mobile Cards section associated with the specified layout. This field is available in API version 29.0 and later.
relatedLists	RelatedList[]	Related list(s) associated with the specified layout.

DescribeLayoutButtonSection

Represents one of two sections of the layout containing either standard or custom buttons.

Name	Type	Description
detailButtons	DescribeLayoutButton[]	Standard or custom button(s) associated with the specified button section.

DescribeLayoutButton

Represents a single standard button, custom button, or custom link in a DescribeLayout.

Name	Type	Description
behavior	WebLinkWindowType	What the button or link does when clicked, such as execute JavaScript or open its content source in a new window, for example. This field is available in API version 31.0 and later.
content	string	The API name of the Visualforce page or s-control being delivered. This field is available in API version 31.0 and later.
contentSource	WebLinkType	The content source of the custom button or link. The `contentSource` for a standard button which hasn't been overridden is `null`. This field is available in API version 31.0 and later.
custom	boolean	Required. Indicates whether this is a custom button or link (`true`) or not (`false`).
encoding	string	Type of encoding assigned to the URL called by the button or link. Valid values are: • `UTF-8`—Unicode (UTF-8) • `ISO-8859-1`—General US & Western Europe (ISO-8859–1, ISO-LATIN-1) • `Shift_JIS`—Japanese (Shift-JIS) • `ISO-2022-JP`—Japanese (JIS) • `EUC-JP`—Japanese (EUC-JP) • `x-SJIS_0213`—Japanese (Shift-JIS_2004) • `ks_c_5601-1987`—Korean (ks_c_5601-1987) • `Big5`—Traditional Chinese (Big5) • `GB2312`—Simplified Chinese (GB2312) • `Big5-HKSCS`—Traditional Chinese Hong Kong (Big5–HKSCS) This field is available in API version 31.0 and later.
height	int	The height (in pixels) when a button or link's `behavior` field value is set to `newWindow`, `sidebar`, or `noSidebar`. This field is available in API version 31.0 and later.

Name	Type	Description
icons	DescribeIcon[]	Array of icons for this button or link. Each icon is associated with a theme. This field is available in API version 29.0 and later.
label	string	Label for the button or link displayed in the Salesforce user interface.
menubar	boolean	Indicates whether the menu bar displays (`true`) or not (`false`) when a button or link's `behavior` field value is set to `newWindow`. This field is available in API version 31.0 and later.
name	string	API name of the button or link.
overridden	boolean	Required. Indicates whether a standard button has been overridden (`true`) or not (`false`). This field is available in API version 31.0 and later.
resizeable	boolean	Indicates whether the new window is resizeable (`true`) or not (`false`) when a button or link's `behavior` field value is set to `newWindow`. This field is available in API version 31.0 and later.
scrollbars	boolean	Indicates whether scrollbars display (`true`) or not (`false`) when a button or link's `behavior` field value is set to `newWindow`. This field is available in API version 31.0 and later.
showsLocation	boolean	Indicates whether the address bar displays (`true`) or not (`false`) when a button or link's `behavior` field value is set to `newWindow`. This field is available in API version 31.0 and later.
showsStatus	boolean	Indicates whether the status bar displays (`true`) or not (`false`) when a button or link's `behavior` field value is set to `newWindow`. This field is available in API version 31.0 and later.
toolbar	boolean	Indicates whether the toolbars display (`true`) or not (`false`) when a button or link's `behavior` field value is set to `newWindow`. This field is available in API version 31.0 and later.
url	string	The URL called by the button or link. This field is available in API version 31.0 and later.
width	int	The width (in pixels) when a button or link's `behavior` field value is set to `newWindow`. This field is available in API version 31.0 and later.

Name	Type	Description
windowPosition	WebLinkPosition	Indicates the window position when a button or link's `behavior` field value is set to `newWindow`. This field is available in API version 31.0 and later.

DescribeLayoutComponent

Represents the smallest unit in a layout—a field or a separator. To reference a field for display, a client application uses the following notation to reference a field in the `describeSObjects()` call: `LayoutComponent.`*`fieldName`*.

In API version 31.0 and later, DescribeLayoutComponent is extended with FieldLayoutComponent if both the LayoutComponentType value is `Field`, and the field being described is either the compound field `Address` or the compound field `Person Name`.

Name	Type	Description
displayLines	int	The number of vertical lines displayed for a field in the edit view. Applies to `textarea` and multi-select picklist fields.
tabOrder	int	Indicates the tab order for the item in the row.
type	LayoutComponentType	The LayoutComponentType for this LayoutComponent.
value	string	Value of this LayoutComponent. The name of the field if the LayoutComponentType value is `Field`. The API name of the canvas app if the LayoutComponentType value is `Canvas`.

DescribeLayoutItem

Represents an individual item in a DescribeLayoutRow. A DescribeLayoutItem consists of a set of components (DescribeLayoutComponent), each of which is either a field or a separator. For most fields on a layout, there is only one component per layout item. However, in a display-only view, the DescribeLayoutItem might be a composite of the individual fields (for example, an address can consist of street, city, state, country, and postal code data). On the corresponding edit view, each component of the address field would be split up into separate DescribeLayoutItems.

Name	Type	Description
editable	boolean	Indicates whether this DescribeLayoutItem can be edited (`true`) or not (`false`). This field is available in API version 30.0 and below. It was replaced by the `editableForNew` and `editableForUpdate` fields in API version 31.0.
editableForNew	boolean	Indicates whether a new DescribeLayoutItem can be edited when creating a new record (`true`) or not (`false`). This field is available in API version 31.0 and later.
editableForUpdate	boolean	Indicates whether an existing DescribeLayoutItem can be edited when editing a record (`true`) or not (`false`). This field is available in API version 31.0 and later.
label	string	Label text for this DescribeLayoutItem.

Name	Type	Description
`layoutComponents`	DescribeLayoutComponent[]	DescribeLayoutComponent for this DescribeLayoutItem.
`placeholder`	boolean	Indicates whether this DescribeLayoutItem is a placeholder (`true`) or not (`false`). If `true`, then this DescribeLayoutItem is blank.
`required`	boolean	Indicates whether this DescribeLayoutItem is required (`true`) or not (`false`). This is useful to know if, for example, you wanted to render required fields in a contrasting color (such as red).

DescribeLayoutRow

Represents a row in a DescribeLayoutSection. A DescribeLayoutRow consists of one or more DescribeLayoutItem objects. For each DescribeLayoutRow, a DescribeLayoutItem refers either to a specific field or to an "empty" DescribeLayoutItem (a DescribeLayoutItem that contains no DescribeLayoutComponent objects). An empty DescribeLayoutItem can be returned when a given DescribeLayoutRow is sparse (for example, containing more fields on the right column than on the left column). Where there are gaps in the layout, an empty DescribeLayoutItem is returned as a placeholder.

Name	Type	Description
`layoutItems`	DescribeLayoutItem[]	Refers to either a specific field or to an empty LayoutItem (a LayoutItem that contains no DescribeLayoutComponent objects).
`numItems`	int	Number of `layoutItems`. This information is redundant but, due to a bug in a popular SOAP toolkit, was required to avoid serialization problems.

DescribeLayoutSection

Represents a section of a DescribeLayout and consists of one or more columns and one or more rows (an array of DescribeLayoutRow).

Name	Type	Description
`columns`	int	Number of columns in this DescribeLayoutSection.
`heading`	string	Heading text (label) for this DescribeLayoutSection.
`layoutRows`	DescribeLayoutRow[]	Array of one or more DescribeLayoutRow objects.
`rows`	int	Number of rows in this DescribeLayoutSection.
`tabOrder`	string	Indicates the tab order for the fields in the section in the edit view. Valid values are: • LeftToRight • TopToBottom This field is available in API version 31.0 and later.
`useCollapsibleSection`	boolean	Indicates whether this DescribeLayoutSection is a collapsible section, also known as a "twistie" (`true`), or not (`false`).

Name	Type	Description
useHeading	boolean	Indicates whether to display the heading (true) or not (false).

DescribeQuickActionListResult

Represents a list of actions assigned to the page layout. Available in API version 28.0 and later.

Name	Type	Description
quickActionListItems	DescribeQuickActionListItemResult[]	Array of zero or more QuickActionListItemResult objects.

DescribeQuickActionListItemResult

Represents a QuickAction assigned to the actions list for a page layout. Available in API version 28.0 and later.

Name	Type	Description
colors	DescribeColor[]	Array of color information. Each color is associated with a theme. This field is available in API version 29.0 and later.
iconUrl	string	The URL of the icon associated with the action. This icon URL corresponds to the 32x32 icon used for the current Salesforce theme, introduced in Spring '10.
icons	DescribeIcon[]	Array of icons for this action. Each icon is associated with a theme. This field is available in API version 29.0 and later.
label	string	The label of the action.
miniIconUrl	string	The URL of the mini icon associated with the action. This icon URL corresponds to the 16x16 icon used for the current Salesforce theme, introduced in Spring '10.
quickActionName	string	The API name of the action.
targetSobjectType	string	The API name of the action's target object.
type	string	The QuickActionType of the action. Valid values are: • Create • VisualforcePage

CustomLinkComponent

When the LayoutComponentType value is CustomLink, this type contains information about a single custom link on the page layout.

Name	Type	Description
customLink	DescribeLayoutButton	A single LayoutComponent object of type CustomLink.

FieldLayoutComponent

Extends the information returned by DescribeLayoutComponent. When the LayoutComponentType value is `Field`, and the field being described is an `Address` or `Person Name` field, FieldLayoutComponent includes information about the field's components.

Available in API version 31.0 and later.

Name	Type	Description
`components`	DescribeLayoutComponent[]	Array of zero or more LayoutComponent objects of type `Field`.
`fieldType`	FieldType	The field type.

Sample Code for Usage of FieldLayoutComponent

```
        DescribeLayoutComponent layoutComponent = layoutComponents[n];
        // Look for a component representing the BillingAddress field
        if (layoutComponent.getType() == LayoutComponentType.Field.toString() &&
        layoutComponent.getValue().equals("BillingAddress") {
        // Cast this component as a FieldLayoutComponent
        DescribeLayoutComponent.FieldLayoutComponent addressFieldComponent =
(FieldLayoutComponent)layoutComponent;
        // At this point you can access addressFieldComponent
FieldLayoutComponent-specific methods such as getComponents() or
getFieldType()
        }
```

LayoutComponentType

Represents the type for a DescribeLayoutComponent. Contains one of these values:

- `Canvas`—A canvas component on the page layout. This layout component type is available in API version 31.0 and later.
- `CustomLink`—A custom link on the page layout.
- `EmptySpace`—A blank space on the page layout.
- `ExpandedLookup`—An Expanded Lookup component in the Mobile Cards section of the page layout.
- `Field`—Field name. A mapping to the `name` field on the describeSObjectResult.
- `ReportChart`—A report chart on the page layout.
- `SControl`—Reserved for future use.
- `Separator`—Separator character, such as a semicolon (:) or slash (/).
- `VisualforcePage`—A Visualforce component on the page layout.

PicklistForRecordType

Represents a single record type picklist in a RecordTypeMapping. The `picklistName` matches up with the `name` attribute of each field in the `fields` array in describeSObjectResult. The `picklistValues` are the set of acceptable values for the `recordType`.

Name	Type	Description
`picklistName`	string	Name of the picklist.

Name	Type	Description
picklistValues	PicklistEntry[]	Set of picklist values associated with the recordTypeIds in the RecordTypeMapping. **Note:** If you retrieve picklistValues, the validFor value is null. If you need the validFor value, get it from the PicklistEntry object obtained from the Field object associated with the DescribeSObjectResult.

RecordTypeMapping

Represents a single record type mapping in the recordTypeMappings field in a DescribeLayoutResult object. This object is a map of valid recordTypeIds to layoutId. For displaying a detail view, a client application uses this mapping to determine which layout is associated with the record type on the record. For displaying an edit view, a client application uses this mapping to determine which layout to use (and possibly to allow the user to choose between multiple record types); it will also determine the set of available picklist values.

Name	Type	Description
available	boolean	Indicates whether this record type is available (true) or not (false). Availability is used to display a list of available record types to the user when they are creating a new record.
defaultRecordTypeMapping	boolean	Indicates whether this is the default record type mapping (true) or not (false).
layoutId	ID	ID of the layout associated with this record type.
name	string	Name of this record type.
picklistsForRecordType	PicklistForRecordType[]	Record type picklist(s) mapped to the recordTypeIds.
recordTypeId	ID	ID of this record type.

Note: Some fields previously in this result have moved to RecordTypeInfo.

RelatedContent

Represents the Mobile Cards section in a DescribeLayout. Available in API version 29.0 and later.

Name	Type	Description
relatedContentItems	DescribeRelatedContentItem[]	An array of items in the Mobile Cards section of the page layout.

DescribeRelatedContentItem

Represents an individual item in the DescribeRelatedContentItem list. Available in API version 29.0 and later.

Name	Type	Description
describeLayoutItem	DescribeLayoutItem	An individual layout item in the Mobile Cards section. Must be wrapped in a

Name	Type	Description
		DescribeRelatedContentItem to be added to the Mobile Cards section.

RelatedList

Represents a single related list in a DescribeLayoutResult.

Name	Type	Description
`columns`	RelatedListColumn[]	Columns associated with this related list. You can pair this value with Field to achieve a number of useful tasks, including determining whether the field is: • A name field, in order to present a link to the detail • Sortable, (to allow the user to include it in an `ORDER BY` clause to sort the rows by the given column • A currency field, to include the currency symbol or code
`custom`	boolean	If `true`, this related list is custom.
`field`	string	Name of the field on the related (associated) object that establishes the relationship with the associating object. For example, for the Contact related list on Account, the value is `AccountId`.
`label`	string	Label for the related list, displayed in the Salesforce user interface.
`limitRows`	int	Number of rows to display.
`name`	string	Name of the ChildRelationship in the DescribeSObjectResult for the `sObjectType` which was provided as the argument to DescribeLayout.
`sobject`	string	Name of the `sObjectType` that is the row type for rows within this related list.
`sort`	RelatedListSort[]	If not null, the column(s) that should be used to order the related objects.

RelatedListColumn

Represents a single field in a related list returned by DescribeLayoutResult.

Name	Type	Description
`field`	string	API name of the field. This value is always of the form ***`object_type.field_name`***. For example, if `name` is `Contact.Account.Owner.Alias`, then this value is `User.Alias`.
`format`	string	Display in date or dateTime format.
`label`	string	Label of the field.

Name	Type	Description
lookupId	string	Optional SOQL field syntax to retrieve the lookup ID value for the main related list sObject. This value may be an expression that uses SOQL relationship query dot notation. For example, if the related list sObjectType is Case and the column display value is Owner.Alias, then the lookup ID value would be Owner.Id.
name	string	SOQL field syntax for the field in relation to the main sObject for the related list. This value may be an expression that uses SOQL relationship query dot notation, or it may use the toLabel() or convertCurrency() format. For example, if the related list sObjectType is Case, then the value might be Owner.Alias or it might be toLabel(Case.Status).

RelatedListSort

Represents the sorting preference for objects in the related list.

Name	Type	Description
column	string	Name of the field that is used to order the related objects.
ascending	boolean	If true, sort order is ascending. If false, descending.

Although in most cases there is only one RelatedListSort in the array, for some special standard related lists, there is more than one. If there is more than one, the RelatedListSorts are ordered according to how they should be included in a corresponding SOQL query, for example:

```
ORDER BY relatedListSort[0].getColumn() DIRECTION, relatedListSort[1].getColumn() DIRECTION
```

WebLinkPosition

Represents the window position for a new window opened upon clicking a DescribeLayoutButton. Applies only to custom buttons. Available in API version 31.0 and later. Contains one of these values:

- fullScreen—The new window opens in a full screen. If this option is selected, any width or height parameters set for the new window are ignored.
- none—No window position preference is set.
- topLeft—The new window opens, positioned at the top left of the screen.

WebLinkType

Represents the content being delivered by the custom button. Contains one of these values:

- javascript
- page—Visualforce page
- sControl
- url

WebLinkWindowType

Represents the behavior for a DescribeLayoutButton. Applies only to custom buttons. Available in API version 31.0 and later. Contains one of these values:

- `newWindow`—The custom button's content opens in a new browser window.
- `noSidebar`—The custom button's content displays in the existing browser window without a sidebar.
- `onClickJavaScript`—Valid only when the DescribeLayoutButton's `contentSource` field value is `javascript`. Clicking the button or link executes JavaScript.
- `replace`—The custom button's content displays in the existing browser window without a sidebar or header.
- `sidebar`—The custom button's content displays in the existing browser window with a sidebar.

describePrimaryCompactLayouts()

Retrieves metadata about the primary compact layout for each of the specified object types. Information returned is limited to 100 objects.

Syntax

```
DescribeCompactLayout[] primaryCompactLayouts =
connection.describePrimaryCompactLayouts(string[] sObjectType)
```

Usage

Use this call to retrieve information about the primary compact layout for the given object types. This call returns metadata about a given primary compact layout. For more information about compact layouts, see the Salesforce Help.

Sample Code—Java

```
public void testDescribePrimaryCompactLayoutsSample() {
  try {
    String[] objectsToDescribe = new String[] {"Account","Lead"};
      DescribeCompactLayout[] primaryCompactLayouts =
connection.describePrimaryCompactLayouts(objectsToDescribe);

      for (int i = 0; i < primaryCompactLayouts.length; i++) {
        DescribeCompactLayout cLayout = primaryCompactLayouts[i];
        System.out.println(" There is a compact layout with name: " + cLayout.getName());

        // Write the objectType
        System.out.println(" This compact layout is the primary compact layout for: " +
cLayout.getObjectType());

        DescribeLayoutItem[] fieldItems = cLayout.getFieldItems();
        System.out.println(" There are " + fieldItems.length + " fields in this compact
layout.");

        // Write field items
        for (int j = 0; j < fieldItems.length; j++) {
          System.out.println(j + " This compact layout has a field with name: " +
fieldItems[j].getLabel());
        }

        DescribeLayoutItem[] imageItems = cLayout.getImageItems();
        System.out.println(" There are " + imageItems.length + " image fields in this
compact layout.");

        // Write the image items
        for (int j = 0; j < imageItems.length; j++) {
          System.out.println(j + " This compact layout has an image field with name: " +
imageItems[j].getLabel());
        }

        DescribeLayoutButton[] actions = cLayout.getActions();
        System.out.println(" There are " + actions.length + " buttons in this compact
```

```
layout.");

        // Write the action buttons
        for (int j = 0; j < actions.length; j++) {
          System.out.println(j + " This compact layout has a button with name: " +
actions[j].getLabel());
        }
      }

    } catch (ConnectionException ce) {
      ce.printStackTrace();
  }
}
```

Arguments

Name	Type	Description
sObjectTypes	string[]	An array of one or more objects. The specified values must be valid objects for your organization.

Response

DescribeCompactLayout

Faults

InvalidSObjectFault

UnexpectedErrorFault

describeQuickActions()

Retrieves details about specified actions.

Note: In the application, QuickActions are referred to as actions or publisher actions.

Syntax

```
DescribeQuickActionResult[] = connection.describeQuickActions(string[] quickActionNames);
```

Usage

Use the `describeQuickActions()` call to retrieve details for specified actions. In API version 28.0, the `describeQuickActions()` call takes the action name in the form of *ParentEntity.ActionName*. In API version 29.0 and greater, it takes the action name in the form of *ContextEntity.ActionName*. Returns an array of DescribeQuickActionResult. You might first call `describeAvailableQuickActions()` for a list of actions available for a specified context and then use `describeQuickActions()` to obtain details about specific actions.

Sample—Java

This sample retrieves and displays publisher action details for a create action on the Account object.

```
public void example() throws Exception {
    DescribeQuickActionResult[] result =
```

```
        conn.describeQuickActions(new String[]
            { "Account.QuickCreateContact", "Account.QuickCreateTask" });
    for(DescribeQuickActionResult r : result) {
        assert r != null;
    DescribeQuickActionDefaultValue [] describeQuickActionDefaultValues =
r.getDefaultValues();
    for(DescribeQuickActionDefaultValue defaultValue : describeQuickActionDefaultValues)
{
        System.out.println("Target Object Field:  " + defaultValue.getField() );
        System.out.println("Target Object Field's default Value:  " +
defaultValue.getDefaultValue );
}
        System.out.println("Action name: " + r.getName());
        System.out.println("Action label: " + r.getLabel());
        System.out.println("ParentOrContext object: " + r.getSourceSobjectType());
        System.out.println("Target object: " + r.getTargetSobjectType());
        System.out.println("Target object record type: " + r.getTargetRecordTypeId());
        System.out.println("Relationship field: " + r.getTargetParentField());
        System.out.println("Quick action type: " + r.getType());
        System.out.println("VF page name for custom actions: " +
        r.getVisualforcePageName());
        System.out.println("Icon name: " + r.getIconName());
        System.out.println("Icon URL: " + r.getIconUrl());
        System.out.println("Mini icon URL: " + r.getMiniIconUrl());
        assert r.getLayout() != null;
        System.out.println("Height of VF page for custom actions: " + r.getHeight());
        System.out.println("Width of VF page for custom actions: " + r.getWidth());
    }
}
```

Arguments

Name	Type	Description
quickActions	string[]	An array of quick actions to be retrieved.

Response

DescribeQuickActionResult

DescribeQuickActionResult

The `describeQuickActions()` call returns an array of DescribeQuickActionResult objects. Each DescribeQuickActionResult object represents a publisher quick action for a specified object.

Note: In the application, QuickActions are referred to as actions or publisher actions.

Name	Type	Description
canvasApplicationName	string	The name of your Force.com Canvas application, if you use it.
colors	DescribeColor[]	Array of color information. Each color is associated with a theme. This field is available in API version 29.0 and later.
defaultValues	DescribeQuickActionDefaultValue[]	The action's default values.
height	int	The height in pixels of the action pane.

Name	Type	Description
iconName	string	Name of icon used for the action. If a custom icon is not used, this value will not be set.
iconUrl	string	URL of icon used for the action. This icon URL corresponds to the 32x32 icon used for the current Salesforce theme, introduced in Spring '10, or the custom icon, if there is one.
icons	DescribeIcon[]	Array of icons. Each icon is associated with a theme. If no custom icon was associated with the quick action and the quick action creates a specific object, the icons will correspond to the icons used for the created object. For example, if the quick action creates an Account, the icon array will contain the icons used for Account. If a custom icon was associated with the quick action, the array will contain that custom icon. This field is available in API version 29.0 and later.
label	string	Label of the action.
layout	DescribeLayoutSection	All the sections that comprise a layout.
miniIconUrl	string	The icon's URL. This icon URL corresponds to the 16x16 icon used for the current Salesforce theme, introduced in Spring '10, or the custom icon, if there is one.
name	string	Name of the action.
contextSobjectType	string	The object used for the action. Named sourceSobjectType in version 29.0 and earlier.
targetParentField	string	The parent object type of the action. Links the target object to the parent object. For example, use Account if the target object is Contact and the parent object is Account.
targetRecordTypeId	ID	The record type of the target record.
targetSobjectType	string	The action's target object type.
type	string	• LogACall • SocialPost • Canvas • Create • VisualforcePage • Update
visualforcePageName	string	If Visualforce is used, the page name of the associated page for the action.
width	int	If a custom action is created, this is the width in pixels of the action pane.

DescribeQuickActionDefaultValue

Represents the default values of fields to use in default layouts.

Name	Type	Description
`defaultValue`	string	The value of the auto-populated default action.
`field`	string	The field name of the action.

DescribeLayoutSection

Represents a section of a DescribeLayout and consists of one or more columns and one or more rows (an array of DescribeLayoutRow).

Name	Type	Description
`columns`	int	Number of columns in this DescribeLayoutSection.
`heading`	string	Heading text (label) for this DescribeLayoutSection.
`layoutRows`	DescribeLayoutRow[]	Array of one or more DescribeLayoutRow objects.
`rows`	int	Number of rows in this DescribeLayoutSection.
`tabOrder`	string	Indicates the tab order for the fields in the section in the edit view. Valid values are: • LeftToRight • TopToBottom This field is available in API version 31.0 and later.
`useCollapsibleSection`	boolean	Indicates whether this DescribeLayoutSection is a collapsible section, also known as a “twistie” (`true`), or not (`false`).
`useHeading`	boolean	Indicates whether to display the `heading` (`true`) or not (`false`).

DescribeLayoutRow

Represents a row in a DescribeLayoutSection. A DescribeLayoutRow consists of one or more DescribeLayoutItem objects. For each DescribeLayoutRow, a DescribeLayoutItem refers either to a specific field or to an “empty” DescribeLayoutItem (a DescribeLayoutItem that contains no DescribeLayoutComponent objects). An empty DescribeLayoutItem can be returned when a given DescribeLayoutRow is sparse (for example, containing more fields on the right column than on the left column). Where there are gaps in the layout, an empty DescribeLayoutItem is returned as a placeholder.

Name	Type	Description
`layoutItems`	DescribeLayoutItem[]	Refers to either a specific field or to an empty LayoutItem (a LayoutItem that contains no DescribeLayoutComponent objects).
`numItems`	int	Number of `layoutItems`. This information is redundant but, due to a bug in a popular SOAP toolkit, was required to avoid serialization problems.

DescribeLayoutItem

Represents an individual item in a DescribeLayoutRow. A DescribeLayoutItem consists of a set of components (DescribeLayoutComponent), each of which is either a field or a separator. For most fields on a layout, there is only one component per layout item. However, in a display-only view, the DescribeLayoutItem might be a composite of the individual fields (for example, an address can consist of street, city, state, country, and postal code data). On the corresponding edit view, each component of the address field would be split up into separate DescribeLayoutItems.

Name	Type	Description
editable	boolean	Indicates whether this DescribeLayoutItem can be edited (true) or not (false). This field is available in API version 30.0 and below. It was replaced by the editableForNew and editableForUpdate fields in API version 31.0.
editableForNew	boolean	Indicates whether a new DescribeLayoutItem can be edited when creating a new record (true) or not (false). This field is available in API version 31.0 and later.
editableForUpdate	boolean	Indicates whether an existing DescribeLayoutItem can be edited when editing a record (true) or not (false). This field is available in API version 31.0 and later.
label	string	Label text for this DescribeLayoutItem.
layoutComponents	DescribeLayoutComponent[]	DescribeLayoutComponent for this DescribeLayoutItem.
placeholder	boolean	Indicates whether this DescribeLayoutItem is a placeholder (true) or not (false). If true, then this DescribeLayoutItem is blank.
required	boolean	Indicates whether this DescribeLayoutItem is required (true) or not (false). This is useful to know if, for example, you wanted to render required fields in a contrasting color (such as red).

DescribeLayoutComponent

Represents the smallest unit in a layout—a field or a separator. To reference a field for display, a client application uses the following notation to reference a field in the `describeSObjects()` call: `LayoutComponent.`*`fieldName`*.

Name	Type	Description
displayLines	int	The number of vertical lines displayed for a field in the edit view. Applies to textarea and multi-select picklist fields.
tabOrder	int	Indicates the tab order for the item in the row.
type	LayoutComponentType	The LayoutComponentType for this LayoutComponent.
value	string	Value of this LayoutComponent. The name of the field if the LayoutComponentType value is Field. The API name of the canvas app if the LayoutComponentType value is Canvas.

LayoutComponentType

Represents the type for a DescribeLayoutComponent.Contains one of these values:

- `Canvas`—A canvas component on the page layout. This layout component type is available in API version 31.0 and later.
- `CustomLink`—A custom link on the page layout.
- `EmptySpace`—A blank space on the page layout.
- `ExpandedLookup`—An Expanded Lookup component in the Mobile Cards section of the page layout.
- `Field`—Field name. A mapping to the `name` field on the describeSObjectResult.
- `ReportChart`—A report chart on the page layout.
- `SControl`—Reserved for future use.

- `Separator`—Separator character, such as a semicolon (:) or slash (/).
- `VisualforcePage`—A Visualforce component on the page layout.

describeSearchScopeOrder()

Retrieves an ordered list of the objects in a user's default global search scope.

Syntax

```
DescribeSearchScopeOrderResult[] describeSearchScopeOrderResults =
connection.describeSearchScopeOrder();
```

Usage

Use `describeSearchScopeOrder()` to retrieve an ordered list of objects in the default global search scope of a logged-in user. Global search keeps track of which objects the user interacts with and how often and arranges the search results accordingly. Objects used most frequently appear at the top of the list. The returned list reflects the object order in the user's default search scope, including any pinned objects on the user's search results page. This call is useful if you want to implement a custom search results page using the optimized global search scope.

Note: You must enable Chatter to enable global search. For more information about global search, see Search Overview in the Salesforce online help.

Sample Code—Java

This sample shows how to retrieve the global search scope for a user and then iteratively display the name of each object in the scope.

```
public void describeSearchScopeOrderSample() {
  try {
    //Get the order of objects in search smart scope for the logged-in user
    DescribeSearchScopeOrderResult[] describeSearchScopeOrderResults =
        connection.describeSearchScopeOrder();
    //Iterate through the results and display the name of each object
    for (int i = 0; i < describeSearchScopeOrderResults.length; i++) {
      System.out.println(describeSearchScopeOrderResults[i].getName());
    }
  }
  catch(ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

Arguments

None.

Response

An array of DescribeSearchScopeOrderResult objects

Fault

UnexpectedErrorFault

See Also:

API Call Basics

DescribeSearchScopeOrderResult

The `describeSearchScopeOrder()` call returns an array of `DescribeSearchScopeOrderResult` objects. Each `DescribeSearchScopeOrderResult` object represents an object in the user's global search scope. The list reflects the order of the objects in the user's scope, including any pinned objects. The `DescribeSearchScopeOrderResult` object has the following properties.

Name	Type	Description
keyPrefix	string	Three-character prefix code in the object ID. Object IDs are prefixed with three-character codes that specify the type of the object. For example, Account objects have a prefix of `001` and Opportunity objects have a prefix of `006`. Note that a key prefix can sometimes be shared by multiple objects so it does not always uniquely identify an object.
name	string	Name of the object. English only.

describeSearchLayouts()

Retrieves the search result layout configuration for one or more objects.

Syntax

```
DescribeSearchLayoutResult[] = binding.describeSearchLayouts(string sObjectType[]);
```

Usage

Use `describeSearchLayouts()` to retrieve search layout information for one or more objects. This is handy when you want to create a custom search results page with the same layout settings as in Salesforce.

Sample

This sample shows how to retrieve the search result layout information for a list of objects.

```
public void describeSearchLayoutSample(String[] sObjectTypes) {
        try {
            // Get the search layout of Account and Group
            DescribeSearchLayoutResult[] searchLayoutResults =
connection.describeSearchLayouts(sObjectTypes);
            // Iterate through the results and display the label of each column
            for (int i = 0; i < sObjectTypes.length; i += 1) {
                String sObjectType = sObjectTypes[i];
                DescribeSearchLayoutResult result = searchLayoutResults[i];
                System.out.println("Top label for search results for " + sObjectType + "
is " + result.getLabel() + " and should display " + result.getLimitRows() + " rows");
```

```
                System.out.println("Column labels for search results for " + sObjectType
+ " are: ");
                for (DescribeColumn column : result.getSearchColumns()) {
                    System.out.println(column.getLabel());
                }
            }
        } catch (ConnectionException ce) {
            ce.printStackTrace();
        }
    }
```

Arguments

Name	Type	Description
sObjectType	string[]	The list of objects you want to obtain search result layout configuration for. For example, if the object is a person account, specify Account, or if it is a person contact, specify Contact. The specified values must be valid objects in your organization. For a complete list of standard objects, see Standard Objects.

Response

DescribeSearchLayoutResult

Faults

InvalidSObjectFault

UnexpectedErrorFault

DescribeSearchLayoutResult

The `describeSearchLayouts()` on page 1571 call returns an array of `DescribeSearchLayoutResult` objects. Each `DescribeSearchLayoutResult` object represents the search layout configuration for each object queried for. The `DescribeSearchLayoutResult` object has the following properties.

Name	Type	Description
label	string	The browser title used for the search results page.
limitRows	int	The maximum number of rows to be displayed in the first page of search results. This number can be changed by the administrator.
searchColumns	DescribeColumn on page 1572[]	The columns associated with the search results for this object.

DescribeColumn

Represents the columns in the search layout configuration for each DescribeSearchLayoutResult object returned by the `describeSearchLayouts()` on page 1571 call.

Name	Type	Description
field	string	Field reference in relation to the object it belongs to. For example, “Lead.Phone.”
format	string	Field data format. For example, “date”. This value can be null.

label	string	Display text for this field in the user interface. For example, “Company Phone” or just “Phone.”
name	string	Field name. Use this in your SOQL query or code. For example, “Name.”

describeSObject()

Describes metadata (field list and object properties) for the specified object.

Note: describeSObjects() supersedes describeSObject(). Use describeSObjects() instead of describeSObject().

Syntax

```
DescribeSObjectResult = connection.describeSObject(string sObjectType);
```

Usage

Use describeSObject() to obtain metadata for a given object. You can first call describeGlobal() to retrieve a list of all objects for your organization, then iterate through the list and use describeSObject() to obtain metadata about individual objects.

Your client application must be logged in with sufficient access rights to retrieve metadata about your organization’s data. For more information, see Factors that Affect Data Access.

Sample Code—Java

This sample calls describeSObject() to perform describes on the Account sObject. It retrieves some properties of the sObject describe result, such as the sObject name, label, and fields. It then iterates through the fields and gets the field properties. For picklist fields, it gets the picklist values and for reference fields, it gets the referenced object names. The sample writes the retrieved sObject and field properties to the console.

```
public void describeSObjectSample() {
    try {
        // Make the describe call
        DescribeSObjectResult describeSObjectResult =
            connection.describeSObject("Account");

        // Get sObject metadata
        if (describeSObjectResult != null) {
          System.out.println("sObject name: " +
                  describeSObjectResult.getName());
        if (describeSObjectResult.isCreateable())
            System.out.println("Createable");

        // Get the fields
        Field[] fields = describeSObjectResult.getFields();
        System.out.println("Has " + fields.length + " fields");

        // Iterate through each field and gets its properties
        for (int i = 0; i < fields.length; i++) {
          Field field = fields[i];
          System.out.println("Field name: " + field.getName());
          System.out.println("Field label: " + field.getLabel());

          // If this is a picklist field, show the picklist values
          if (field.getType().equals(FieldType.picklist)) {
```

```
                PicklistEntry[] picklistValues =
                    field.getPicklistValues();
                if (picklistValues != null) {
                  System.out.println("Picklist values: ");
                  for (int j = 0; j < picklistValues.length; j++) {
                    if (picklistValues[j].getLabel() != null) {
                      System.out.println("\tItem: " +
                          picklistValues[j].getLabel()
                      );
                    }
                  }
                }
            }

            // If a reference field, show what it references
            if (field.getType().equals(FieldType.reference)) {
                System.out.println("Field references the " +
                        "following objects:");
                String[] referenceTos = field.getReferenceTo();
                for (int j = 0; j < referenceTos.length; j++) {
                    System.out.println("\t" + referenceTos[j]);
                }
            }
      }
    }
  } catch (ConnectionException ce) {
      ce.printStackTrace();
    }
}
```

Sample Code—C#

This sample calls `describeSObject()` to perform describes on the Account sObject. It retrieves some properties of the sObject describe result, such as the sObject name, label, and fields. It then iterates through the fields and gets the field properties. For picklist fields, it gets the picklist values and for reference fields, it gets the referenced object names. The sample writes the retrieved sObject and field properties to the console.

```
public void describeSObjectSample() {
    try {
        // Make the describe call
        DescribeSObjectResult describeSObjectResult =
            binding.describeSObject("Account");

        // Get sObject metadata
        if (describeSObjectResult != null) {
            Console.WriteLine("sObject name: " +
                    describeSObjectResult.name);
        if (describeSObjectResult.createable)
            Console.WriteLine("Createable");

        // Get the fields
        Field[] fields = describeSObjectResult.fields;
        Console.WriteLine("Has " + fields.Length + " fields");

        // Iterate through each field and gets its properties
        for (int i = 0; i < fields.Length; i++) {
            Field field = fields[i];
            Console.WriteLine("Field name: " + field.name);
            Console.WriteLine("Field label: " + field.label);

            // If this is a picklist field, show the picklist values
            if (field.type.Equals(fieldType.picklist)) {
                PicklistEntry[] picklistValues =
                    field.picklistValues;
                if (picklistValues != null) {
                    Console.WriteLine("Picklist values: ");
                    for (int j = 0; j < picklistValues.Length; j++) {
```

```
                        if (picklistValues[j].label != null) {
                        Console.WriteLine("\tItem: " +
                            picklistValues[j].label);
                        }
                    }
                }
            }

            // If a reference field, show what it references
            if (field.type.Equals(fieldType.reference)) {
                Console.WriteLine("Field references the " +
                      "following objects:");
                String[] referenceTos = field.referenceTo;
                for (int j = 0; j < referenceTos.Length; j++) {
                    Console.WriteLine("\t" + referenceTos[j]);
                }
            }
        }
        }
    } catch (SoapException e) {
        Console.WriteLine("An unexpected error has occurred: " +
            e.Message + "\n" + e.StackTrace);
    }
}
```

Arguments

Name	Type	Description
sObjectType	string	Object. The specified value must be a valid object for your organization. For a complete list of objects, see Standard Objects.

Response

DescribeSObjectResult

Faults

InvalidSObjectFault

UnexpectedErrorFault

See Also:

describeSObjects()
describeGlobal()
API Call Basics
Using the Partner WSDL
https://developer.salesforce.com/page/Sample_SOAP_Messages

describeSObjectResult

The `describeSObject()` call returns a `DescribeSObjectResult` object.

Note: `describeSObjects()` supersedes `describeSObject()`. Use `describeSObjects()` instead of `describeSObject()`.

describeSObjects()

An array-based version of `describeSObject()`; describes metadata (field list and object properties) for the specified object or array of objects.

Note: Use this call instead of `describeSObject()`.

Syntax

```
DescribeSObjectResult [] = connection.describeSObjects(string sObjectType[] );
```

Usage

Use `describeSObjects()` to obtain metadata for a given object or array of objects. You can first call `describeGlobal()` to retrieve a list of all objects for your organization, then iterate through the list and use `describeSObjects()` to obtain metadata about individual objects. The `describeSObjects()` call is limited to a maximum of 100 objects returned.

Your client application must be logged in with sufficient access rights to retrieve metadata about your organization's data. For more information, see Factors that Affect Data Access.

In organizations where person accounts are enabled, this call shows Accounts as not createable if the profile does not have access to any business account record types.

Sample Code—Java

This sample calls `describeSObjects()` to perform describes on account, contact, and lead. It iterates through the sObject describe results, gets the properties and fields for each sObject in the result, and writes them to the console. For picklist fields, it writes the picklist values. For reference fields, it writes the referenced object names.

```
public void describeSObjectsSample()
{
  try {
    // Call describeSObjectResults and pass it an array with
    // the names of the objects to describe.
    DescribeSObjectResult[] describeSObjectResults =
                       connection.describeSObjects(
                        new String[] { "account", "contact", "lead" });

    // Iterate through the list of describe sObject results
    for (int i=0;i < describeSObjectResults.length; i++)
    {
        DescribeSObjectResult desObj = describeSObjectResults[i];
        // Get the name of the sObject
        String objectName = desObj.getName();
        System.out.println("sObject name: " + objectName);

        // For each described sObject, get the fields
        Field[] fields = desObj.getFields();

        // Get some other properties
        if (desObj.getActivateable()) System.out.println("\tActivateable");

        // Iterate through the fields to get properties for each field
        for(int j=0;j < fields.length; j++)
        {
            Field field = fields[j];
            System.out.println("\tField: " + field.getName());
            System.out.println("\t\tLabel: " + field.getLabel());
```

```
            if (field.isCustom())
                System.out.println("\t\tThis is a custom field.");
            System.out.println("\t\tType: " + field.getType());
            if (field.getLength() > 0)
                System.out.println("\t\tLength: " + field.getLength());
            if (field.getPrecision() > 0)
                System.out.println("\t\tPrecision: " + field.getPrecision());

            // Determine whether this is a picklist field
            if (field.getType() == FieldType.picklist)
            {
                // Determine whether there are picklist values
                PicklistEntry[] picklistValues = field.getPicklistValues();
                if (picklistValues != null && picklistValues[0] != null)
                {
                    System.out.println("\t\tPicklist values = ");
                    for (int k = 0; k < picklistValues.length; k++)
                    {
                       System.out.println("\t\t\tItem: " + picklistValues[k].getLabel());

                    }
                }
            }

            // Determine whether this is a reference field
            if (field.getType() == FieldType.reference)
            {
                // Determine whether this field refers to another object
                String[] referenceTos = field.getReferenceTo();
                if (referenceTos != null && referenceTos[0] != null)
                {
                    System.out.println("\t\tField references the following objects:");
                    for (int k = 0; k < referenceTos.length; k++)
                    {
                        System.out.println("\t\t\t" + referenceTos[k]);
                    }
                }
            }
        }
    }
  } catch(ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

Sample Code—C#

This sample calls `describeSObjects()` to perform describes on account, contact, and lead. It iterates through the sObject describe results, gets the properties and fields for each sObject in the result, and writes them to the console. For picklist fields, it writes the picklist values. For reference fields, it writes the referenced object names.

```
public void describeSObjectsSample()
{
    try
    {
    // Call describeSObjectResults and pass it an array with
    // the names of the objects to describe.
    DescribeSObjectResult[] describeSObjectResults =
                        binding.describeSObjects(
                        new string[] { "account", "contact", "lead" });

    // Iterate through the list of describe sObject results
    foreach (DescribeSObjectResult describeSObjectResult in describeSObjectResults)
    {
        // Get the name of the sObject
        String objectName = describeSObjectResult.name;
        Console.WriteLine("sObject name: " + objectName);
```

```
        // For each described sObject, get the fields
        Field[] fields = describeSObjectResult.fields;

        // Get some other properties
        if (describeSObjectResult.activateable) Console.WriteLine("\tActivateable");

        // Iterate through the fields to get properties for each field
        foreach (Field field in fields)
        {
            Console.WriteLine("\tField: " + field.name);
            Console.WriteLine("\t\tLabel: " + field.label);
            if (field.custom)
                Console.WriteLine("\t\tThis is a custom field.");
            Console.WriteLine("\t\tType: " + field.type);
            if (field.length > 0)
                Console.WriteLine("\t\tLength: " + field.length);
            if (field.precision > 0)
                Console.WriteLine("\t\tPrecision: " + field.precision);

            // Determine whether this is a picklist field
            if (field.type == fieldType.picklist)
            {
                // Determine whether there are picklist values
                PicklistEntry[] picklistValues = field.picklistValues;
                if (picklistValues != null && picklistValues[0] != null)
                {
                    Console.WriteLine("\t\tPicklist values = ");
                    for (int j = 0; j < picklistValues.Length; j++)
                    {
                        Console.WriteLine("\t\t\tItem: " + picklistValues[j].label);
                    }
                }
            }

            // Determine whether this is a reference field
            if (field.type == fieldType.reference)
            {
                // Determine whether this field refers to another object
                string[] referenceTos = field.referenceTo;
                if (referenceTos != null && referenceTos[0] != null)
                {
                    Console.WriteLine("\t\tField references the following objects:");
                    for (int j = 0; j < referenceTos.Length; j++)
                    {
                        Console.WriteLine("\t\t\t" + referenceTos[j]);
                    }
                }
            }
        }
    }
    }
    catch (SoapException e)
    {
        Console.WriteLine("An unexpected error has occurred: " + e.Message
            + "\n" + e.StackTrace);
    }
}
```

Arguments

The `describeSObjects()` call takes in an array of sObjects.

Name	Type	Description
`sObjectType`	string	Object. The specified value must be a valid object for your organization. For a complete list of objects, see Standard Objects.

Response

DescribeSObjectResult

Faults

InvalidSObjectFault

UnexpectedErrorFault

See Also:

describeSObject()
describeGlobal()
API Call Basics
Using the Partner WSDL

DescribeSObjectResult

The `describeSObjects()` call returns an array of `DescribeSObjectResult` objects. Each object has the following properties:

Name	Type	Description
`activateable`	boolean	Reserved for future use.
`childRelationships`	ChildRelationship[]	An array of child relationships, which is the name of the sObject that has a foreign key to the sObject being described.
`compactLayoutable`	boolean	Indicates that the object can be used in `describeCompactLayouts()`.
`createable`	boolean	Indicates whether the object can be created via the `create()` call (`true`) or not (`false`).
`custom`	boolean	Indicates whether the object is a custom object (`true`) or not (`false`).
`customSetting`	boolean	Indicates whether the object is a custom setting object (`true`) or not (`false`).
`deletable`	boolean	Indicates whether the object can be deleted via the `delete()` call (`true`) or not (`false`).
`deprecatedAndHidden`	boolean	Reserved for future use.
`feedEnabled`	boolean	Indicates whether Chatter feeds are enabled for the object (`true`) or not (`false`). This property is available in API version 19.0 and later.
`fields`	Field[]	Array of fields associated with the object. The mechanism for retrieving information from this list varies among development tools.
`keyPrefix`	string	Three-character prefix code in the object ID. Object IDs are prefixed with three-character codes that specify the type of the object. For example, Account objects have a prefix of `001` and Opportunity objects have a prefix of `006`. Note that a key prefix can sometimes be shared by multiple objects so it does not always uniquely identify an object.

Name	Type	Description
		Use the value of this field to determine the object type of a parent in those cases where the child may have more than one object type as parent (polymorphic). For example, you may need to obtain the `keyPrefix` value for the parent of a Task or Event.
`label`	string	Label text for a tab or field renamed in the user interface, if applicable, or the object name, if not. For example, an organization representing a medical vertical might rename Account to `Patient`. Tabs and fields can be renamed in the Salesforce user interface. See the Salesforce online help for more information.
`labelPlural`	string	Label text for an object that represents the plural version of an object name, for example, "Accounts."
`layoutable`	boolean	Indicates whether the object supports the `describeLayout()` call (`true`) or not (`false`).
`mergeable`	boolean	Indicates whether the object can be merged with other objects of its type (`true`) or not (`false`). `true` for leads, contacts, and accounts.
`name`	string	Name of the object. This is the same string that was passed in as the `sObjectType` parameter.
`namedLayoutInfos`	NamedLayoutInfo[]	The specific named layouts that are available for the objects other than the default layout.
`queryable`	boolean	Indicates whether the object can be queried via the `query()` call (`true`) or not (`false`).
`recordTypeInfos`	RecordTypeInfo[]	An array of the record types supported by this object. The user need not have access to all the returned record types to see them here.
`replicateable`	boolean	Indicates whether the object can be replicated via the `getUpdated()` and `getDeleted()` calls (`true`) or not (`false`).
`retrieveable`	boolean	Indicates whether the object can be retrieved via the `retrieve()` call (`true`) or not (`false`).
`searchable`	boolean	Indicates whether the object can be searched via the `search()` call (`true`) or not (`false`).
`searchLayoutable`	boolean	Indicates whether search layout information can be retrieved via the `describeSearchLayouts()` call (`true`) or not (`false`).
`triggerable`	boolean	Indicates whether the object supports Apex triggers.
`undeletable`	boolean	Indicates whether an object can be undeleted using the `undelete()` call (`true`) or not (`false`).
`updateable`	boolean	Indicates whether the object can be updated via the `update()` call (`true`) or not (`false`).
`urlDetail`	string	URL to the read-only detail page for this object. Compare with `urlEdit`, which is read-write. Client applications can use this URL to redirect to, or access, the Salesforce user interface for standard and custom objects. To provide flexibility and allow for future enhancements, returned `urlDetail` values are dynamic. To ensure

Name	Type	Description
		that client applications are forward compatible, it is recommended that they use this capability where possible. Note that, for objects for which a stable URL is not available, this field is returned empty.
`urlEdit`	string	URL to the edit page for this object. For example, the `urlEdit` field for the Account object returns `https://na1.salesforce.com/{ID}/e`. Substituting the `{ID}` field for the current object ID will return the edit page for that specific account in the Salesforce user interface. Compare with `urlDetail`, which is read-only. Client applications can use this URL to redirect to, or access, the Salesforce user interface for standard and custom objects. To provide flexibility and allow for future enhancements, returned `urlDetail` values are dynamic. To ensure that client applications are forward compatible, it is recommended that they use this capability where possible. Note that, for objects for which a stable URL is not available, this field is returned empty.
`urlNew`	string	URL to the new/create page for this object. Client applications can use this URL to redirect to, or access, the Salesforce user interface for standard and custom objects. To provide flexibility and allow for future enhancements, returned `urlNew` values are dynamic. To ensure that client applications are forward compatible, it is recommended that they use this capability where possible. Note that, for objects for which a stable URL is not available, this field is returned empty.

Note: The properties with a Boolean value indicate whether certain API calls can be used for an object. However, other factors, such as permissions, also affect whether such operations can be performed on the object.

ChildRelationship

The name of the sObject that has a foreign key to the sObject being described.

Name	Type	Description
`cascadeDelete`	boolean	Indicates whether the child object is deleted when the parent object is deleted (`true`) or not (`false`).
`childSObject`	string	The name of the object on which there is a foreign key back to the parent sObject.
`deprecatedAndHidden`	boolean	Reserved for future use.
`field`	string	The name of the field that has a foreign key back to the parent sObject.
`relationshipName`	string	The name of the relationship, usually the plural of the value in `childSObject`.

Field

In the DescribeSObjectResult, the `fields` property contains an array of `Field` objects. Each field represents a field in an API object. The array contains only the fields that the user can view, as defined by the user's field-level security settings.

Name	Type	Description
autonumber	boolean	Indicates whether this field is an autonumber field (true) or not (false). Analogous to a SQL IDENTITY type, autonumber fields are read only, non-createable text fields with a maximum length of 30 characters. Autonumber fields are read-only fields used to provide a unique ID that is independent of the internal object ID (such as a purchase order number or invoice number). Autonumber fields are configured entirely in the Salesforce user interface. The API provides access to this attribute so that client applications can determine whether a given field is an autonumber field.
byteLength	int	For variable-length fields (including binary fields), the maximum size of the field, in bytes.
calculated	boolean	Indicates whether the field is a custom formula field (true) or not (false). Note that custom formula fields are always read-only.
caseSensitive	boolean	Indicates whether the field is case sensitive (true) or not (false).
controllerName	string	The name of the field that controls the values of this picklist. It only applies if type is picklist or multipicklist and dependentPicklist is true. See About Dependent Picklists. The mapping of controlling field to dependent field is stored in the validFor attribute of each PicklistEntry for this picklist. See validFor.
createable	boolean	Indicates whether the field can be created (true) or not (false). If true, then this field value can be set in a create() call.
custom	boolean	Indicates whether the field is a custom field (true) or not (false).
defaultedOnCreate	boolean	Indicates whether this field is defaulted when created (true) or not (false). If true, then Salesforce implicitly assigns a value for this field when the object is created, even if a value for this field is not passed in on the create() call. For example, in the Opportunity object, the Probability field has this attribute because its value is derived from the Stage field. Similarly, the Owner has this attribute on most objects because its value is derived from the current user (if the Owner field is not specified).
defaultValueFormula	string	The default value specified for this field if the formula is not used. If no value has been specified, this field is not returned.
dependentPicklist	boolean	Indicates whether a picklist is a dependent picklist (true) where available values depend on the chosen values from a controlling field, or not (false). See About Dependent Picklists.
deprecatedAndHidden	boolean	Reserved for future use.
digits	int	For fields of type integer. Maximum number of digits. The API returns an error if an integer value exceeds the number of digits.
displayLocationInDecimal	boolean	Indicates how the value of a Geolocation custom field appears in the user interface. If true, the geolocation appears in decimal notation. If false, the geolocation appears as degrees, minutes, and seconds.

Name	Type	Description
extraTypeInfo	string	If the field is a textarea field type, indicates if the text area is plain text (plaintextarea) or rich text (richtextarea). If the field is a url field type, if this value is imageurl, the URL references an image file. Available on standard fields on standard objects only, for example, Account.photoUrl, Contact.photoUrl, and so on.
filterable	boolean	Indicates whether the field is filterable (true) or not (false). If true, then this field can be specified in the WHERE clause of a query string in a query() call.
filteredLookupInfo	FilteredLookupInfo	If the field is a reference field type with a lookup filter, filteredLookupInfo contains the lookup filter information for the field. If there is no lookup filter, or the filter is inactive, this field is null. This field is available in API version 31.0 and later.
formula	string	The formula specified for this field. If no formula is specified for this field, it is not returned.
groupable	boolean	Indicates whether the field can be included in the GROUP BY clause of a SOQL query (true) or not (false). See GROUP BY in the *Salesforce SOQL and SOSL Reference Guide*. Available in API version 18.0 and later.
htmlFormatted	boolean	Indicates whether a field such as a hyperlink custom formula field has been formatted for HTML and should be encoded for display in HTML (true) or not (false). Also indicates whether a field is a custom formula field that has an IMAGE text function.
idLookup	boolean	Indicates whether the field can be used to specify a record in an upsert() call (true) or not (false).
inlineHelpText	string	The text that displays in the field-level help hover text for this field. **Note:** This property is not returned unless at least one field on the object contains a value. When at least one field has field-level help, all fields on the object list the property with either the field-level help value or null for fields that have blank field-level help.
label	string	Text label that is displayed next to the field in the Salesforce user interface. This label can be localized.
length	int	For string fields, the maximum size of the field in Unicode characters (not bytes).
mask	string	Reserved for future use.
maskType	string	Reserved for future use.
name	string	Field name used in API calls, such as create(), delete(), and query().
nameField	boolean	Indicates whether this field is a name field (true) or not (false). Used to identify the name field for standard objects (such as AccountName for an Account object) and custom objects. Limited to one per object, except where FirstName and LastName fields are used (such as in the Contact object).

Name	Type	Description
		If a compound name is present, for example the `Name` field on a person account, `nameField` is set to `true` for that record. If no compound name is present, `FirstName` and `LastName` have this field set to `true`.
`namePointing`	boolean	Indicates whether the field's value is the Name of the parent of this object (`true`) or not (`false`). Used for objects whose parents may be more than one type of object, for example a task may have an account or a contact as a parent.
`nillable`	boolean	Indicates whether the field is nillable (`true`) or not (`false`). A nillable field can have empty content. A non-nillable field must have a value in order for the object to be created or saved.
`permissionable`	boolean	Indicates whether FieldPermissions can be specified for the field (`true`) or not (`false`).
`picklistValues`	PicklistEntry[]	Provides the list of valid values for the picklist. Specified only if `restrictedPicklist` is `true`.
`precision`	int	For fields of type double. Maximum number of digits that can be stored, including all numbers to the left and to the right of the decimal point (but excluding the decimal point character).
`relationshipName`	string	The name of the relationship, if this is a master-detail relationship field.
`relationshipOrder`	int	The type of relationship for a master-detail relationship field. Valid values are: • `0` if the field is the primary relationship • `1` if the field is the secondary relationship
`referenceTo`	string[]	For fields that refer to other objects, this array indicates the object types of the referenced objects.
`restrictedPicklist`	boolean	Indicates whether the field is a restricted picklist (`true`) or not (`false`).
`scale`	int	For fields of type double. Number of digits to the right of the decimal point. The API silently truncates any extra digits to the right of the decimal point, but it returns a fault response if the number has too many digits to the left of the decimal point.
`soapType`	SOAPType	See SOAPType for a list of allowable values.
`sortable`	boolean	Indicates whether a query can sort on this field (`true`) or not (`false`).
`type`	FieldType	See FieldType for a list of allowable values.
`unique`	boolean	Indicates whether the value must be unique `true`) or not `false`).
`updateable`	boolean	Indicates one of the following: • Whether the field is updateable, (`true`) or not (`false`). If `true`, then this field value can be set in an `update()` call. • If the field is in a master-detail relationship on a custom object, indicates whether the child records can be reparented to different parent records (`true`), `false` otherwise.

Name	Type	Description
writeRequiresMasterRead	boolean	This field only applies to master-detail relationships. Indicates whether a user requires read sharing access (true) or write sharing access (false) to the parent record to insert, update, and delete a child record. In both cases, a user also needs Create, Edit, and Delete object permissions for the child object.

FieldType

In the Field object associated with the DescribeSObjectResult, the type field can contain one of the following strings. For more information about field types, see Field Types.

type Field Value	What the Field Object Contains
string	String values.
boolean	Boolean (true / false) values.
int	Integer values.
double	Double values.
date	Date values.
datetime	Date and time values.
base64	Base64-encoded arbitrary binary data (of type base64Binary). Used for Attachment, Document, and Scontrol objects.
ID	Primary key field for the object. For information on IDs, see ID Field Type.
reference	Cross-references to a different object. Analogous to a foreign key field in SQL.
currency	Currency values.
textarea	String that is displayed as a multiline text field.
percent	Percentage values.
phone	Phone numbers. Values can include alphabetic characters. Client applications are responsible for phone number formatting.
url	URL values. Client applications should commonly display these as hyperlinks. If Field.extraTypeInfo is imageurl, the URL references an image, and can be displayed as an image instead.
email	Email addresses.
combobox	Comboboxes, which provide a set of enumerated values and allow the user to specify a value not in the list.
picklist	Single-select picklists, which provide a set of enumerated values from which only one value can be selected.
multipicklist	Multi-select picklists, which provide a set of enumerated values from which multiple values can be selected.
anyType	Values can be any of these types: string, picklist, boolean, int, double, percent, ID, date, dateTime, url, or email.

type Field Value	What the Field Object Contains
location	Geolocation values, including latitude and longitude, for custom geolocation fields on custom objects.

FilteredLookupInfo

In the Field object associated with the DescribeSObjectResult, the filteredLookupInfo field contains information about the lookup filter associated with the field.

This subtype is available in API version 31.0 and later.

Name	Type	Description
controllingFields	string[]	Array of the field's controlling fields when the lookup filter is dependent on the source object.
dependent	boolean	Indicates whether the lookup filter is dependent upon the source object (true) or not (false).
optionalFilter	boolean	Indicates whether the lookup filter is optional (true) or not (false).

SOAPType

The DescribeSObjectResult returns the fields property, which contains an array of fields whose value provides information about the object being described. One of those fields, soapType, contains one of the following string values. All of the values preceded by xsd: are XML schema primitive data types. For more information about the XML schema primitive data types, see the World Wide Web Consortium's publication *XML Schema Part 2: Data Types* at: http://www.w3.org/TR/xmlschema-2/.

Value	Description
tns:ID	Unique ID associated with an sObject. For information on IDs, see ID Field Type.
xsd:anyType	Can be ID, Boolean, double, integer, string, date, or dateTime.
xsd:base64Binary	Base 64-encoded binary data.
xsd:boolean	Boolean (true / false) values.
xsd:date	Date values.
xsd:dateTime	Date/time values.
xsd:double	Double values.
xsd:int	Integer values.
xsd:string	Character strings.

PicklistEntry

In the Field object associated with the DescribeSObjectResult, the picklistValues field contains an array of PicklistEntry properties. Each PicklistEntry can contain any one of the following string values. For more information, see Picklist Field Type.

Name	Type	Description
active	boolean	Indicates whether this item must be displayed (true) or not (false) in the drop-down list for the picklist field in the user interface.
validFor	byte[]	A set of bits where each bit indicates a controlling value for which this PicklistEntry is valid. See About Dependent Picklists.
defaultValue	boolean	Indicates whether this item is the default item (true) in the picklist or not (false). Only one item in a picklist can be designated as the default.
label	string	Display name of this item in the picklist.
value	string	Value of this item in the picklist.

About Dependent Picklists

A dependent picklist works in conjunction with a controlling field to filter its values. The value chosen in the controlling field affects the values available in the dependent picklist.

A dependent picklist can be any custom picklist or multi-select picklist field that displays available values based on the value selected in its corresponding controlling field. A controlling field can be any standard or custom picklist (with at least one and less than 200 values) or checkbox field whose values control the available values in one or more corresponding dependent fields.

In the following example, the controlling picklist `Beverage` has two values, which relate to the values of the dependent picklist `Beverage Variety`:

Beverage	Beverage Variety
Coffee	Decaffeinated
	Regular
Tea	Chamomile
	Earl Grey
	English Breakfast

For each `PicklistEntry` that represents a value in a dependent picklist, the `validFor` attribute contains a set of bits. Each bit indicates a controlling field value for which the `PicklistEntry` is valid. Read the bits from left to right.

For more information on dependent picklists, see the “About Dependent Picklists” topic in the Salesforce online help.

Sample Java Code for Dependent Picklists

```
public void dependentPicklistSample() {
  // inner class to decode a "validFor" bitset
  class Bitset {
    byte[] data;

    public Bitset(byte[] data) {
      this.data = data == null ? new byte[0] : data;
    }

    public boolean testBit(int n) {
      return (data[n >> 3] & (0x80 >> n % 8)) != 0;
    }

    public int size() {
```

```
      return data.length * 8;
    }
  }

  try {
    DescribeSObjectResult describeSObjectResult = connection.describeSObject("Case");
    Field[] fields = describeSObjectResult.getFields();
    // create a map of all fields for later lookup
    Map fieldMap = new HashMap();
    for (int i = 0; i < fields.length; i++) {
      fieldMap.put(fields[i].getName(), fields[i]);
    }
    for (int i = 0; i < fields.length; i++) {
      // check whether this is a dependent picklist
      if (fields[i].getDependentPicklist()) {
        // get the controller by name
        Field controller = (Field)fieldMap.get(fields[i].getControllerName());
        System.out.println("Field '" + fields[i].getLabel() + "' depends on '" +
        controller.getLabel() + "'");
        PicklistEntry[] picklistValues = fields[i].getPicklistValues();
        for (int j = 0; j < picklistValues.length; j++) {
          // for each PicklistEntry: list all controlling values for which it is valid
          System.out.println("Item: '" + picklistValues[j].getLabel() +
          "' is valid for: ");
          Bitset validFor = new Bitset(picklistValues[j].getValidFor());
          if (FieldType.picklist == controller.getType()) {
            // if the controller is a picklist, list all
            // controlling values for which this entry is valid
            for (int k = 0; k < validFor.size(); k++) {
              if (validFor.testBit(k)) {
                // if bit k is set, this entry is valid for the
                // for the controlling entry at index k
                System.out.println(controller.getPicklistValues()[k].getLabel());
              }
            }
          } else if (FieldType._boolean == controller.getType()) {
            // the controller is a checkbox
            // if bit 1 is set this entry is valid if the controller is checked
            if (validFor.testBit(1)) {
              System.out.println(" checked");
            }
            // if bit 0 is set this entry is valid if the controller is not checked
            if (validFor.testBit(0)) {
              System.out.println(" unchecked");
            }
          }
        }
      }
    }
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

RecordTypeInfo

Base class for the old RecordTypeMapping object. This object contains all of the existing fields of RecordTypeMapping except layoutId and picklistForRecordType.

Name	Type	Description
`available`	boolean	Indicates whether this record type is available (`true`) or not (`false`). Availability is used to display a list of available record types to the user when they are creating a new record.

Name	Type	Description
defaultRecordTypeMapping	boolean	Indicates whether this is the default record type mapping (true) or not (false).
name	string	Name of this record type.
recordTypeId	ID	ID of this record type.

NamedLayoutInfo

The name of the named layout.

Name	Type	Description
name	string	Name of this layout.

describeSoftphoneLayout()

Retrieves layout information for a Salesforce CRM Call Center SoftPhone.

Syntax

```
DescribeSoftphoneLayoutResult[] = connection.describeSoftphoneLayout();
```

Usage

Use this call to obtain information about the layout of a SoftPhone. Use only in the context of Salesforce CRM Call Center; do not call directly from client programs.

Arguments

This call does not take any objects.

Response

The response is a DescribeSoftphoneLayoutResult object:

Name	Type	Description
callTypes	DescribeSoftphoneLayoutCallType[]	A set of attributes associated with each allowed call type. A call type may be Inbound, Outbound, or Internal.
id	ID	ID of layout. Note that layout objects are not exposed via the API.
name	string	Name of the call type: Inbound, Outbound, or Internal.

DescribeSoftphoneLayoutCallType

Each DescribeSoftphoneLayoutResult object contains one or more call types:

Name	Type	Description
infoFields	DescribeSoftphoneLayoutInfoField[]	A set of information field in the SoftPhone layout.
name	string	Name of the layout.

Name	Type	Description
screenPopOptions	DescribeSoftphoneScreenPopOption[]	Settings in the SoftPhone layout that specify how to display screen pops when the details of calls match or don't match existing records. This setting only displays for SoftPhone layouts associated with CTI 2.0 adapters or higher. See "Salesforce CTI Toolkit Overview" in the Salesforce online help. This field is available in API version 18.0 and later.
screenPopsOpenWithin	string	Setting in the SoftPhone layout that specify whether to display screen pops in a new browser window or tab when the details of calls match or don't match existing records. This setting only displays for SoftPhone layouts associated with CTI 2.0 adapters or higher. See "Salesforce CTI Toolkit Overview" in the Salesforce online help. This field is available in API version 18.0 and later.
sections	DescribeSoftphoneLayoutSection[]	A set of object names and the corresponding item name in the SoftPhone layout. There is one section for each object in a call type.

DescribeSoftphoneLayoutInfoField

An information field in the SoftPhone layout.

Name	Type	Description
name	string	The name of an information field in the SoftPhone layout that does not correspond to a Salesforce object. For example, caller ID may be specified in an information field. Information fields hold static information about the call type.

DescribeSoftphoneLayoutSection

Each call type returned in a DescribeSoftphoneLayoutResult object contains one section for each call type. Each section contains object-item pairs:

Name	Type	Description
entityApiName	string	The name of an object in the Salesforce application that corresponds to an item displayed in the SoftPhone layout, for example, a set of accounts or cases.
items	DescribeSoftphoneLayoutItem[]	A set of SoftPhone layout items.

DescribeSoftphoneLayoutItem

Each layout item corresponds to a record in Salesforce:

Name	Type	Description
itemApiName	string	The name of a record in the Salesforce application that corresponds to an item displayed in the SoftPhone layout, for example, the Acme account.

DescribeSoftphoneScreenPopOption

Each call type returned in a DescribeSoftphoneLayoutResult object contains one `screenPopOptions` field for each call type. Each `screenPopOptions` field contains details about screen pop settings:

Name	Type	Description
`matchType`	string	Setting on a SoftPhone layout to pop a screen for call details that match a single record, multiple records, or no records.
`screenPopData`	string	Setting on a SoftPhone layout for a specific object or page to pop for a call's `matchType`. For example, pop a specified Visualforce page when the details of a call match a record.
`screenPopType`	picklist	Setting that specifies how to pop a screen for a call's `matchType`. For example, pop a detail page or don't pop any page when the details of a call match a record.

Sample Code—Java

This sample describes the soft phone layout and writes its properties to the console. It then gets the allowed call types. For each call type, it gets its information fields, layout sections, and the layout items in the layout sections. It writes these values to the console.

```
public void describeSoftphoneLayout() {
  try {
    DescribeSoftphoneLayoutResult result =
        connection.describeSoftphoneLayout();
    System.out.println("ID of retrieved Softphone layout: " +
        result.getId());
    System.out.println("Name of retrieved Softphone layout: " +
        result.getName());
    System.out.println("\nContains following " +
        "Call Type Layouts\n");
    for (DescribeSoftphoneLayoutCallType type :
          result.getCallTypes()) {
      System.out.println("Layout for " + type.getName() +
          " calls");
      System.out.println("\tCall-related fields:");
      for (DescribeSoftphoneLayoutInfoField field :
            type.getInfoFields()) {
        System.out.println("\t\t{" + field.getName());
      }
      System.out.println("\tDisplayed Objects:");
      for (DescribeSoftphoneLayoutSection section :
            type.getSections()) {
        System.out.println("\t\tFor entity " +
            section.getEntityApiName() +
            " following records are displayed:"
        );
        for (DescribeSoftphoneLayoutItem item :
              section.getItems()) {
          System.out.println("\t\t\t" + item.getItemApiName());
        }
      }
    }
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

Sample Code—C#

This sample describes the soft phone layout and writes its properties to the console. It then gets the allowed call types. For each call type, it gets its information fields, layout sections, and the layout items in the layout sections. It writes these values to the console.

```
/// Demonstrates how to retrieve the layout information
/// for a Salesforce CRM Call Center SoftPhone
public void DescribeSoftphoneLayoutSample()
{
   try
   {
       DescribeSoftphoneLayoutResult dsplResult = binding.describeSoftphoneLayout();

       // Display the ID and Name of the layout
       Console.WriteLine("ID of retrieved Softphone layout: {0}", dsplResult.id);
       Console.WriteLine("Name of retrieved Softphone layout: {0}", dsplResult.name);

       // Display the contents of each Call Type
       Console.WriteLine("\nContains following Call Type Layouts\n");
       foreach (DescribeSoftphoneLayoutCallType dsplCallType in dsplResult.callTypes)
       {
           Console.WriteLine("Layout for {0} calls", dsplCallType.name);

           // Display the call-related fields contained in the call type
           Console.WriteLine("\tCall-related fields:");
           foreach (DescribeSoftphoneLayoutInfoField dsplInfoField
               in dsplCallType.infoFields)
           {
            Console.WriteLine("\t\t{0}", dsplInfoField.name);
           }

           // Display the objects that are included in the layout
           Console.WriteLine("\tDisplayed Objects:");
           foreach (DescribeSoftphoneLayoutSection dsplSection
               in dsplCallType.sections)
           {
               Console.WriteLine("\t\tFor entity {0} following records are displayed:",
                   dsplSection.entityApiName);
               foreach (DescribeSoftphoneLayoutItem dsplItem in dsplSection.items)
               {
                   Console.WriteLine("\t\t\t{0}", dsplItem.itemApiName);
               }
            }
       }
   }
   catch (SoapException e)
   {
       Console.WriteLine(e.Message);
       Console.WriteLine(e.StackTrace);
       Console.WriteLine(e.InnerException);
   }
}
```

`describeTabs()`

Returns information about the standard and custom apps available to the logged-in user, as listed in the Force.com app menu at the top of the page. An app is a set of tabs that works as a unit to provide application functionality. For example, two of the standard Salesforce apps are “Sales” and “Call Center.”

Syntax

```
describeTabSetResult []  = connection.describeTabs();
```

Usage

Use the `describeTabs()` call to obtain information about the standard and custom apps to which the logged-in user has access. The `describeTabs()` call returns the minimum required metadata that can be used to render apps in another user interface. Typically this call is used by partner applications to render Salesforce data in another user interface.

For each app, the call returns the app name, the URL of the logo, whether or not it's the currently selected application for the user, and details about the tabs included in that app.

Important: The `describeTabs()` call returns information only about tabs that display in the Salesforce user interface for the logged-in user. If a user clicks the All Tabs (+) tab and hides some tabs from his Salesforce user interface, those user-hidden tabs aren't included in the set of tabs returned by `describeTabs()`.

Use the `describeAllTabs()` call to obtain information about all the tabs that are available to the logged-in user.

For each tab, the call returns the tab name, the primary sObject that's displayed on the tab, whether it's a custom tab, and the URL for viewing that tab. Note that the "All Tabs" tab and Flexible Page tabs aren't included in the list of tabs.

Sample Code—Java

This sample calls `describeTabs()`, which returns an array of tab set results. Next, for each tab set result, which represents an app, it retrieves some of its properties and gets all the tabs for this app. It writes all retrieved properties to the console.

```
public void describeTabsSample() {
  try {
    // Describe tabs
    DescribeTabSetResult[] dtsrs = connection.describeTabs();
    System.out.println("There are " + dtsrs.length +
        " tab sets defined.");

    // For each tab set describe result, get some properties
    for (int i = 0; i < dtsrs.length; i++) {
      System.out.println("Tab Set " + (i + 1) + ":");
      DescribeTabSetResult dtsr = dtsrs[i];
      System.out.println("Label: " + dtsr.getLabel());
      System.out.println("\tLogo URL: " + dtsr.getLogoUrl());
      System.out.println("\tTab selected: " +
          dtsr.isSelected());

      // Describe the tabs for the tab set
      DescribeTab[] tabs = dtsr.getTabs();
      System.out.println("\tTabs defined: " + tabs.length);

      // Iterate through the returned tabs
      for (int j = 0; j < tabs.length; j++) {
        DescribeTab tab = tabs[j];
        System.out.println("\tTab " + (j + 1) + ":");
        System.out.println("\t\tName: " +
            tab.getSobjectName());
        System.out.println("\t\tLabel: " + tab.getLabel());
        System.out.println("\t\tURL: " + tab.getUrl());
        DescribeColor[] tabColors = tab.getColors();
        // Iterate through tab colors as needed
        DescribeIcon[] tabIcons = tab.getIcons();
        // Iterate through tab icons as needed
      }
    }
  } catch (ConnectionException ce) {
    ce.printStackTrace();
```

```
    }
}
```

Sample Code—C#

This sample calls `describeTabs()`, which returns an array of tab set results. Next, for each tab set result, which represents an app, it retrieves some of its properties and gets all the tabs for this app. It writes all retrieved properties to the console.

```
public void describeTabsSample() {
    try {
      // Describe tabs
      DescribeTabSetResult[] dtsrs = binding.describeTabs();
      Console.WriteLine("There are " + dtsrs.Length +
          " tab sets defined.");

      // For each tab set describe result, get some properties
      for (int i = 0; i < dtsrs.Length; i++) {
          Console.WriteLine("Tab Set " + (i + 1) + ":");
          DescribeTabSetResult dtsr = dtsrs[i];
          Console.WriteLine("Label: " + dtsr.label);
          Console.WriteLine("\tLogo URL: " + dtsr.logoUrl);
          Console.WriteLine("\tTab selected: " +
              dtsr.selected);

          // Describe the tabs for the tab set
          DescribeTab[] tabs = dtsr.tabs;
          Console.WriteLine("\tTabs defined: " + tabs.Length);

          // Iterate through the returned tabs
          for (int j = 0; j < tabs.Length; j++) {
              DescribeTab tab = tabs[j];
              Console.WriteLine("\tTab " + (j + 1) + ":");
              Console.WriteLine("\t\tName: " +
                  tab.sobjectName);
              Console.WriteLine("\t\tLabel: " + tab.label);
              Console.WriteLine("\t\tURL: " + tab.url);
              DescribeColor[] tabColors = tab.colors;
              // Iterate through tab colors as needed
              DescribeIcon[] tabIcons = tab.icons;
              // Iterate through tab icons as needed
          }
      }
    } catch (SoapException e) {
        Console.WriteLine("An unexpected error has occurred: " +
                  e.Message + "\n" + e.StackTrace);
    }
}
```

Arguments

None.

Response

describeTabSetResult, DescribeTab

See Also:

API Call Basics
Using the Partner WSDL
DescribeTab
describeTabSetResult

describeTabSetResult

The `describeTabs()` call returns an array of `DescribeTabSetResult` objects, which has the following properties:

Name	Type	Description
`label`	string	The display label for this standard or custom app. This value changes when tabs are renamed in the Salesforce user interface. See the Salesforce online help for more information.
`logoUrl`	string	A fully qualified URL to the logo image associated with the standard or custom app.
`namespace`	string	If this is a custom app, and a set of tabs in the custom app was installed as part of a managed package, the value of this attribute is the developer namespace prefix that the creator of the package chose when the Developer Edition organization was enabled to allow publishing a managed package. This attribute identifies elements of a Force.com AppExchange package.
`selected`	boolean	If `true`, then this standard or custom app is the user's currently selected app.
`tabs`	DescribeTab	An array of tabs that are displayed for the specified standard app or custom app.

DescribeColor

DescribeColor contains color metadata information for a tab. The `describeTabs()` call returns an array of DescribeTabSetResult values. Each DescribeTabSetResult contains an array of DescribeTab values, and each DescribeTab contains an array of DescribeColor values.

Each DescribeColor is associated with a theme. See Identifying the Salesforce Style Your Users See in the Visualforce Developer's Guide for more details on Salesforce themes.

Color information can also be retrieved via the `describeTheme()` and `describeGlobalTheme()` calls. These calls return information on colors used for each object in your organization that can use theme icons and colors.

Name	Type	Description
`color`	string	The color described in Web color RGB format, for example, "00FF00".
`context`	string	The color context, which determines whether the color is the main color ("primary") for the tab, or not.
`theme`	string	The associated theme. Possible values include: • "theme2": Salesforce theme used prior to Spring '10 • "theme3": The current Salesforce theme, introduced in Spring '10 • "theme4": Theme introduced in Winter '14 for the mobile touchscreen version of Salesforce • "custom": Theme associated with a custom icon

DescribeIcon

DescribeIcon contains icon metadata information for a tab. The `describeTabs()` call returns an array of DescribeTabSetResult values. Each DescribeTabSetResult contains an array of DescribeTab values, and each DescribeTab contains an array of DescribeIcon values.

Icon information can also be retrieved via the `describeTheme()` and `describeGlobalTheme()` calls. These calls return information on icons used for each object in your organization that can use theme icons and colors.

Name	Type	Description
contentType	string	The tab icon's content type, for example, "image/png."
height	int	The tab icon's height in pixels. If the icon content type is an SVG type, height and width values are not used.
theme	string	The associated theme. Possible values include: • "theme2": Salesforce theme used prior to Spring '10 • "theme3": The current Salesforce theme, introduced in Spring '10 • "theme4": Theme introduced in Winter '14 for the mobile touchscreen version of Salesforce • "custom": Theme associated with a custom icon
url	string	The fully qualified URL for this icon.
width	int	The tab icon's width in pixels. If the icon content type is an SVG type, height and width values are not used.

DescribeTab

The `describeTabs()` call returns a describeTabSetResult object, of which DescribeTab is a property:

Name	Type	Description
colors	DescribeColor[]	Array of color information used for a tab. This field is available in API version 29.0 and later.
custom	boolean	`true` if this is a custom tab, `false` if this is a standard tab.
iconUrl	string	The URL for the main 32 x 32 pixel icon for a tab. This icon appears next to the heading at the top of most pages. This icon URL corresponds to the 32 x 32 icon used for the current Salesforce theme, introduced in Spring '10.
icons	DescribeIcon[]	Array of icon information used for a tab. This field is available in API version 29.0 and later.
label	string	The display label for this tab.
miniIconUrl	string	The URL for the 16 x 16 pixel icon that represents a tab. This icon appears in related lists and other locations. This icon URL corresponds to the 16x16 icon used for the current Salesforce theme, introduced in Spring '10.
name	string	The API name of the tab.
sobjectName	string	The name of the sObject that is primarily displayed on this tab (for tabs that display a particular SObject). For a list of objects, see Standard Objects.
url	string	A fully qualified URL for viewing this tab.

See Also:

DescribeColor
DescribeIcon

describeTheme()

Returns information about themes available to the current logged-in user.

Syntax

```
DescribeThemeResult = connection.describeTheme(string sObjectType[]);
```

Usage

Use `describeTheme()` to get current theme information for a given array of objects. Theme information consists of colors and icons for an object in Salesforce, used for a particular theme. For example, the Merchandise__c object might use the "computer32" icon and a primary tab color of red for the regular Salesforce application theme, and a different set of colors and icons for the mobile touchscreen version of Salesforce.

If you pass `null` instead of an array of objects, `describeTheme()` returns theme information for all objects in your organization that use theme colors and icons.

Your client application must be logged in with sufficient access rights to retrieve theme information about your organization's data. For more information, see Factors that Affect Data Access.

`describeTheme()` is available in API version 29.0 and later.

Sample

This Java sample calls `describeTheme()` to retrieve theme information for Account and Contact, and then iterates over the retrieved theme information.

```
public static void describeThemeExample() {
    try {
        // Get current themes
        DescribeTheme themeResult = connection.describeTheme(
                                        new String[] { "Account", "Contact" });
        DescribeThemeItem[] themeItems = themeResult.getThemeItems();
        for (int i = 0; i < themeItems.length; i++) {
            DescribeThemeItem themeItem = themeItems[i];
            System.out.println("Theme information for object " + themeItem.getName());
            // Get color and icon info for each themeItem
            DescribeColor colors[] = themeItem.getColors();
            System.out.println("    Number of colors: " + colors.length);
            int k;
            for (k = 0; k < colors.length; k++) {
                DescribeColor color = colors[k];
                System.out.println("    For Color #" + k + ":");
                System.out.println("      Web RGB Color: " + color.getColor());
                System.out.println("      Context: " + color.getContext());
                System.out.println("      Theme: " + color.getTheme());
            }
            DescribeIcon icons[] = themeItem.getIcons();
            System.out.println("    Number of icons: " + icons.length);
            for (k = 0; k < icons.length; k++) {
                DescribeIcon icon = icons[k];
                System.out.println("    For Icon #" + k + ":");
                System.out.println("      ContentType: " + icon.getContentType());
                System.out.println("      Height: " + icon.getHeight());
                System.out.println("      Theme: " + icon.getTheme());
                System.out.println("      URL: " + icon.getUrl());
                System.out.println("      Width: " + icon.getWidth());
            }
        }
    } catch (ConnectionException ce) {
        ce.printStackTrace();
```

```
    }
}
```

Response

DescribeThemeResult

Faults

`UnexpectedErrorFault`

See Also:

DescribeThemeResult
DescribeThemeItem
DescribeColor
DescribeIcon

DescribeThemeResult

The `describeTheme()` and `describeGlobalTheme()` calls return DescribeThemeResult, which contains an array of DescribeThemeItem values.

Name	Type	Description
`themes`	DescribeThemeItem[]	Array of themes. Theme information is provided for each object in the organization that can use theme icons and colors.

DescribeThemeItem

The `describeTheme()` and `describeGlobalTheme()` calls return DescribeThemeResult, which contains an array of DescribeThemeItem values. Each DescribeThemeItem contains an array of colors and icons used for themes, and the name of the object the theme information applies to.

Name	Type	Description
`colors`	DescribeColor[]	Array of colors.
`icons`	DescribeIcon[]	Array of icons.
`name`	string	Name of the object that the theme colors and icons are associated with.

Chapter 11

Utility Calls

This topic describes API calls that your client applications can invoke to obtain the system timestamp, user information, and change user passwords.

Note: For a list of general API calls, see Core Calls. For a list of describe calls, see Describe Calls.

The following table lists the API utility calls described in this topic:

Task / Call	Description
`getServerTimestamp()`	Retrieves the current system timestamp from the API.
`getUserInfo()`	Retrieves personal information for the user associated with the current session.
`resetPassword()`	Changes a user's password to a system-generated value.
`sendEmail()`	Immediately sends an email message.
`sendEmailMessage()`	Immediately sends up to 10 draft email messages.
`setPassword()`	Sets the specified user's password to the specified value.

Samples

The samples in this section are based on the enterprise WSDL file. They assume that you have already imported the WSDL file and created a connection. To learn how to do so, see the Quick Start tutorial.

getServerTimestamp()

Retrieves the current system timestamp (Coordinated Universal Time (UTC) time zone) from the API.

Syntax

```
GetServerTimestampResult timestamp = connection.getServerTimestamp();
```

Usage

Use `getServerTimestamp()` to obtain the current system timestamp from the API. You might do this if, for example, you need to use the exact timestamp for timing or data synchronization purposes. When you `create()` or `update()` an object, the API uses the system timestamp to update the `CreatedDate` and `LastModifiedDate` fields, respectively, in the object.

The `getServerTimestamp()` call always returns the timestamp in Coordinated Universal Time (UTC) time zone. However, your local system might automatically display the results in your local time based on your time zone settings.

Note: Development tools differ in the way that they handle time data. Some development tools report the local time, while others report only the Coordinated Universal Time (UTC) time zone. To determine how your development tool handles time values, refer to its documentation.

Sample Code—Java

This sample gets the server time and writes it to the console in the user's local time zone.

```
public void doGetServerTimestamp() {
   try {
      GetServerTimestampResult result = connection.getServerTimestamp();
      Calendar serverTime = result.getTimestamp();
      System.out.println("Server time is: "
            + serverTime.getTime().toString());
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample gets the server time and writes it to the console in the user's local time zone.

```
public void doGetServerTimestamp()
{
   try
   {
      GetServerTimestampResult result =
            binding.getServerTimestamp();
      DateTime serverTime = result.timestamp;
      Console.WriteLine("Server time is: " +
            serverTime.ToLocalTime().ToString());
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                        e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

None.

Response

getServerTimestampResult

Fault

`UnexpectedErrorFault`

See Also:

Utility Calls

https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

getServerTimestampResult

The getServerTimestamp() call returns a GetServerTimestampResult object, which has the following properties:

Name	Type	Description
timestamp	dateTime	System timestamp of the API when the getServerTimestamp() call was executed.

getUserInfo()

Retrieves personal information for the user associated with the current session.

Syntax

```
getUserInfoResult result = connection.getUserInfo();
```

Usage

Use getUserInfo() to obtain personal information about the currently logged-in user. This convenience API call retrieves and aggregates common profile information that your client application can use for display purposes, performing currency calculations, and so on.

The getUserInfo() call applies only to the username under which your client application has logged in. To retrieve additional personal information not found in the getUserInfoResult object, you can call retrieve() on the User object and pass in the userID returned by this call. To retrieve personal information about other users, you could call retrieve() (if you know their user ID) or query() on the User object.

Sample Code—Java

This sample calls getUserInfo() and writes information about the current user to the console.

```
public void doGetUserInfo() {
   try {
      GetUserInfoResult result = connection.getUserInfo();
      System.out.println("\nUser Information");
      System.out.println("\tFull name: " + result.getUserFullName());
      System.out.println("\tEmail: " + result.getUserEmail());
      System.out.println("\tLocale: " + result.getUserLocale());
      System.out.println("\tTimezone: " + result.getUserTimeZone());
      System.out.println("\tCurrency symbol: " + result.getCurrencySymbol());
      System.out.println("\tOrganization is multi-currency: " +
            result.isOrganizationMultiCurrency());
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample calls getUserInfo() and writes information about the current user to the console.

```
public void doGetUserInfo()
{
   try
   {
```

```
        GetUserInfoResult result = binding.getUserInfo();
        Console.WriteLine("\nUser Information");
        Console.WriteLine("\tFull name: " + result.userFullName);
        Console.WriteLine("\tEmail: " + result.userEmail);
        Console.WriteLine("\tLocale: " + result.userLocale);
        Console.WriteLine("\tTimezone: " + result.userTimeZone);
        Console.WriteLine("\tCurrency symbol: " + result.currencySymbol);
        Console.WriteLine("\tOrganization is multi-currency: " +
              result.organizationMultiCurrency);
    }
    catch (SoapException e)
    {
        Console.WriteLine("An unexpected error has occurred: " +
                          e.Message + "\n" + e.StackTrace);
    }
}
```

Arguments

None.

Response

getUserInfoResult

Fault

`UnexpectedErrorFault`

See Also:

Utility Calls

https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

getUserInfoResult

The `getUserInfo()` call returns a `GetUserInfoResult` object, which has the following properties:

Name	Type	Description
`accessibilityMode`	boolean	Available in API version 7.0 and later. Indicates whether user interface modifications for the visually impaired are on (`true`) or off (`false`). The modifications facilitate the use of screen readers such as JAWS.
`currencySymbol`	string	Currency symbol to use for displaying currency values. Applicable only when `organizationMultiCurrency` is `false`.
`organizationId`	ID	ID of the organization. Allows third-party tools to uniquely identify individual organizations in Salesforce, which is useful for retrieving billing or organization-wide setup information.
`organizationMultiCurrency`	boolean	Indicates whether the user's organization uses multiple currencies (`true`) or not (`false`).
`organizationName`	string	Name of the user's organization or company.
`orgDefaultCurrencyIsoCode`	string	Default currency ISO code. Applicable only when `organizationMultiCurrency` is `false`. When the logged-in user

Name	Type	Description
		creates any objects that have a currency ISO code, the API uses this currency ISO code if it is not explicitly specified in the `create()` call.
`profileID`	ID	ID of the profile associated with the role currently assigned to the user.
`roleID`	ID	Role ID of the role currently assigned to the user.
`sessionSecondsValid`	int	Number of seconds remaining until the user's session expires. Available in API version 21.0 and later.
`userDefaultCurrencyIsoCode`	string	Default currency ISO code. Applicable only when `organizationMultiCurrency` is `true`. When the logged-in user creates any objects that have a currency ISO code, the API uses this currency ISO code if it is not explicitly specified in the `create()` call.
`userEmail`	string	User's email address.
`userFullName`	string	User's full name.
`userID`	ID	User ID.
`userLanguage`	string	User's language, which controls the language for labels displayed in an application. String is 2-5 characters long. The first two characters are always an ISO language code, for example "fr" or "en." If the value is further qualified by country, then the string also has an underscore (_) and another ISO country code, for example "US" or "UK. For example, the string for the United States is "en_US", and the string for French Canadian is "fr_CA." For a list of the languages that Salesforce supports, see the Salesforce online help topic "What languages does Salesforce support?"
`userLocale`	string	User's locale, which controls the formatting of dates and choice of symbols for currency. The first two characters are always an ISO language code, for example "fr" or "en." If the value is further qualified by country, then the string also has an underscore (_) and another ISO country code, for example "US" or "UK. For example, the string for the United States is "en_US", and the string for French Canadian is "fr_CA."
`userName`	string	User's login name.
`userTimeZone`	string	User's time zone.
`userType`	string	Type of user license assigned to the Profile associated with the user.
`userUiSkin`	string	Available in API version 7.0 and later. Returns the value `Theme2` if the user is using the newer user interface theme of the online application, labeled "Salesforce." Returns `Theme1` if the user is using the older user interface theme, labeled "Salesforce Classic." In the online application, this look and feel setting is configurable from Setup by clicking **Customize** > **User Interface**. See User Interface Themes.

resetPassword()

Changes a user's password to a temporary, system-generated value.

Syntax

```
string password = connection.resetPassword(ID userID);
```

Usage

Use `resetPassword()` to request that the API change the password of a User or SelfServiceUser, and return a system-generated password string of random letters and numbers. Use `setPassword()` instead if you want to set the password to a specific value.

Your client application must be logged in with sufficient access rights to change the password for the specified user. For more information, see Factors that Affect Data Access.

For information on IDs, see ID Field Type.

Sample Code—Java

This sample resets the password for the user specified by the `userId` parameter. It calls `resetPassword()` with this ID and gets the temporary password from the call result. It writes this temporary password to the console and returns it.

```
public String doResetPassword(String userId) {
   String result = "";
   try {
      ResetPasswordResult rpr = connection.resetPassword(userId);
      result = rpr.getPassword();
      System.out.println("The temporary password for user ID " + userId
            + " is " + result);
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
   return result;
}
```

Sample Code—C#

This sample resets the password for the user specified by the `userId` parameter. It calls `resetPassword()` with this ID and gets the temporary password from the call result. It writes this temporary password to the console and returns it.

```
public String doResetPassword(String userId)
{
   String result = "";
   try
   {
      ResetPasswordResult rpr = binding.resetPassword(userId);
      result = rpr.password;
      Console.WriteLine("The temporary password for user ID " + userId + " is " +
            result);
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                        e.Message + "\n" + e.StackTrace);
   }
   return result;
}
```

Arguments

Name	Type	Description
`userID`	ID	ID of the User or SelfServiceUser whose password you want to reset. For information on IDs, see ID Field Type.

Response

Name	Type	Description
`password`	string	New password generated by the API. Once the user logs in with this password, they will be asked to provide a new password. This password is temporary, meaning that it cannot be reused once the user has set his or her new password.

Fault

`InvalidIdFault`

`UnexpectedErrorFault`

See Also:

Utility Calls
https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

`sendEmail()`

Immediately sends an email message.

Syntax

For single email messages:

```
SendEmailResult = connection.sendEmail(SingleEmailMessage emails[]);
```

For mass email messages:

```
SendEmailResult = connection.sendEmail(MassEmailMessage emails[]);
```

Usage

Use this call with Force.com AppExchange applications, custom applications, or other applications outside of Salesforce to send individual and mass email. The email can include all standard email attributes (such as subject line and blind carbon copy address), use Salesforce email templates, and be in plain text or HTML format. You can use Salesforce to track the status of HTML email, including the date the email was sent, first opened, last opened, and the total number of times it was opened. (See “Tracking HTML Email” in the Salesforce online help for more information.)

The email address of the logged-in user is inserted in the `From Address` field of the email header. All return email and out-of-office replies go to the logged-in user. If bounce management is enabled and `targetObjectId` or `targetObjectIds` is set, bounces are processed by Salesforce automatically, and the appropriate records are updated; otherwise, they go to the logged-in user.

Note:

- Single email messages sent with this call count against the sending organization's daily single email limit. When this limit is reached, `sendEmail()` calls using SingleEmailMessage are rejected, and the user receives a `SINGLE_EMAIL_LIMIT_EXCEEDED` error code. However, single emails sent through the application are allowed.
- Mass email messages sent with this call count against the sending organization's daily mass email limit. When this limit is reached, `sendEmail()` calls using MassEmailMessage are rejected, and the user receives a `MASS_MAIL_LIMIT_EXCEEDED` error code.
- If you use this call to send email to one or more recipients who have the **Email Opt Out** option selected, the call returns `UNKNOWN_EXCEPTION`. In such cases, do not report the error. Instead, either remove the recipient or have the administrator deselect the option for that recipient.

SingleEmailMessage has an optional field called `OrgWideEmailAddressId`. This is an object ID to an OrgWideEmailAddress object. If `OrgWideEmailAddressId` is set, the OrgWideEmailAddress.`DisplayName` field is used in the email header, instead of the logged-in user's `Display Name`. The sending email address in the header is also set to the field defined in OrgWideEmailAddress.`Address`.

Note: If both the `DisplayName` in an OrgWideEmailAddress and `senderDisplayName` are defined, the user receives a `DUPLICATE_SENDER_DISPLAY_NAME` error.

Sample Code—Java

This sample creates an email message and sets its fields, including the To, CC and BCC recipients, subject, and body text. It also sets a recipient to the ID of the logged-in user using the `setTargetObjectId` method, which causes the email to be sent to the email address of the specified user. The sample creates an attachment and sends the email message with the attachment. Finally, it writes a status message or an error message, if any, to the console.

```
public void doSendEmail() {
   try {
      EmailFileAttachment efa = new EmailFileAttachment();
      byte[] fileBody = new byte[1000000];
      efa.setBody(fileBody);
      efa.setFileName("attachment");
      SingleEmailMessage message = new SingleEmailMessage();
      message.setBccAddresses(new String[] {
         "someone@salesforce.com"
      });
      message.setCcAddresses(new String[] {
         "person1@salesforce.com", "person2@salesforce.com"
      });
      message.setBccSender(true);
      message.setEmailPriority(EmailPriority.High);
      message.setReplyTo("person1@salesforce.com");
      message.setSaveAsActivity(false);
      message.setSubject("This is how you use the " + "sendEmail method.");
      // We can also just use an id for an implicit to address
      GetUserInfoResult guir = connection.getUserInfo();
      message.setTargetObjectId(guir.getUserId());
      message.setUseSignature(true);
      message.setPlainTextBody("This is the humongous body "
            + "of the message.");
      EmailFileAttachment[] efas = { efa };
      message.setFileAttachments(efas);
      message.setToAddresses(new String[] { "person3@salesforce.com" });
      SingleEmailMessage[] messages = { message };
      SendEmailResult[] results = connection.sendEmail(messages);
      if (results[0].isSuccess()) {
         System.out.println("The email was sent successfully.");
      } else {
         System.out.println("The email failed to send: "
```

```
                 + results[0].getErrors()[0].getMessage());
         }
   } catch (ConnectionException ce) {
         ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample creates an email message and sets its fields, including the To, CC and BCC recipients, subject, and body text. It also sets a recipient to the ID of the logged-in user using the `setTargetObjectId` method, which causes the email to be sent to the email address of the specified user. The sample creates an attachment and sends the email message with the attachment. Finally, it writes a status message or an error message, if any, to the console.

```
public void doSendEmail()
{
   try
   {
      EmailFileAttachment efa = new EmailFileAttachment();
      byte[] fileBody = new byte[1000000];
      efa.body = fileBody;
      efa.fileName = "attachment";
      SingleEmailMessage message = new SingleEmailMessage();
      message.setBccAddresses(new String[] {
         "someone@salesforce.com"
      });
      message.setCcAddresses(new String[] {
         "person1@salesforce.com", "person2@salesforce.com"
      });
      message.bccSender = true;
      message.emailPriority = EmailPriority.High;
      message.replyTo = "person1@salesforce.com";
      message.saveAsActivity = false;
      message.subject = "This is how you use the " + "sendEmail method.";
      // We can also just use an id for an implicit to address
      GetUserInfoResult guir = binding.getUserInfo();
      message.targetObjectId = guir.userId;
      message.useSignature = true;
      message.plainTextBody = "This is the humongous body "
            + "of the message.";
      EmailFileAttachment[] efas = { efa };
      message.fileAttachments = efas;
      message.toAddresses = new String[] { "person3@salesforce.com" };
      SingleEmailMessage[] messages = { message };
      SendEmailResult[] results = binding.sendEmail(messages);
      if (results[0].success)
      {
         Console.WriteLine("The email was sent successfully.");
      }
      else
      {
         Console.WriteLine("The email failed to send: "
               + results[0].errors[0].message);
      }
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                        e.Message + "\n" + e.StackTrace);
   }
}
```

BaseEmail

The following table contains the arguments used in both single and mass email.

Note: If templates are not being used, all email content must be in plain text, HTML, or both.

Name	Type	Description
bccSender	boolean	Indicates whether the email sender receives a copy of the email that is sent. For a mass mail, the sender is only copied on the first email sent. **Note:** If the BCC compliance option is set at the organization level, the user cannot add BCC addresses on standard messages. The following error code is returned: BCC_NOT_ALLOWED_IF_BCC_COMPLIANCE_ENABLED. Contact your salesforce.com representative for information on BCC compliance.
saveAsActivity	boolean	Optional. The default value is true, meaning the email is saved as an activity. This argument only applies if the recipient list is based on targetObjectId or targetObjectIds. If HTML email tracking is enabled for the organization, you will be able to track open rates.
useSignature	boolean	Indicates whether the email includes an email signature if the user has one configured. The default is true, meaning if the user has a signature it is included in the email unless you specify false.
emailPriority	picklist	Optional. The priority of the email. • Highest • High • Normal • Low • Lowest The default is Normal.
replyTo	string	Optional. The email address that receives the message when a recipient replies. This cannot be set if you are using a Visualforce email template that specifies a replyTo value.
subject	string	Optional. The email subject line. If you are using an email template and attempt to override the subject line, an error message is returned.
templateId	ID	The ID of the template to be merged to create this email.
senderDisplayName	string	Optional. The name that appears on the From line of the email. This cannot be set if the object associated with a OrgWideEmailAddressId for a SingleEmailMessage has defined its DisplayName field.

SingleEmailMessage

The following table contains the arguments single email uses in addition to the base email arguments.

Name	Type	Description
bccAddresses	string[]	Optional. An array of blind carbon copy (BCC) addresses. The maximum allowed is 25. This argument is allowed only when a template is not used. If the BCC COMPLIANCE option is set at the organization level, the user cannot add BCC addresses on standard

Name	Type	Description
		messages. The following error code is returned: BCC_NOT_ALLOWED_IF_BCC_COMPLIANCE_ENABLED. All emails must have a recipient value in at least one of the following: • toAddresses • ccAddresses • bccAddresses • targetObjectId • targetObjectIds
ccAddresses	string[]	Optional. An array of carbon copy (CC) addresses. The maximum allowed is 25. This argument is allowed only when a template is not used.
charset	string	Optional. The character set for the email. If this value is null, the user's default value is used. Unavailable if specifying templateId because the template specifies the character set.
documentAttachments	ID[]	Optional. An array listing the ID of each Document you want to attach to the email. You can attach multiple documents as long as the total size of all attachments does not exceed 10 MB.
fileAttachments	EmailFileAttachment[]	Optional. An array listing the file names of the binary and text files you want to attach to the email. You can attach multiple files as long as the total size of all attachments does not exceed 10 MB.
htmlBody	string	Optional. The HTML version of the email, specified by the sender. The value is encoded according to the specification associated with the organization.
inReplyTo	string	Optional. The In-Reply-To field of the outgoing email. Identifies the emails to which this one is a reply (parent emails). Contains the parent emails' Message-IDs. See RFC2822 - Internet Message Format.
orgWideEmailAddressId	ID	Optional. The object ID of the OrgWideEmailAddress associated with the outgoing email. OrgWideEmailAddress.DisplayName cannot be set if the senderDisplayName field is already set.
plainTextBody	string	Optional. The text version of the email, specified by the sender.
references	string	Optional. The References field of the outgoing email. Identifies an email thread. Contains the parent emails' Message-ID and References fields and possibly In-Reply-To fields. See RFC2822 - Internet Message Format.
targetObjectId	ID	Optional. The object ID of the contact, lead, or user the email will be sent to. The object ID you enter sets the context and ensures that merge fields in the template contain the correct data

Name	Type	Description
		Do not enter the object IDs of records that have the `Email Opt Out` option selected. All emails must have a recipient value in at least one of the following: • `toAddresses` • `ccAddresses` • `bccAddresses` • `targetObjectId` • `targetObjectIds`
`toAddresses`	string[]	Optional. An array of email address you are sending the email to. The maximum allowed is 100. This argument is allowed only when a template is not used. All emails must have a recipient value in at least one of the following: • `toAddresses` • `ccAddresses` • `bccAddresses` • `targetObjectId` • `targetObjectIds`
`whatId`	ID	Optional. If you specify a contact for the `targetObjectId` field, you can specify a `whatId` as well. This helps to further ensure that merge fields in the template contain the correct data. The value must be one of the following types: • Account • Asset • Campaign • Case • Contract • Opportunity • Order • Product • Solution • Custom

MassEmailMessage

The following table contains the arguments mass email uses in addition to the base email arguments.

Name	Type	Description
`description`	string	A value used internally to identify the object in the mass email queue.
`targetObjectIds`	ID[]	An array of object IDs of the contacts, leads, or users the email will be sent to. The object IDs you enter set the context and ensure that merge fields in the template contain the correct data. The objects must be of the same type (either all contacts, all leads, or all users). You can list up to 250 IDs per email. If you specify a value for the `targetObjectIds` field, optionally

Name	Type	Description
		specify a whatId as well to set the email context to a user, contact, or lead. This ensures that merge fields in the template contain the correct data. Do not enter the object IDs of records that have the Email Opt Out option selected. All emails must have a recipient value in at least one of the following: • toAddresses • ccAddresses • bccAddresses • targetObjectId • targetObjectIds
whatIds	ID[]	Optional. If you specify an array of contacts for the targetObjectIds field, you can specify an array of whatIds as well. This helps to further ensure that merge fields in the template contain the correct data. The values must be one of the following types: • Contract • Case • Opportunity • Product If you specify whatIds, specify one for each targetObjectId; otherwise, you will receive an INVALID_ID_FIELD error.

EmailFileAttachment

The following table contains properties that the EmailFileAttachment uses in the SingleEmailMessage object to specify attachments passed in as part of the request, as opposed to a Document passed in using the documentAttachments argument.

Property	Type	Description
body	base64	The attachment itself.
contentType	string	Optional. The attachment's Content-Type.
fileName	string	The name of the file to attach.
inline	boolean	Optional. Specifies a Content-Disposition of inline (true) or attachment (false). In most cases, inline content is displayed to the user when the message is opened. Attachment content requires user action to be displayed.

Response

SendEmailResult

Fault

BCC_NOT_ALLOWED_IF_BCC_COMPLIANCE_ENABLED

BCC_SELF_NOT_ALLOWED_IF_BCC_COMPLIANCE_ENABLED

DUPLICATE_SENDER_DISPLAY_NAME

EMAIL_NOT_PROCESSED_DUE_TO_PRIOR_ERROR

ERROR_IN_MAILER

INSUFFICIENT_ACCESS_ON_CROSS_REFERENCE_ENTITY

INVALID_CONTENT_TYPE

INVALID_EMAIL_ADDRESS

INVALID_ID_FIELD

INVALID_MESSAGE_ID_REFERENCE

INVALID_SAVE_AS_ACTIVITY_FLAG

LIMIT_EXCEEDED

MALFORMED_ID

MASS_MAIL_LIMIT_EXCEEDED

NO_MASS_MAIL_PERMISSION

REQUIRED_FIELD_MISSING

SINGLE_EMAIL_LIMIT_EXCEEDED

TEMPLATE_NOT_ACTIVE

UNVERIFIED_SENDER_ADDRESS

SendEmailResult

The `sendEmail()` call returns a list of SendEmailResult objects. Each SendEmailResult object has the following properties:

Name	Type	Description
`success`	boolean	If sending single email: Indicates whether the email was successfully accepted for delivery by the message transfer agent (`true`) or not (`false`). Even if `success` = `true`, it does not mean the intended recipients received the email, as it could have bounced or been blocked by a spam blocker. Also, even if the email is successfully accepted for delivery by the message transfer agent, there can still be errors in the error array related to individual addresses within the email. If sending mass email: Indicates whether the email was successfully added to the queue for processing (`true`) or not (`false`). Even if the email was added to the queue, there can still be processing errors that prevent delivery to the intended recipients.
`SendEmailError`	Error[]	If an error occurred during the `sendEmail()` call, a list of `SendEmailError` objects is returned. For single email, errors indicate that Salesforce wasn't able to deliver the email. For mass email, errors indicate that the email wasn't added to the queue for processing.

SendEmailError

SendEmailError can have the following attributes:

Name	Type	Description
`Fields`	Field[]	Reserved for future use. Array of one or more field names. Identifies which fields in the object, if any, affected the error condition.

Name	Type	Description
Message	string	Error message text.
StatusCode	statusCode	A code that characterizes the error. The full list of status codes is available in the WSDL file for your organization.
TargetObjectId	ID	The object ID of the target for which the error occurred.

Note: If an error occurs that prevents `sendEmail()` from sending the email to one or more targets, each `TargetObjectId` for those targets has an associated error in SendEmailResult. A `TargetObjectId` that does not have an associated error in SendEmailResult indicates the email was sent to the target. If SendEmailResult has an error that does not have an associated `TargetObjectId`, no email was sent.

The following is an example of how to parse through a resulting set for errors:

```
Messaging.SingleEmailMessage email = new Messaging.SingleEmailMessage();
email.setToAddresses(new String[] { 'admin@acme.com' });
email.setSubject('my subject');
email.setPlainTextBody('plain text body');
List<Messaging.SendEmailResult> results =
    Messaging.sendEmail(new Messaging.Email[] { email });
if (!results.get(0).isSuccess()) {
    System.StatusCode statusCode = results.get(0).getErrors()[0].getStatusCode();
    String errorMessage = results.get(0).getErrors()[0].getMessage();
}
```

sendEmailMessage()

Immediately sends up to 10 draft email messages.

Syntax

For Enterprise SOAP:

```
SendEmailResult[] = connection.sendEmailMessage( String[] draftEmailIds);
```

For Partner SOAP:

```
SendEmailResult[] = connection.sendEmailMessage( ID[] draftEmailIds);
```

Usage

Use this call with Force.com AppExchange applications, custom applications, or other applications outside of Salesforce to send up to 10 draft email messages. The messages can include all standard email attributes (such as subject line and blind carbon copy address), use Salesforce email templates, and be in plain text or HTML format. You can use Salesforce to track the status of HTML email, including the date the email was sent, first opened, last opened, and the total number of times it was opened. (See "Tracking HTML Email" in the Salesforce online help for more information.)

The email address of the logged-in user is inserted in the `From Address` field of the email header. All return email and out-of-office replies go to the logged-in user. If bounce management is enabled and `targetObjectId` or `targetObjectIds` is set, bounces are processed by Salesforce automatically, and the appropriate records are updated; otherwise, they go to the logged-in user.

Note:

- Email messages sent with this call count against the sending organization's daily single email limit. When this limit is reached, `sendEmailMessage()` calls using SingleEmailMessage are rejected, and the user receives a `SINGLE_EMAIL_LIMIT_EXCEEDED` error code. However, single emails sent through the application are allowed.
- Mass email messages sent with this call count against the sending organization's daily mass email limit. When this limit is reached, `sendEmail()` calls using MassEmailMessage are rejected, and the user receives a `MASS_MAIL_LIMIT_EXCEEDED` error code.
- The `AllOrNone` header is not honored by this call. `sendEmailMessage()` returns partial success even if the `AllOrNone` header is set to `true`.

Sample Code—Java

This sample creates a case and a draft email message, and sets the message fields, including the `From`, `To`, `CC`, and `BCC` recipients, subject, and body text. It also creates an attachment and sends the email message with the attachment. Finally, it writes a status message or an error message, if any, to the console.

```
public void doSendEmail() {
   try {
      //Create a case
      Case theCase = new Case();
      theCase.setSubject("Sample Case");
      SaveResult[] saveResult = connection.create(new SObject[] { theCase });
      String caseId = saveResult[0].getId();

      //Create a draft EmailMessage
      EmailMessage message = new EmailMessage();
      message.setParentId(theCase.getId());
      message.setBccAddress("bcc@email.com");
      message.setCcAddress("cc1@salesforce.com; cc2@email.com");
      message.setSubject("This is how you use the sendEmailMessage method.");
      message.setFromAddress("from@email.com");
      message.setFromName("Sample Code");
      message.setTextBody("This is the text body of the message.");
      message.setStatus("5"); //"5" means Draft
      message.setToAddress("to@email.com");
      saveResult = connection.create(new SObject[] { message });
      String emailMessageId = saveResult[0].getId();

      //Create an attachment for the draft EmailMessage
      Attachment att = new Attachment();
      byte[] fileBody = new byte[1000000];
      att.setBody(fileBody);
      att.setName("attachment");
      att.setParentId(emailMessageId);
      connection.create(new SObject[] { att });

      //Send the draft EmailMessage
      SendEmailResult[] results = connection.sendEmailMessage(messages);
      if (results[0].isSuccess()) {
         System.out.println("The email was sent successfully.");
      } else {
         System.out.println("The email failed to send: " +
            results[0].getErrors()[0].getMessage());
      }
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
```

Arguments

None.

Response

SendEmailResult[]

Fault

BCC_NOT_ALLOWED_IF_BCC_COMPLIANCE_ENABLED

BCC_SELF_NOT_ALLOWED_IF_BCC_COMPLIANCE_ENABLED

EMAIL_NOT_PROCESSED_DUE_TO_PRIOR_ERROR

ERROR_IN_MAILER

INSUFFICIENT_ACCESS_ON_CROSS_REFERENCE_ENTITY

INVALID_CONTENT_TYPE

INVALID_EMAIL_ADDRESS

INVALID_ID_FIELD

INVALID_MESSAGE_ID_REFERENCE

LIMIT_EXCEEDED

MALFORMED_ID

REQUIRED_FIELD_MISSING

SINGLE_EMAIL_LIMIT_EXCEEDED

TEMPLATE_NOT_ACTIVE

UNVERIFIED_SENDER_ADDRESS

setPassword()

Sets the specified user's password to the specified value.

Syntax

```
SetPasswordResult setPasswordResult = connection.setPassword(ID userID, string password);
```

Usage

Use `setPassword()` to change the password of a User or SelfServiceUser to a value that you specify. For example, a client application might prompt a user to specify a different password, and then invoke `setPassword()` to change the user's password. Use `resetPassword()` instead if you want to reset the password with a random value generated by the API.

Your client application must be logged in with sufficient access rights to change the password for the specified user. For more information, see Factors that Affect Data Access.

For information on IDs, see ID Field Type.

This call can use the session ID returned in LoginResult if the password has expired. For more information, see `passwordExpired`.

Sample Code—Java

This sample accepts user ID and password parameters, which it uses in the `setPassword()` call to set the password of the specified user.

```
public void doSetPassword(String userId, String newPasswd) {
   try {
      SetPasswordResult result = connection.setPassword(userId, newPasswd);
      System.out.println("The password for user ID " + userId + " changed to "
            + newPasswd);
   } catch (ConnectionException ce) {
      ce.printStackTrace();
   }
}
```

Sample Code—C#

This sample accepts user ID and password parameters, which it uses in the `setPassword()` call to set the password of the specified user.

```
public void doSetPassword(String userId, String newPasswd)
{
   try
   {
      SetPasswordResult result = binding.setPassword(userId, newPasswd);
      Console.WriteLine("The password for user ID " + userId + " changed to "
            + newPasswd);
   }
   catch (SoapException e)
   {
      Console.WriteLine("An unexpected error has occurred: " +
                        e.Message + "\n" + e.StackTrace);
   }
}
```

Arguments

Name	Type	Description
`userID`	ID	ID of the User or SelfServiceUser whose password you want to reset. For information on IDs, see ID Field Type.
`password`	string	New password to use for the specified user.

Response

None.

Fault

`InvalidIdFault`

`UnexpectedErrorFault`

See Also:

resetPassword()
Utility Calls
https://wiki.developerforce.com/index.php/Sample_SOAP_Messages

Chapter 12

SOAP Headers

The API provides SOAP headers to client applications. All of these headers are available in both the enterprise and partner WSDL files.

Header	Description
AllOrNoneHeader	Specifies whether a call rolls back all changes unless all records are processed successfully. This header is available in API version 20.0 and later.
AllowFieldTruncationHeader	Specifies the truncation behavior for some field types in API version 15.0 and later.
AssignmentRuleHeader	Specifies the assignment rule to use when creating or updating an Account, Case, or Lead.
CallOptions	Specifies the call options for an API request.
DisableFeedTrackingHeader	Specifies whether the changes made in the current call are tracked in feeds.
EmailHeader	Sends an email notification when a request is processed. Provides equivalent functionality for the Salesforce user interface.
LimitInfoHeader—Pilot	A response header returned from calls to the SOAP API. This header returns limit information for the organization. Use this header to monitor your API limits as you make calls against the organization.
LocaleOptions	Specifies the language of the labels returned. The value must be a valid user locale (language and country), such as `de_DE` or `en_GB`. For more information on locales, see the `Language` field on the CategoryNodeLocalization object.
LoginScopeHeader	Specifies the organization ID so that you can authenticate Self-Service users for your organization using the `login()` call.
MruHeader	Indicates whether to update the list of most recently used items (`true`) or not (`false`).
OwnerChangeOptions	Specifies ownership of attachments and notes.
PackageVersionHeader	Specifies the package version for each installed managed package in API version 16.0 and later.
QueryOptions	Specifies the batch size for query results.
SessionHeader	Specifies the session ID returned from the login server after a successful `login()`.
UserTerritoryDeleteHeader	Specifies a user to whom open opportunities are assigned when the current owner is removed from a territory.

AllOrNoneHeader

The AllOrNoneHeader header allows a call to roll back all changes unless all records are processed successfully. The default behavior is to allow partial success of a call: records without errors are committed, while records with errors are marked as failed in the call results. This header is available in API version 20.0 and later.

Even if the header is enabled, it's still necessary to inspect the `success` field in the call result for each record to identify records with errors. All `success` fields are marked as `true` or `false` indicating whether the call was processed successfully. If there is an error associated with at least one record, the `errors` field in the call result for the record gives more information on the error. If other records in the same call have no errors, their `errors` fields indicate that they were rolled back due to failures in other records in the same call.

API Calls

`create()`, `delete()`, `undelete()`, `update()`, `upsert()`

Fields

Element Name	Type	Description
`allOrNone`	boolean	If `true`, any failed records in a call cause all changes for the call to be rolled back. Record changes aren't committed unless all records are processed successfully. The default is `false`. Some records can be processed successfully while others are marked as failed in the call results.

Sample Code—Java

This sample shows how to use the `AllOrNoneHeader`. It attempts to create two contacts. The second contact doesn't have all required fields set and causes a failure on creation. Next, the sample sets the `AllOrNoneHeader` to `true`, and then makes the call to create the contacts. Creating one of the contacts results in an error, so the entire transaction is rolled back and no contacts are created because the `AllOrNoneHeader` was set to `true`.

```
public void allOrNoneHeaderSample() {
  try {
    // Create the first contact.
    SObject[] sObjects = new SObject[2];
    Contact contact1 = new Contact();
    contact1.setFirstName("Robin");
    contact1.setLastName("Van Persie");

    // Create the second contact. This contact doesn't
    // have a value for the required
    // LastName field so the create will fail.
    Contact contact2 = new Contact();
    contact2.setFirstName("Ashley");
    sObjects[0] = contact1;
    sObjects[1] = contact2;

    // Set the SOAP header to roll back the create unless
    // all contacts are successfully created.
    connection.setAllOrNoneHeader(true);
    // Attempt to create the two contacts.
    SaveResult[] sr = connection.create(sObjects);
    for (int i = 0; i < sr.length; i++) {
      if (sr[i].isSuccess()) {
        System.out.println("Successfully created contact with id: " +
          sr[i].getId() + ".");
```

```
        }
        else {
          // Note the error messages as the operation was rolled back
          // due to the all or none header.
          System.out.println("Error creating contact: " +
            sr[i].getErrors()[0].getMessage());
          System.out.println("Error status code: " +
            sr[i].getErrors()[0].getStatusCode());
        }
      }
    } catch (ConnectionException ce) {
      ce.printStackTrace();
    }
  }
}
```

AllowFieldTruncationHeader

The `AllowFieldTruncationHeader` header specifies the truncation behavior of the following field datatypes that contain strings:

- anyType, if it represents one of the other datatypes in this list
- email
- encryptedstring
- mulitpicklist
- phone
- picklist
- string
- textarea

In API versions previous to 15.0, if you specify a value for a field that is one of the datatypes listed, and that value is too large, the value is truncated. For API version 15.0 and later, if a value is specified that is too large, the operation fails and the fault code `STRING_TOO_LONG` is returned. `AllowFieldTruncationHeader` allows you to specify that the previous behavior, truncation, be used instead of the new behavior in API versions 15.0 and later. This header has no effect in versions 14.0 and earlier.

API Calls

`convertLead()`, `create()`, `merge()`, `process()`, `undelete()`, `update()`, and `upsert()`

Apex: `executeanonymous()`

Fields

Element Name	Type	Description
`allowFieldTruncation`	boolean	If `true`, truncate field values that are too long, which is the behavior in API versions 14.0 and earlier. Default is `false`: no change in behavior. If a `string` or `textarea` value is too large, the operation fails and the fault code `STRING_TOO_LONG` is returned. The following list shows the field types affected by truncation and this header: • anyType, if it represents one of the other datatypes in this list • email • encryptedstring

Element Name	Type	Description
		• mulitpicklist • phone • picklist • string • textarea

Sample Code—Java

This sample shows how to turn field truncation on to allow the creation of an account with a name that is longer than the maximum field length. It creates an account sObject with a name that exceeds the field limit of 255 characters. It then attempts to make the create call, which fails because of the name field length. Next, it sets the `AllowFieldTruncationHeader` to `true` and retries the account creation, which succeeds.

```
public void allowFieldTruncationSample() {
  try {
    Account account = new Account();
    // Construct a string that is 256 characters long.
    // Account.Name's limit is 255 characters.
    String accName = "";
    for (int i = 0; i < 256; i++) {
      accName += "a";
    }
    account.setName(accName);
    // Construct an array of SObjects to hold the accounts.
    SObject[] sObjects = new SObject[1];
    sObjects[0] = account;
    // Attempt to create the account. It will fail in API version 15.0
    // and above because the account name is too long.
    SaveResult[] results = connection.create(sObjects);
    System.out.println("The call failed because: "
       + results[0].getErrors()[0].getMessage());
    // Now set the SOAP header to allow field truncation.
    connection.setAllowFieldTruncationHeader(true);
    // Attempt to create the account now.
    results = connection.create(sObjects);
    System.out.println("The call: " + results[0].isSuccess());
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

AssignmentRuleHeader

The `AssignmentRuleHeader` must be specified in the `create()` or `update()` call of a Case or Lead for the specified assignment rule to be applied, and it must be specified in the `update()` call of an Account for the territory assignment rules to be applied.

API Calls

`create()`, `merge()`, `update()`, `upsert()`

Fields

Element Name	Type	Description
assignmentRuleId	ID	The ID of a specific assignment rule to run for the Case or Lead. The assignment rule can be active or inactive. The ID can be retrieved by querying the AssignmentRule object. If specified, do not specify useDefaultRule. This element is ignored for accounts, because all territory assignment rules are applied. If the value is not in correct ID format (15-character or 18-character Salesforce ID), the call fails and a MALFORMED_ID exception is returned.
useDefaultRule	boolean	If true for a Case or Lead, uses the default (active) assignment rule for a Case or Lead. If specified, do not specify an assignmentRuleId. If true for an Account, all territory assignment rules are applied, and if false, no territory assignment rules are applied.

Sample Code

For a code example, see Lead.

See Also:

AssignmentRule

CallOptions

Specifies the options needed to work with a specific client. This header is only available for use with the Partner WSDL.

API Calls

The defaultNamespace element supports the following calls: create(), merge(), queryAll(), query(), queryMore(), retrieve(), search(), update(), and upsert().

The client element supports all of the above calls, plus the following: convertLead(), login(), delete(), describeGlobal(), describeLayout(), describeTabs(), describeSObject(), describeSObjects(), getDeleted(), getUpdated(), process(), undelete(), getServerTimestamp(), getUserInfo(), setPassword(), and resetPassword().

Fields

Element Name	Type	Description
client	string	A string that identifies a client.
defaultNamespace	string	A string that identifies a developer namespace prefix. Use this field to resolve field names in managed packages without having to fully specify the fieldName everywhere. For example, if the developer namespace prefix is battle, and you have a custom field in your package called botId, you can set this header, and then queries such as the following will succeed: `query("SELECT id, botId__c from Account");`

Element Name	Type	Description
		In this case the actual field queried is the `battle__botId__c` field. Using this field allows you to write client code without having to specify the namespace prefix. Without this field specified, the full name of the field would have to be used for the query to succeed. In the example above, you would have to specify `battle__botId__c`. Note that if this field is set, and the query specifies the namespace as well, the response will not include the prefix. For example, if you set this header to `battle`, and issue a query like `query("SELECT id, battle__botId__c from Account");`, the response would use a `botId__c` element, not a `battle_botId__c` element. Describe calls ignore this header, so there will be no ambiguity between fields with namespace prefixes and customer fields of the same name without the prefix.

Sample Code—C#

This sample shows how to use the `CallOptions` header. It sets a client ID and a developer namespace prefix, which is used to resolve field names in managed packages. Next, the sample logs the specified user in.

```
public void CallOptionsSample()
{
    // Web Reference to the imported Partner WSDL.
    APISamples.partner.SforceService partnerBinding;

    string username = "USERNAME";
    string password = "PASSWORD";

    // The real Client ID will be an API Token provided by salesforce.com
    // to partner applications following a security review.
    // For more details, see the Security Review FAQ in the online help.
    string clientId = "SampleCaseSensitiveToken/100";

    partnerBinding = new SforceService();
    partnerBinding.CallOptionsValue = new CallOptions();
    partnerBinding.CallOptionsValue.client = clientId;

    // Optionally, if a developer namespace prefix has been registered for
    // your Developer Edition organization, it may also be specified.
    string prefix = "battle";
    partnerBinding.CallOptionsValue.defaultNamespace = prefix;

    try
    {
        APISamples.partner.LoginResult lr =
            partnerBinding.login(username, password);
    }
    catch (SoapException e)
    {
        Console.WriteLine(e.Code);
        Console.WriteLine(e.Message);
    }
}
```

DisableFeedTrackingHeader

The `DisableFeedTrackingHeader` header specifies whether the changes made in the current call are tracked in feeds. A user can subscribe to a record or to another user. Changes to the record and updates from the users are displayed in the Chatter feed on the user's home page, which is a useful way to stay up-to-date with other users and with changes made to records in Salesforce.

Use this header if you want to process a large number of records without tracking the changes in various feeds related to the records. This header is available if the Chatter feature is enabled for your organization. For more information on Chatter, see "Chatter Overview" in the Salesforce online help.

API Calls

`convertLead()`, `create()`, `delete()`, `merge()`, `process()`, `undelete()`, `update()`, `upsert()`

Fields

Element Name	Type	Description
`disableFeedTracking`	boolean	If `true`, the changes made in the current call are not tracked in feeds. The default is `false`.

Sample Code—Java

This sample shows how to use the `DisableFeedTrackingHeader`. It sets this header to `true` to disable feed tracking and then creates a large number of accounts in bulk.

```
public void disableFeedTrackingHeaderSample() {
  try {
    // Insert a large number of accounts.
    SObject[] sObjects = new SObject[500];
    for (int i = 0; i < 500; i++)  {
       Account a = new Account();
       a.setName("my-account-" + i);
       sObjects[i] = a;
    }
    // Set the SOAP header to disable feed tracking to avoid generating a
    // large number of feed items because of this bulk operation.
    connection.setDisableFeedTrackingHeader(true);
    // Perform the bulk create. This won't result in 500 feed items, which
    // would otherwise be generated without the DisableFeedTrackingHeader.
    SaveResult[] sr = connection.create(sObjects);
    for (int i = 0; i < sr.length; i++) {
      if (sr[i].isSuccess()) {
        System.out.println("Successfully created account with id: " +
          sr[i].getId() + ".");
      } else {
        System.out.println("Error creating account: " +
          sr[i].getErrors()[0].getMessage());
      }
    }
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

See Also:

NewsFeed
EntitySubscription

EmailHeader

The Salesforce user interface allows you to specify whether or not to send an email when these events occur:

- Create a new Case or Task
- Create a CaseComment
- Convert Case email to a Contact
- New User email notification
- A `resetPassword()` call

In API version 8.0 and later, you can send an API request that sends email as well.

A group event is an Event for which `IsGroupEvent` is true. The EventRelation object tracks the users, leads, or contacts that are invited to a group event. Note the following behaviors for group event email sent through the API:

- Sending a group event invitation to a User respects the `triggerUserEmail` option
- Sending a group event invitation to a Lead or Contact respects the `triggerOtherEmail` option
- Email sent when updating or deleting a group event also respect `triggerUserEmail` and `triggerOtherEmail`, as appropriate

API Calls

`create()`, `delete()`, `resetPassword()`, `update()`, `upsert()`

Fields

Element Name	Type	Description
`triggerAutoResponseEmail`	boolean	Indicates whether to trigger auto-response rules (`true`) or not (`false`), for leads and cases. In the Salesforce user interface, this email can be automatically triggered by a number of events, for example creating a case or resetting a user password. If this value is set to `true`, when a Case is created, if there is an email address for the contact specified in `ContactId`, the email is sent to that address. If not, the email is sent to the address specified in `SuppliedEmail`.
`triggerOtherEmail`	boolean	Indicates whether to trigger email outside the organization (`true`) or not (`false`). In the Salesforce user interface, this email can be automatically triggered by creating, editing, or deleting a contact for a case.
`triggerUserEmail`	boolean	Indicates whether to trigger email that is sent to users in the organization (`true`) or not (`false`). In the Salesforce user interface, this email can be automatically triggered by a number of events; resetting a password, creating a new user, adding comments to a case, or creating or modifying a task.

Sample Code—Java

This sample shows how to use the `EmailHeader`. It sets the `triggerAutoResponseEmail` email header field to `true`, which triggers an email to be sent when a case is created. Next, it creates a case. This sample assumes an auto-response rule has been set for cases, and an email address is specified in the contact referenced by `ContactId`.

```
public void createCaseWithAutoResponse(String contactId) {
  try {
    connection.setEmailHeader(true, false, false);
    Case c = new Case();
    c.setSubject("Sample Subject");
    c.setContactId(contactId);
    SaveResult[] sr = connection.create(new SObject[] { c });
    // Parse sr array to see if case was created successfully.
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

LimitInfoHeader

A response header returned from calls to the SOAP API. This header returns limit information for the organization. Use this header to monitor your API limits as you make calls against the organization.

API Calls

All calls, except for `login()`.

Fields

<table>
<tr><th>Element Name</th><th>Type</th><th>Description</th></tr>
<tr><td>current</td><td>string</td><td>The number of calls for the specified limit type that have already been used in the organization.</td></tr>
<tr><td>limit</td><td>string</td><td>The organization’s limit for the specified limit type.</td></tr>
<tr><td>type</td><td>string</td><td>

The type of limit information specified in the header.

- `API REQUESTS`—contains limit information about API calls for the organization.
- `API REQUESTS PER APP`—contains limit quota information, if there is any, for the currently connected app. API limit app quotas are currently available through a pilot program. For information on enabling this feature for your organization, contact salesforce.com. This example includes the limit quota for a `sample-connected-app` connected app.

```
<limitInfo>
    <appName>sample-connected-app</appName>
    <current>20</current>
    <limit>250</limit>
    <type>API REQUESTS PER APP</type>
</limitInfo>
```

</td></tr>
</table>

Sample Code

This is an example of a response to a SOAP request for a Merchandise record. The `LimitInfoHeader` contains the API usage information for the organization.

```
<?xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns="urn:partner.soap.sforce.com" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

    xmlns:sf="urn:sobject.partner.soap.sforce.com">
    <soapenv:Header>
        <LimitInfoHeader>
            <limitInfo>
                <current>5</current>
                <limit>5000</limit>
                <type>API REQUESTS</type>
            </limitInfo>
        </LimitInfoHeader>
    </soapenv:Header>
    <soapenv:Body>
        <queryResponse>
            <result xsi:type="QueryResult">
                <done>true</done>
                <queryLocator xsi:nil="true"/>
                <records xsi:type="sf:sObject">
                    <sf:type>dev_ns__Merchandise__c</sf:type>
                    <sf:Id>a00D0000008pQSNIA2</sf:Id>
                    <sf:dev_ns__Description__c>Phone Case for iPhone
                        4/4S</sf:dev_ns__Description__c>
                    <sf:dev_ns__Price__c>16.99</sf:dev_ns__Price__c>
                    <sf:dev_ns__Stock_Price__c>12.99</sf:dev_ns__Stock_Price__c>
                    <sf:dev_ns__Total_Inventory__c>108.0</sf:dev_ns__Total_Inventory__c>
                    <sf:Id>a00D0000008pQSNIA2</sf:Id>
                </records>
                <size>1</size>
            </result>
        </queryResponse>
    </soapenv:Body>
</soapenv:Envelope>
```

LocaleOptions

Specifies the language of the labels returned.

API Calls

`describeSObject()`, `describeSObjects()`, `describeDataCategoryGroups()`, `describeDataCategoryGroupStructures()`

Fields

Element Name	Type	Description
`language`	string	Specifies the language of the labels returned. The value must be a valid user locale (language and country), such as `de_DE` or `en_GB`. For more information on locales, see the `Language` field on the CategoryNodeLocalization object.

Sample Code—Java

This sample sets the `LocaleOptions` header to the locale of the logged-in user, and then performs a describe on Account.

```
public void localeOptionsExample() {
  try {
    connection.setLocaleOptions("en_US");
    connection.describeSObject("Account");
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

LoginScopeHeader

Specifies your organization ID so that you can authenticate Self-Service users for your organization using the existing `login()`.

API Calls

`login()`

Fields

Element Name	Type	Description
`organizationId`	ID	The ID of the organization against which you will authenticate Self-Service users.
`portalId`	ID	Specify only if user is a Customer Portal user. The ID of the portal for this organization. The ID is available in the Salesforce user interface: • From Setup, click **Customize** > **Customer Portal** > **Settings** • Select a Customer Portal name, and on the Customer Portal detail page, the URL of the Customer Portal displays. The Portal ID is in the URL.

Sample Code—C#

This sample shows how to use the `LoginScopeHeader`. It sets the organization ID and the portal ID for a Customer Portal user. It also sets the `CallOptions` header. It then logs the specified user in.

```
/// Demonstrates how to set the LoginScopeHeader values.
public void LoginScopeHeaderSample()
{
    // Web Reference to the imported Partner WSDL.
    APISamples.partner.SforceService partnerBinding;

    string username = "USERNAME";
    string password = "PASSWORD";

    // The real Client ID will be an API Token provided by salesforce.com
    // to partner applications following a security review. For more details,
    // see the Security Review FAQ in the online help.
    string clientId = "SampleCaseSensitiveToken/100";

    partnerBinding = new SforceService();
    partnerBinding.CallOptionsValue = new CallOptions();
    partnerBinding.CallOptionsValue.client = clientId;

    // To authenticate Self-Service users, we need to set the OrganizationId
    // in the LoginScopeHeader.
```

```
    string orgId = "00ID0000OrgFoo";
    partnerBinding.LoginScopeHeaderValue = new LoginScopeHeader();
    partnerBinding.LoginScopeHeaderValue.organizationId = orgId;
    // Specify the Portal ID if the user is a Customer Portal user.
    string portalId = "00ID0000FooPtl";
    partnerBinding.LoginScopeHeaderValue.portalId = portalId;

    try
    {
        APISamples.partner.LoginResult lr =
            partnerBinding.login(username, password);
    }
    catch (SoapException e)
    {
        Console.WriteLine(e.Code);
        Console.WriteLine(e.Message);
    }
}
```

MruHeader

In API version 7.0 and later, the `create()`, `update()`, and `upsert()` calls do not update the list of most recently used (MRU) items in the Recent Items section of the sidebar in the Salesforce user interface unless this header is used. Be advised that using this header to update the Recent Items list may negatively impact performance.

API Calls

`create()`, `merge()`, `query()`, `retrieve()`, `update()`, `upsert()`

Fields

Element Name	Type	Description
`updateMru`	boolean	Indicates whether to update the list of most recently used items (`true`) or not (`false`). For `retrieve()`, if the result has only one row, the MRU is updated to the ID of the retrieve result. For `query()`, if the result has only one row and the ID field is selected, the MRU is updated to the ID of the query result.

Sample Code—Java

This sample turns on the MRU list update option by setting the `MruHeader` to `true`. Next, it creates an account.

```
public void mruHeaderSample() {
  connection.setMruHeader(true);
  Account account = new Account();
  account.setName("This will be in the MRU");
  try {
    SaveResult[] sr = connection.create(new SObject[]{account});
    System.out.println("ID of account added to MRU: " +
      sr[0].getId());
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

OwnerChangeOptions

Specifies ownership of attachments, notes, and open activities when transfers occur.

API Calls

merge(), update(), upsert()

Fields

Element Name	Type	Description
transferAttachments	boolean	If true, the record's notes, attachments, and Google Docs are transferred to the new record owner. If false, the original record owner retains ownership. The default is true.
transferOpenActivites	boolean	If true, the record's open activities are transferred to the new record owner. If false, the original record owner retains ownership. The default is true.

Sample Code—Java

This sample creates an account, a note and task for the account, and sets the owner change options so that the note and task are transferred to the new owner.

```
public void ownerChangeOptionsHeaderSample() {

    // Create account. Accounts don't transfer activities, notes, or attachments by default


    Account account = new Account();
    account.setName("Account");
    com.sforce.soap.enterprise.SaveResult[] sr = connection.create(new
com.sforce.soap.enterprise.sobject.SObject[] { account } );
    String accountId = null;

    if(sr[0].isSuccess()) {
        System.out.println("Successfully saved the account");
        accountId = sr[0].getId();

        // Create a note and a task for the account

        Note note = new Note();
        note.setTitle("Note Title");
        note.setBody("Note Body");
        note.setParentId(accountId);

        Task task = new Task();
        task.setWhatId(accountId);

        sr = connection.create(new com.sforce.soap.enterprise.sobject.SObject[] { note,
task } );

        if(sr[0].isSuccess()) {
            System.out.println("Successfully saved the note and task");

           com.sforce.soap.enterprise.QueryResult qr = connection.query("SELECT Id FROM
User WHERE FirstName = 'Jane' AND LastName = 'Doe'");
           String newOwnerId = qr.getRecords()[0].getId();
           account.setId(accountId);
           account.setOwnerId(newOwnerId);
```

```
          // Set owner change options so account's child note & task transfer to new owner


          connection.setOwnerChangeOptions(true, true);
          connection.update(new com.sforce.soap.enterprise.sobject.SObject[] { account }
 );

          // The account, account's note, & task should be transferred to the new owner.
       }

   } else {
      System.out.println("Account save failed: " + sr[0].getErrors().toString());
    }
}
```

PackageVersionHeader

Specifies the package version for each installed managed package. A package version is a number that identifies the set of components uploaded in a package. The version number has the format *majorNumber.minorNumber.patchNumber* (for example, 2.1.3). The major and minor numbers increase to a chosen value during every major release. The *patchNumber* is generated and updated only for a patch release. As well as a set of components, a package version encompasses specific behavior. Publishers can use package versions to evolve the components in their managed packages gracefully by releasing subsequent package versions without breaking existing customer integrations using the package.

A managed package can have several versions with different content and behavior. This header allows you to specify the version used for each package referenced by your API client. If a package version is not specified for a package, the API client uses the version of the package that is selected in the Version Settings section under Setup, in **Develop** > **API**. This header is available in API version 16.0 and later.

Associated API Calls

`convertLead()`, `create()`, `delete()`, `describeGlobal()`, `describeLayout()`, `describeSObject()`, `describeSObjects()`, `describeSoftphoneLayout()`, `describeTabs()`, `merge()`, `process()`, `query()`, `retrieve()`, `search()`, `undelete()`, `update()`, `upsert()`

Fields

Element Name	Type	Description
`packageVersions`	PackageVersion[]	A list of package versions for installed managed packages referenced by your API client.

PackageVersion

Specifies a version of an installed managed package. It includes the following fields:

Field	Type	Description
`majorNumber`	int	The major version number of a package version. A package version is denoted by *majorNumber.minorNumber*, for example `2.1`.
`minorNumber`	int	The minor version number of a package version. A package version is denoted by *majorNumber.minorNumber*, for example `2.1`.
`namespace`	string	The unique namespace of the managed package.

Sample Code—Java

This sample sets the package version for one installed package in the `PackageVersionHeader`. Next, it executes the code passed into this method via the `executeAnonymous` Apex method.

```
public void PackageVersionHeaderSample(String code) throws Exception
{
    _PackageVersionHeader pvh = new _PackageVersionHeader();
    PackageVersion pv = new PackageVersion();
    pv.setNamespace("installedPackageNamespaceHere");
    pv.setMajorNumber(1);
    pv.setMinorNumber(0);
    // In this case, we are only referencing one installed package.
    PackageVersion[] pvs = new PackageVersion[]{pv};
    pvh.setPackageVersions(pvs);

    apexBinding.setHeader(new SforceServiceLocator().getServiceName().getNamespaceURI(),
        "PackageVersionHeader", pvh);
    // Execute the code passed into the method.
    ExecuteAnonymousResult r = apexBinding.executeAnonymous(code);
    if (r.isSuccess()) {
        System.out.println("Code executed successfully");
    }
    else {
        System.out.println("Exception message: " + r.getExceptionMessage());
        System.out.println("Exception stack trace: " + r.getExceptionStackTrace());
    }
}
```

QueryOptions

Specifies the batch size for queries. Batches that are larger or smaller than the specified size may be used, in order to maximize performance.

Associated API Calls

`query()`, `queryMore()`, `retrieve()`

Fields

Element Name	Type	Description
batchSize	int	The batch size for the number of records returned in a `query()` or `queryMore()` call. Child objects count toward the number of records for the batch size. For example, in relationship queries, multiple child objects may be returned per parent row returned. The default is 500; the minimum is 200, and the maximum is 2,000. There is no guarantee that the requested batch size will be the actual batch size. This is done to maximize performance.

Sample Code

For code examples, see Changing the Batch Size in Queries in the *Salesforce SOQL and SOSL Reference Guide*.

SessionHeader

Specifies the session ID returned from the login server after a successful `login()`. This session ID is used in all subsequent calls.

In version 12.0 and later, you must include the API namespace in the SOAP message associated with this header. The namespace is defined in the enterprise or partner WSDL.

API Calls

All calls, including utility calls.

Fields

Element Name	Type	Description
`sessionId`	string	Session ID returned by the `login()` call to be used for subsequent call authentication.

Sample Code

See the examples provided for `login()`.

UserTerritoryDeleteHeader

Specify a user to whom open opportunities are assigned when the current owner is removed from a territory. If this header is not used or the value of its element is null, the opportunities are transferred to the forecast manager in the territory above, if one exists. If one does not exist, the user being removed from the territory keeps the opportunities.

API Calls

`delete()`

Fields

Element Name	Type	Description
`transferToUserId`	ID	The ID of the user to whom open opportunities in that user's territory will be assigned when an opportunity's owner (user) is removed from a territory.

See Also:

Opportunity Forecast Override Business Rules

Chapter 13

Implementation Considerations

In this chapter ...

- Choosing a User for an Integration
- Login Server URL
- Log in to the Login Server
- Typical API Call Sequence
- Salesforce Sandbox
- Multiple Instances of Salesforce.com Database Servers
- Content Type Requirement
- Monitoring API Traffic
- API Usage Metering
- Compression
- HTTP Persistent Connections
- HTTP Chunking
- Internationalization and Character Sets
- XML Compliance
- .NET, Non-String Fields, and the Enterprise WSDL

Before you build an integration application or other client application, consider the data management, use limits, and communication issues explained in this section.

Choosing a User for an Integration

When your client application connects to the API, it must first log in. You must specify a user to log in to Salesforce when calling `login()`. Client applications run with the permissions and sharing of the logged-in user. The following sections include information that help you to decide how to configure a user for your client application.

Permissions

An organization's Salesforce administrator controls the availability of various features and views by configuring profiles and permission sets, and assigning users to them. To access the API (to issue calls and receive the call results), a user must be granted the "API Enabled" permission. Client applications can query or update only those objects and fields to which they have access via the permissions of the logged-in user.

If the client application logs in as a user who has access to data via a sharing rule, then the API must issue an extra query to check access. To avoid this, log in as a user with the "Modify All Data" permission. This can speed up the call response time. If providing the "Modify All Data" permission is too permissive for a particular user, consider using the "Modify All" object-level permission to restrict data access on an object basis. For more information on permissions, see Factors that Affect Data Access.

Limits

There is a limit on the number of queries that a user can execute concurrently. A user can have up to 10 query cursors open at a time. If 10 `QueryLocator` cursors are open when a client application, logged in as the same user, attempts to open a new one, then the oldest of the 10 cursors is released. If the client application attempts to open the released query cursor, an error results.

Multiple client applications can log in using the same `username` argument. However, this increases your risk of getting errors due to query limits.

If multiple client applications are logged in using the same user, they all share the same session. If one of the client applications calls `logout()`, it invalidates the session for all the client applications. Using a different user for each client application makes it easier to avoid these limits.

Note: In addition to user limits, there are limits for API requests for each organization. For more information, see API Usage Metering.

Login Server URL

The SOAP implementation of the API also provides a single login server. You can log in to any organization via a single entry point, without having to hard-code the instance for your organization. To access an organization via the API, you must first authenticate the session by sending a `login()` request to the login server at one of the following URLs, depending on your choice of WSDL.

- `https://login.salesforce.com/services/Soap/c/31.0` (enterprise WSDL)
- `https://login.salesforce.com/services/Soap/u/31.0` (partner WSDL)

The less secure version of each URL is also supported.

```
http://login.salesforce.com/services/Soap/c/31.0
```

- `http://login.salesforce.com/services/Soap/c/31.0` (enterprise WSDL)
- `http://login.salesforce.com/services/Soap/u/31.0` (partner WSDL)

The less secure version of the URL is supported, but not recommended. It is helpful for debugging through proxy servers.

All subsequent calls to the server during the session should be made to the URL returned in the `login()` response, which points to the server instance for your organization.

Log in to the Login Server

Before invoking any other calls, a client application must first invoke the `login()` call to establish a session with the login server, set the returned server URL as the target server for subsequent API requests, and set the returned session ID in the SOAP header to provide server authorization for subsequent API requests. For more information, see `login()` and Step 4: Walk Through the Sample Code.

Salesforce checks the IP address from which the client application is logging in, and blocks logins from unknown IP addresses. For a blocked login via the API, Salesforce returns a login fault. Then, the user must add their security token to the end of their password in order to log in. A security token is an automatically-generated key from Salesforce. For example, if a user's password is `mypassword`, and their security token is `XXXXXXXXXX`, then the user must enter `mypasswordXXXXXXXXXX` to log in. Users can obtain their security token by changing their password or resetting their security token via the Salesforce user interface. When a user changes their password or resets their security token, Salesforce sends a new security token to the email address on the user's Salesforce record. The security token is valid until a user resets their security token, changes their password, or has their password reset. When the security token is invalid, the user must repeat the login process to log in. To avoid this, the administrator can make sure the client's IP address is added to the organization's list of trusted IP addresses. For more information, see Security Token.

Once you are logged in, you can issue API calls. For each operation, client applications submit a synchronous request to the API, await the response, and process the results. The API commits any changed data automatically.

API calls:

- Core Calls
- Describe Calls
- Utility Calls

Typical API Call Sequence

For each call, your client application typically:

1. Prepares the request by defining request parameters, if applicable.
2. Invokes the call, which passes the request with its parameters to the Force.com Web Service for processing.
3. Receives the response from the API.
4. Handles the response, either by processing the returned data (for a successful invocation) or by handling the error (for a failed invocation).

Salesforce Sandbox

Enterprise, Unlimited, and Performance Edition customers have access to the Salesforce Sandbox, which is a testing environment that offers a full or partial copy of your organization's live production data. For more information about Salesforce Sandbox, please visit the Salesforce.com Community website at www.salesforce.com/community or see “Sandbox Overview” in the Salesforce online help.

To access your organization's Salesforce Sandbox via the API, use the following URLs to make login requests.

- `https://test.salesforce.com/services/Soap/c/31.0` (enterprise WSDL)
- `https://test.salesforce.com/services/Soap/u/31.0` (partner WSDL)

Multiple Instances of Salesforce.com Database Servers

Salesforce.com provides many database server instances. Although organizations are generally allocated by geographic regions, an organization may be on any instance.

Content Type Requirement

In the API version 7.0 and later, all requests must contain a correct content type HTTP header, for example: `Content-Type: text/xml; charset=utf-8`. Earlier versions of the API do not enforce this requirement.

Monitoring API Traffic

You can monitor the number of API requests generated by your organization in two ways.

- Any user can see the number of API requests sent in the last 24 hours. To view the information, from Setup, click **Company Profile** > **Company Information**. Look for the “API Requests, Last 24 Hours” field in the right column.
- If a user has the “Modify All Data” permission, the user can view a report of the API requests sent for the last seven days. To view the information, click the Reports tab, scroll to the **Administrative Reports** section and click the **API Usage Last 7 Days** link. Users can sort the report by any of the fields listed in the **Summarize Information by:** drop-down list. For more information about sorting, filtering, or customizing reports, see the Salesforce online help for reports.

API Usage Metering

To maintain optimum performance and ensure that the Force.com API is available to all of our customers, salesforce.com balances transaction loads by imposing two types of limits:

- Concurrent API Request Limits
- Total API Request Limits

When a call exceeds a request limit, an error is returned.

Concurrent API Request Limits

The following table lists the limits for various types of organizations for concurrent requests (calls) with a duration of 20 seconds or longer:

Organization Type	Limit
Developer Edition	5
Trial organizations	5
Production organizations	25
Sandbox	25

Total API Request Limits

The following table lists the limits for the total API requests (calls) per 24-hour period for an organization.

Salesforce Edition	API Calls Per License Type	Minimum	Maximum
All Editions: DebuggingHeader on API testing calls for Apex specified. Valid in API version 20 and later.	N/A	1,000	1,000
Developer Edition	N/A	15,000	15,000
• Enterprise Edition • Professional Edition with API access enabled	• Salesforce: 1,000 • Force.com App Subscription: 200 • Salesforce Platform: 1,000 **Note:** This license is not available to new customers. • Force.com - One App: 200 **Note:** This license is not available to new customers. • Partner Community: 200 • Gold Partner: 200 **Note:** This license is not available to new customers.	5,000	1,000,000
• Unlimited Edition • Performance Edition	• Salesforce: 5,000 • Force.com App Subscription: 200 • Salesforce Platform: 5,000 **Note:** This license is not available to new customers. • Force.com - One App: 200 **Note:** This license is not available to new customers. • Partner Community: 200 • Gold Partner: 200 **Note:** This license is not available to new customers.	5,000	Unlimited. However, at any high limit, it is likely that other limiting factors such as system load may prevent you from using your entire allocation of calls in a 24–hour period.
Sandbox	N/A	N/A	5,000,000

Limits are enforced against the aggregate of all API calls made by the organization in a 24 hour period; limits are not on a per-user basis. When an organization exceeds a limit, all users in the organization may be temporarily blocked from making additional calls. Calls will be blocked until usage for the preceding 24 hours drops below the limit.

Note: Limits are automatically enforced for all editions.

Any action that sends a call to the API counts toward usage limits, except the following:

- Outbound messages
- Apex callouts

You can create an API usage metering notification, so that Salesforce will send an email to a designated user when API limits have exceeded a specified limit in a specified time period. For more information, see "About API Usage Notifications" in the Salesforce online help.

There are also limits on the number of requests allowed per organization from the Salesforce user interface. For details, see "Concurrent Usage Limits" in the Salesforce online help.

Example API Usage Metering Calculations

The following examples illustrate API usage metering calculations for several scenarios:

- For an Enterprise Edition organization with five Salesforce licenses, the request limit is 5,000 requests (5 licenses X 1,000 calls).
- For an Enterprise Edition organization with 15,000 Salesforce licenses, the request limit is 1,000,000 (the number of licenses X 1,000 calls is greater than the maximum value, so the lower limit of 1,000,000 is used).
- For a Developer Edition organization that made 4,500 calls at 5:00 AM Wednesday, 499 calls at 11:00 PM Wednesday, only one more call could successfully be made until 5:00 AM Thursday.
- For an Enterprise Edition organization with 20 Gold Partner licenses, the request limit is 5,000 (the number of licenses X 200 calls is less than the minimum value of 5,000).

Increasing Total API Request Limit

The calculation of the API request limit based on user licenses is designed to allow sufficient capacity for your organization based on your number of users. If you need a higher limit and you don't want to purchase additional user licenses or upgrade to Performance Edition, you can purchase additional API calls. For more information, contact your account representative.

Before you purchase additional API calls, you should perform a due diligence of your current API usage. Client applications, whether it is your own enterprise applications or partner applications, that make calls to the API can often be optimized to use fewer API calls to do the same work. If you are using a partner product, you should consult with the vendor to verify that the product is making optimal use of the API. A product that makes inefficient use of the API will incur unnecessary cost for your organization.

Compression

The API allows the use of compression on the request and the response, using the standards defined by the HTTP 1.1 specification. This is automatically supported by some SOAP/WSDL clients, and can be manually added to others. Visit `https://developer.salesforce.com/page/Tools` for more information on particular clients.

Compression is not used unless the client specifically indicates that it supports compression. For better performance, we suggest that clients accept and support compression as defined by the HTTP 1.1 specification.

To indicate that the client supports compression, you should include the HTTP header "Accept-Encoding: gzip, deflate" or a similar heading. The API compresses the response if the client properly specifies this header. The response includes the header "Content-Encoding: deflate" or "Content-Encoding: gzip," as appropriate. You can also compress any request by including a "Content-Encoding: deflate" or "gzip" header.

Most clients are partially constrained by their network connection, even on a corporate LAN. The API allows the use of compression to improve performance. Almost all clients can benefit from response compression, and many clients may benefit from compression of requests as well. The API supports deflate and gzip compression according the HTTP 1.1 specification.

Response Compression

The API can optionally compress responses. Responses are compressed only if the client sends an Accept-Encoding header with either gzip or deflate compression specified. The API is not required to compress the response even if you have specified Accept-Encoding, but it normally does. If the API compresses the response, it also specifies a Content-Encoding header with the name of the compression algorithm used, either gzip or deflate.

Request Compression

Clients can also compress requests. The API decompresses any requests before processing. The client must send up a Content-Encoding HTTP header with the name of the appropriate compression algorithm. For more information, see:

- Content-Encoding at: `www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.11`
- Accept-Encoding at: `www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14.3`
- Content Codings at: `www.w3.org/Protocols/rfc2616/rfc2616-sec3.html#sec3.5`

Note: To implement request SOAP compression in a Java client with WSC (Web Service Connector), call `setCompression()` on the `Config` you use to instantiate a `Connection` object with. For an example, see `login()` sample on page 1481 code.

HTTP Persistent Connections

Most clients achieve better performance if they use HTTP 1.1 persistent connection to reuse the socket connection for multiple requests. Persistent connections are normally handled by your SOAP/WSDL client automatically. For more details, see the HTTP 1.1 specification at:

`http://www.w3.org/Protocols/rfc2616/rfc2616-sec8.html#sec8.1`

HTTP Chunking

Clients that use HTTP 1.1 may receive chunked responses. Chunking is normally handled by your SOAP/WSDL client automatically.

Internationalization and Character Sets

The API supports either full Unicode characters or ISO-8859-1 characters. The character set for your organization depends on the Salesforce instance your organization uses. If your organization logs into `ssl.salesforce.com`, then your encoding is `ISO-8859-1`. All other instances use `UTF-8`. You can determine the character set for your organization by calling `describeGlobal()` and inspecting the `encoding` value returned in the DescribeGlobalResult.

If your organization uses ISO-8859-1 encoding, then all data sent to the API must be encoded in ISO-8859-1. Characters outside the valid ISO-8859-1 range might be truncated or cause an error.

Note: The API response is encoded in the character set used by your organization (UTF-8 or ISO-8859-1). Either way, the encoded data is usually handled for you by the SOAP client.

XML Compliance

The API is based on XML, which requires all documents to be well formed. Part of that requirement is that certain Unicode characters are not allowed in an XML document, even in an escaped form, and that others must be encoded according to their location. Normally this is handled for you by any standard SOAP or XML client. Clients must be able to parse any normal XML escape sequence, and must not pass up invalid XML characters.

Some characters, as mentioned, are illegal even if they are escaped. The illegal characters include the Unicode surrogate blocks and a few other Unicode characters. All are seldom-used control characters that are usually not important in any data, and tend to cause problems with many programs. Although they are not allowed in XML documents, they are allowed in HTML documents and may be present in Salesforce data. The illegal characters will be stripped from any API response.

Illegal characters:

- `0xFFFE`
- `0xFFFF`
- Control characters `0x0` - `0x19`, except the following characters, which are legal: `0x9`, `0xA`, `0xD`, tab, newline, and carriage return)
- `0xD800` - `0xDFFF`

For UTF-8 encoding, Salesforce supports only the basic UCS-2 encoding (two byte, Basic Multilingual Plane), and does not support any of the extended UCS-4 characters. UCS-4 support is extremely rare in any system. UCS-2 is the encoding that Java and Windows support. For more information about XML characters and character sets, see: http://www.w3.org/TR/REC-xml#charsets.

.NET, Non-String Fields, and the Enterprise WSDL

If you use .NET with the enterprise WSDL, .NET will generate an extra Boolean field for each non-string field. For example, if you had a date value in `MyField__c`, .NET generates a Boolean field for each. In this example, the generated fields would be `MyField__cSpecified` and `LastModifiedSpecified`. These field values are `false` by default. If a Specified field value is `false`, then the values in the corresponding original field will not be included in the SOAP message.

For example, before the values in the currency field `annualRevenue` can be included in a SOAP message generated by your client application, the value of `annualRevenueSpecified` must be set to `true`.

```
account.annualRevenue = 10000;
account.annualRevenueSpecified = true;
```

Chapter 14

Outbound Messaging

In this chapter ...

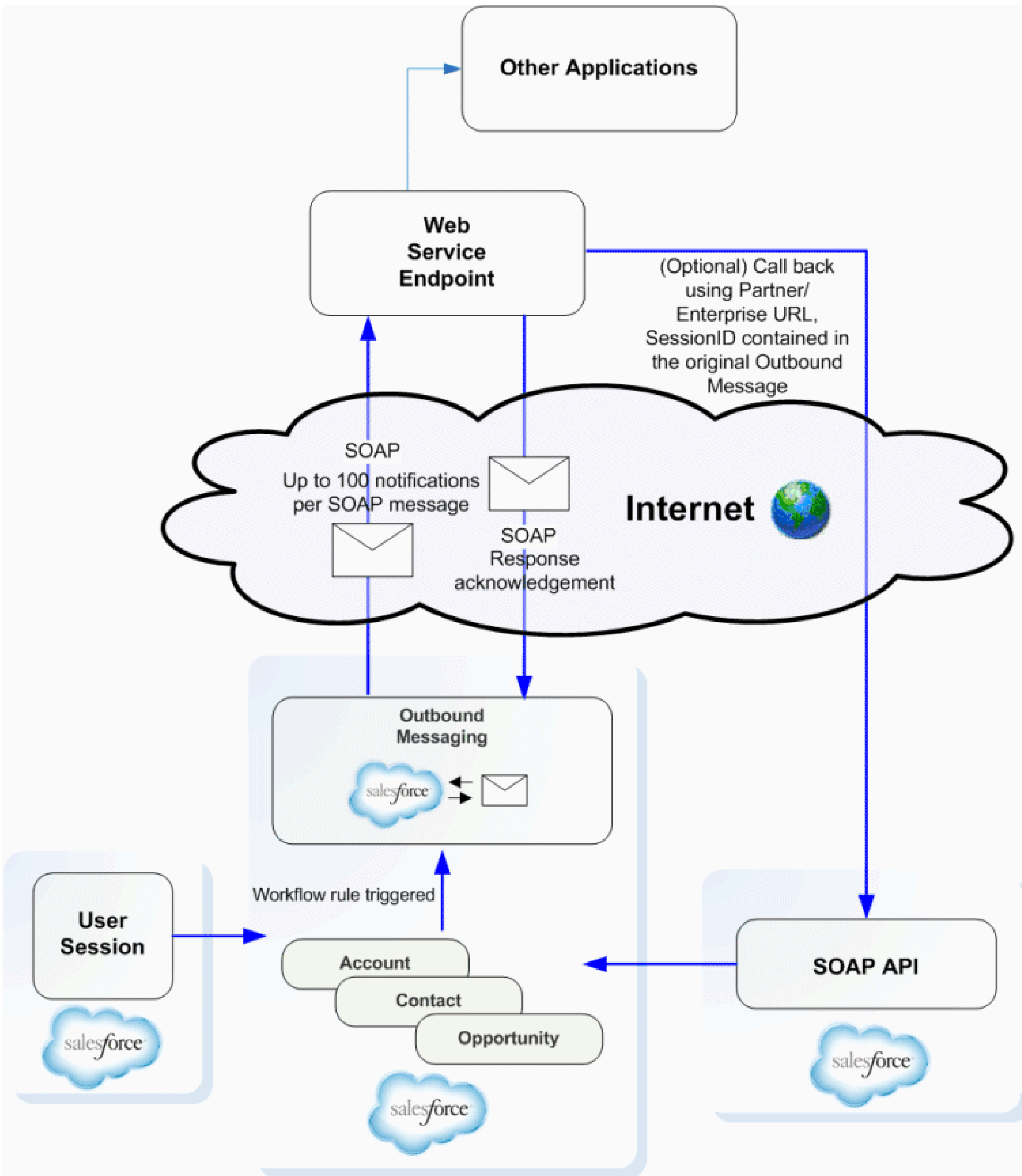
- Understanding Outbound Messaging
- Understanding Notifications
- Setting Up Outbound Messaging
- Important Security Considerations
- Understanding the Outbound Messaging WSDL
- Building a Listener

Outbound messaging allows you to specify that changes to fields within Salesforce can cause messages with field values to be sent to designated external servers.

Outbound messaging is part of the workflow rule functionality in Salesforce. Workflow rules watch for specific kinds of field changes and trigger automatic Salesforce actions, such as sending email alerts, creating task records, or sending an outbound message.

Understanding Outbound Messaging

Outbound messaging uses the `notifications()` call to send SOAP messages over HTTP(S) to a designated endpoint when triggered by a workflow rule.



After you set up outbound messaging, when a triggering event occurs, a message is sent to the specified endpoint URL. The message contains the fields specified when you created the outbound message. Once the endpoint URL receives the message, it can take the information from the message and process it. To do that, you need to examine the outbound messaging WSDL.

Understanding Notifications

A single SOAP message can include up to 100 notifications. Each notification contains the object ID and a reference to the associated sObject data. Note that if the information in the object changes after the notification is queued but before it is sent, only the updated information will be delivered.

If you issue multiple discrete calls, the calls may be batched together into one or more SOAP messages.

Messages will be queued locally. A separate background process performs the actual sending, to preserve message reliability:

- If the endpoint is unavailable, messages will stay in the queue until sent successfully, or until they are 24 hours old. After 24 hours, messages are dropped from the queue.
- If a message cannot be delivered, the interval between retries increases exponentially, up to a maximum of two hours between retries.
- Messages are retried independent of their order in the queue. This may result in messages being delivered out of order.
- You cannot build an audit trail using outbound messaging. While each message should be delivered at least once, it may be delivered more than once. Also, it may not be delivered at all if delivery cannot be done within 24 hours. Finally, as noted above, the source object may change after a notification is queued but before it is sent, so the endpoint will only receive the latest data, not any intermediate changes.
- Because a message may be delivered more than once, your listener client should check the notification IDs delivered in the notification before processing.

Note: Instead of polling, which was required in previous releases, you can now use outbound messaging to trigger execution logic when Salesforce raises an event. In previous versions of the API, client applications had to poll Salesforce to find out if relevant changes had occurred. Because most changes eventually trigger workflow if a rule exists for it, you can use this to trigger actions based on Salesforce events.

The metadata needed for outbound messaging, including the definition of the `notifications()` call, which sends the outbound SOAP message to an external service, is in a separate WSDL. The WSDL is created and available from the Salesforce user interface once a workflow rule has been associated with an outbound message. The WSDL is bound to the outbound message and contains the instructions about how to reach the endpoint service and what data is sent to it. For more information about setting up outbound messaging, see Defining Outbound Messaging.

Setting Up Outbound Messaging

Before you can use outbound messaging, you must set it up via the Salesforce user interface:

- Setting Up User Profiles
- Defining Outbound Messaging
- Downloading the Salesforce Client Certificate
- Viewing Outbound Messages
- Tracking Outbound Message Status

Setting Up User Profiles

It is possible to create circular changes with outbound messaging. For example, if a user is performing integrations that trigger workflow, and the workflow actions trigger account updates, those account updates trigger new workflow, and so on. In order to prevent these circular changes, you can disable a user's ability to send outbound messages.

The following is another example of a circular change scenario:

1. You configure an outbound message to include a `sessionId` and specify a user in the **User to send as** field. The user does not have outbound messaging disabled.
2. A change in a contact record triggers an outbound message from the specified user, with the `sessionId` to your outbound message listener.
3. Your outbound message listener calls the Force.com API and updates the same contact record which triggered the outbound message.
4. The update triggers an outbound message.
5. Your outbound message listener updates the record.
6. The update triggers an outbound message.
7. Your outbound message listener updates the record.

To disable outbound message notifications for a user, deselect “Send Outbound Messages” in the user's Profile. We recommend specifying a single user to respond to outbound messages, and disabling this user's ability to send outbound messages.

Defining Outbound Messaging

To define outbound messages, use this procedure in the Salesforce user interface:

1. From Setup, click **Create** > **Workflow & Approvals** > **Outbound Messages**.
2. Click **New Outbound Message**.
3. Choose the object that has the information you want included in the outbound message, and click **Next**.
4. Configure the outbound message.
   a. Enter a name and description for this outbound message.
   b. Enter an endpoint URL for the recipient of the message. Salesforce sends a SOAP message to this endpoint.

      For security reasons, Salesforce restricts the outbound ports you may specify to one of the following:

      - 80: This port only accepts HTTP connections.
      - 443: This port only accepts HTTPS connections.
      - 1024–66535 (inclusive): These ports accept HTTP or HTTPS connections.

   c. Select the Salesforce user to use when sending the message by specifying a username in the **User to send as** field. The chosen user controls data visibility for the message that is sent to the endpoint.
   d. Select **Include Session ID** if you want a `sessionId` to be included in the outbound message. Include the `sessionId` in your message if you intend to make API calls back to Salesforce from your listener. The `sessionId` represents the user defined in the previous step and not the user who triggered the workflow.
   e. Select the fields you want included in the outbound message and click **Add**.
5. Click **Save**, and review the outbound message detail page:
   - The `API Version` field is automatically generated and set to the current API version when the outbound message was created. This API version is used in API calls back to Salesforce using the enterprise or partner WSDLs. The `API Version` can only be modified by using the Metadata API.

- Click the **Click for WSDL** link to view the WSDL associated with this message.

  The WSDL is bound to the outbound message and contains the instructions about how to reach the endpoint service and what data is sent to it.

Note: If you do not have these options, your organization does not have outbound messaging enabled. Contact salesforce.com to enable outbound messaging for your organization.

Downloading the Salesforce Client Certificate

Your application (endpoint) server's SSL/TLS may be configured to require client certificates (two-way SSL/TLS), in order to validate the identity of the Salesforce server when it takes the role of client to your server. If this is the case, you can download the Salesforce client certificate from the Salesforce application user interface. This is the client certificate that Salesforce sends with each outbound message for authentication.

To download the certificate, use this procedure:

- From Setup, click **Develop** > **API** to display the **WSDL Download** page.
- In the **WSDL Download** page, right-click **Download Client Certificate** and save it to an appropriate location on your local drive.
- Import the downloaded certificate into your application server, and configure your application server to request the client certificate. The application server then checks that the certificate used in the SSL/TLS handshake matches the one you downloaded.

Note: Your application (endpoint) server must send any intermediate certificates in the certificate chain, and the certificate chain must be in the correct order. The correct order is:

1. Server certificate.
2. Intermediate certificate that signed the server certificate if the server certificate was not signed directly by a root certificate.
3. Intermediate certificate that signed the certificate in step 2.
4. Any remaining intermediate certificates. Do not include the root certificate authority certificate. The root certificate is not sent by your server. Salesforce already has its own list of trusted certificates on file, and a certificate in the chain must be signed by one of those root certificate authority certificates.

Viewing Outbound Messages

To view existing outbound messages, from Setup, click **Create** > **Workflow & Approvals** > **Outbound Messages** in the Salesforce user interface.

- Click **New Outbound Message** to define a new outbound message.
- Click **View Message Delivery Status** to track the status of an outbound message.
- Select an existing outbound message to view details about it or view workflow rules and approval processes that use it.
- Click **Edit** to make changes to an existing outbound message.
- Click **Del** to delete an outbound message.

Tracking Outbound Message Status

To track the status of an outbound message, from Setup, click **Monitoring** > **Outbound Messages** in the Salesforce user interface. Alternatively, from Setup, click **Create** > **Workflow & Approvals** > **Outbound Messages**, and then click **View Message Delivery Status**. From this page you can perform several tasks:

- View the status of your outbound messages including the total number of attempted deliveries
- View the action that triggered the outbound message by clicking any workflow or approval process action ID.
- Click **Retry** to change the **Next Attempt** date to now. This causes the message delivery to be immediately retried.
- Click **Del** to permanently remove the outbound message from the queue.

Important Security Considerations

In order to safely use outbound messaging, you must ensure that no third party can send messages to the endpoint while pretending to be from Salesforce:

- Lock down the client application's listener to accept requests only from the Salesforce IP range. While this guarantees the message came from Salesforce, it does not guarantee that another customer is not pointing to your endpoint and sending messages. The Salesforce IP ranges are:
  - 204.14.232.0/23 (East Coast Data Center set one)
  - 204.14.237.0/24 (East Coast Data Center set two)
  - 96.43.144.0/22 (Midwest Data Centers)
  - 96.43.148.0/22 (Midwest Data Centers)
  - 204.14.234.0/23 (West Coast Data Center set one)
  - 204.14.238.0/23 (West Coast Data Center set two)
  - 182.50.76.0/22 (Japan Data Center)

  Note: Recommendation: The best and easiest whitelist would be

  - 204.14.232.0/21
  - 96.43.144.0/20

- Use SSL/TLS. Using SSL/TLS provides confidentiality while data is transported across the internet. Without it, a malicious third party can eavesdrop on your data. This issue is especially important if you pass data with privacy requirements and you pass a `SessionId` with the message. Also, we authenticate the certificate presented on connection, ensure that it is from a valid Certificate Authority, and check that the domain in the certificate matches the one Salesforce is trying to connect. This prevents us from communicating with the wrong endpoint.
- If the configuration of your application (endpoint) server's SSL/TLS allows, validate the identity of the Salesforce server when it takes the role of a client to your server, using the Salesforce client certificate. For instructions to download the certificate, see Downloading the Salesforce Client Certificate.
- The organization `Id` is included in each message (see ID Field Type for more information about the Id field type). Your client application should validate that messages contain your organization `Id`.

Understanding the Outbound Messaging WSDL

The rest of this topic examines relevant sections of the outbound messaging WSDL. Your WSDL may differ, depending on the choices you made when you set up outbound messaging for a particular event on a particular object.

notifications()

This section defines the `notifications()` call, which creates an outbound message containing specified fields and values for a particular object or objects, and sends the values to a specified endpoint URL:

```
<schema elementFormDefault="qualified" xmlns="http://www.w3.org/2001/XMLSchema"
 targetNamespace="http://soap.sforce.com/2005/09/outbound">
    <import namespace="urn:enterprise.soap.sforce.com" />
    <import namespace="urn:sobject.enterprise.soap.sforce.com" />

    <element name="notifications">
        <complexType>
            <sequence>
                <element name="OrganizationId" type="ent:ID" />
                <element name="ActionId" type="ent:ID" />
                <element name="SessionId" type="xsd:string" nillable="true" />
                <element name="EnterpriseUrl" type="xsd:string" />
                <element name="PartnerUrl" type="xsd:string" />
                <element name="Notification" maxOccurs="100"
                  type="tns:OpportunityNotification" />
            </sequence>
        </complexType>
    </element>
</schema>
```

Use this table to understand the elements named in the notifications method definition:

Name	Type	Description
`OrganizationId`	ID	ID of the organization sending the message.
`ActionId`	string	The workflow rule (action) that triggers the message.
`SessionId`	string	Optional, a session ID to be used by endpoint URL client that is responding to the outbound message. It is used by the receiving code to make calls back to Salesforce.
`EnterpriseURL`	string	URL to use to make API calls back to Salesforce using the enterprise WSDL.
`PartnerURL`	string	URL to use to make API calls back to Salesforce using the partner WSDL.
`Notification`	Notification	Defined in the next section, contains the object datatype and its `Id`, for example OpportunityNotification or ContactNotification.

The Notification datatype is defined in the WSDL. In the following example, a Notification for opportunities is defined, based on the Notification entry of the `notifications()` call definition:

```
<complexType name="OpportunityNotification">
    <sequence>
        <element name="Id" type="ent:ID" />
        <element name="sObject" type="ens:Opportunity" />
    </sequence>
</complexType>
```

Each object element (in our example, opportunities) contains the subset of the fields that you selected when you created the outbound message. Each message Notification also has the object ID, and this needs to be used to track redelivery attempts of notifications you've already processed.

notificationsResponse

This element is the schema for sending an acknowledgement (ack) response to Salesforce.

```
            <element name="notificationsResponse">
                <complexType>
                    <sequence>
                        <element name="Ack" type="xsd:boolean" />
                    </sequence>
                </complexType>
            </element> //This section is the last in the types definition section.
```

You acknowledge all notifications in the message if there is more than one.

Building a Listener

Once you have defined an outbound message and configured an outbound messaging endpoint, download the WSDL and create a listener:

1. Right-click **Click for WSDL** and select Save As to save the WSDL to a local directory with an appropriate file name. For example, for an outbound message that deals with leads, you could name the WSDL file `leads.wsdl`.
2. Unlike the enterprise or partner WSDLs, which describe the messages the client sends to Salesforce, this WSDL defines the messages that Salesforce will send to your client application.
3. Most Web services tools will generate stub listeners for you, in much the same way as they generate a client stub for the enterprise or partner WSDL. Look for a server side stub option.

   For example, for .Net 2.0:

   a. Run `wsdl.exe /serverInterface leads.wsdl` with .Net 2.0. This generates `NotificationServiceInterfaces.cs`, which defines the notification interface.
   b. Create a class that implements `NotificationServiceInterfaces.cs`.
   c. You implement your listener by writing a class that implements this interface. There are a number of ways to do this. One simple way is to compile the interface to a DLL first (DLLs need to be in the `bin` directory in ASP.NET).

   ```
   mkdir bin
   csc /t:library /out:bin\nsi.dll NotificationServiceInterfaces.cs
   ```

   Now write an ASMX based Web service that implements this interface. For example, in `MyNotificationListener.asmx`:

   ```
   <%@WebService class="MyNotificationListener" language="C#"%>
   class MyNotificationListener : INotificationBinding
   {             public notificationsResponse notifications(notifications n)
                 {
                        notificationsResponse r = new notificationsResponse();
                        r.Ack = true;
                        return r;
                 }
   }
   ```

   This example is a simple implementation, actual implementations will be more complex.
   d. Deploy the service by creating a new virtual directory in IIS for the directory that contains the `MyNotificationListener.asmx`.
   e. You can now test that the service is deployed by viewing the service page with a browser. For example, if you create a virtual directory `salesforce`, you'd go to `http://localhost/salesforce/MyNotificationListener.asmx`.

The process for other Web service tools is similar, please consult the documentation for your Web service tool.

Your listener must meet these requirements:

- Must be reachable from the public Internet.
- For security reasons, Salesforce restricts the outbound ports you may specify to one of the following:
  - ◊ 80: This port only accepts HTTP connections.
  - ◊ 443: This port only accepts HTTPS connections.
  - ◊ 1024–66535 (inclusive): These ports accept HTTP or HTTPS connections.
- To be valid, the common name (CN) of the certificate must match the domain name for your endpoint's server, and the certificate must be issued by a Certificate Authority trusted by Java 2 Platform, Standard Edition (J2SE) 5.0 (JDK 1.5).
- If your certificate expires, message delivery will fail.

Warning: To avoid an infinite loop of outbound messages that trigger changes that trigger more outbound messages, the user who updates the objects should **not** have the “Send Outbound Messages” permission.

Chapter 15

Data Loading and Integration

In this chapter ...

- Choosing the Right API
- Client Application Design
- Salesforce Settings
- Best Practices with Any Data Loader
- Integration and Single Sign-On

If you need to load large volumes of data (hundreds of thousands to millions of records), there are a number of factors you must consider. Use the topics in this section to become familiar with issues of choosing an API, client application design, organization configuration, and data loader best practices.

Choosing the Right API

The first decision that you must make is which API to use for your data loading process.

SOAP API

Use SOAP API to create, retrieve, update or delete records, such as accounts, leads, and custom objects. With more than 20 different calls, SOAP API also allows you to maintain passwords, perform searches, and much more. Use SOAP API in any language that supports Web services.

REST API

REST API provides a powerful, convenient, and simple REST-based Web services interface for interacting with Salesforce. Its advantages include ease of integration and development, and it's an excellent choice of technology for use with mobile applications and Web projects. However, if you have a large number of records to process, you may wish to use Bulk API, which is based on REST principles and optimized for large sets of data.

Bulk API

Bulk API is based on REST principles and is optimized for loading or deleting large sets of data. You can use it to query, insert, update, upsert, or delete a large number of records asynchronously by submitting batches which are processed in the background by Salesforce.

SOAP API, in contrast, is optimized for real-time client applications that update small numbers of records at a time. Although SOAP API can also be used for processing large numbers of records, when the data sets contain hundreds of thousands of records, it becomes less practical. Bulk API is designed to make it simple to process data from a few thousand to millions of records.

Client Application Design

Although the Bulk API is the best choice for loading large numbers of records, you can also use the SOAP-based API. There are many ways you can design your application to improve the speed of data loads:

- **Use persistent connections.** Opening a socket takes time, mostly when opening a socket stems from the SSL/TLS negotiation. Without SSL or TLS, the API request would not be secure. Included in the HTTP 1.1 specification is support for reusing sockets among requests (persistent connections) instead of having to re-open a socket per request as in HTTP 1.0. Whether your client supports persistent connections depends on the SOAP stack you are using. By default, .NET uses persistent connections. If you change the configuration to use the Apache http-commons libraries, your client will be compliant with the HTTP 1.1 specification and use persistent connections.

  For information about HTTP 1.1, see HTTP Persistent Connections and `http://www.w3.org/Protocols/rfc2616/rfc2616-sec8.html#sec8.1`.

- **Minimize the number of requests.** There is some processing associated with each request, so to save time your client should batch as many records per request as possible. Set `batchSize` to the limit of 2,000. If that is not the most efficient batch size, the API will change it. For more information about setting batch sizes, see QueryOptions.
- **Minimize the size of the requests.** Your client application should send as many records per request as possible, but it should also send as small a request as possible to reduce network transmission time. To minimize the request size, use compression on both the request and the response. Gzip is the most popular type of compression, and there are multiple posts on the community boards at the Force.com Developer Boards that describe how to implement compression with different SOAP stacks. The full Gzip analysis and discussion is available at Simon Fell's blog: `http://www.pocketsoap.com/weblog/2005/12/1583.html`.

- **Do Not Design a Multi-Threaded Client Application.** Multi-threading is not allowed for a single client application using the SOAP-based API.

Salesforce Settings

Most processing takes place in the database. Setting these parameters correctly will help the database process as quickly as possible:

- **Enable or Disable the Most Recently Used (MRU) functionality.** Records marked as most recently used (MRU) are listed in the "Recent Items" section of the sidebar in the Salesforce user interface. Check that you are not enabling it for calls where it is not needed.

  In API version 7.0 and above, MRU functionality is disabled by default. To enable the MRU functionality, create this header and set the `updateMru` to `true`. The following sample shows how to use MRU functionality:

```
public void mruHeaderSample() {
  connection.setMruHeader(true);
  Account account = new Account();
  account.setName("This will be in the MRU");
  try {
    SaveResult[] sr = connection.create(new SObject[]{account});
    System.out.println("ID of account added to MRU: " +
      sr[0].getId());
  } catch (ConnectionException ce) {
    ce.printStackTrace();
  }
}
```

- **Log in as a user with the "Modify All Data" permission to avoid sharing rules.** If the client application logs in as a user who has access to data via a sharing rule, then the API must issue an extra query to check access. To avoid this, log in as a user with the "Modify All Data" permission. In general, fewer sharing rules quickens load speeds, as there are fewer operations that have to be performed when setting properties such as ownership.

  Alternatively, you can set organization-wide defaults for some objects as public read/write for the duration of the load. For more information, see "Setting Your Organization-Wide Sharing Defaults" in the Salesforce online help.

- **Avoid workflow or assignment rules.** Anything that causes a post-operation action slows down the load. You can temporarily disable automatic rules if the loaded objects are exempt from them.
- **Avoid triggering cascading updates.** For example, if you update the owner of an account, the contacts and opportunities associated with that account may also require updates. Instead of updating a single object, the client application must access multiple objects, which slows down the load.

The Force.com Data Loader is a good reference for data loading. It disables the MRU, uses HTTP/1.1 persistent connections, and applies GZIP compression on the request and response. If you are performing a data load, or are looking at a place to start when writing your own Java integration, the Force.com Data Loader can serve as a fast and reliable solution. For more information about the Force.com Data Loader, see: Data Loader Overview in the Salesforce online help.

Best Practices with Any Data Loader

While this section presents a best practice process using the Force.com Data Loader, the general principles apply to any client data loader:

1. Identify which data you will migrate.

You may not want or need to migrate a whole set of data—choose which objects you wish to migrate. For example, you may want to migrate only the contact information from each account, or only migrate account information from a particular division.

2. Create templates for the data.

   Create one template for each object, for example in an Excel worksheet.

   Identify the required fields for each object. In addition to the required fields for each standard object, there may be additional required fields such as those needed to follow business rules, or legacy ID fields. Use this guide or see the page layout definitions in the Salesforce user interface to find out which fields are required on standard objects.

   You may wish to highlight the required fields in red for easier review of the data after you populate the templates.

   You should also identify any ordering dependencies. Objects may have mandatory relationships, for example all accounts have an owner, and all opportunities are associated with an account. The dependencies in these relationships dictate the order of data migration. For Salesforce data, for example, you should load users first, then accounts, then opportunities.

   To identify dependencies, review the related lists and lookup fields in the page layout of the given object, and IDs (foreign keys) in the database.

3. Populate the templates.

   Clean your data before populating the template, and review the data in the templates.

4. Migrate the data.

   Create custom fields to store legacy ID information. Optionally, give the custom field the `External ID` attribute so it will be indexed. This will help maintain relationships, and help you build custom reports for validation.

   Load one record, check the results, then load all records.

5. Validate the data.

   Use all of these techniques to validate your migration:

   - Create custom reports that validate record counts and provide an overall snapshot of migration.
   - Spot check the data.
   - Review exception reports to see what data was not migrated.

6. Re-migrate or update data as needed.

Integration and Single Sign-On

Warning: To avoid getting into an unrecoverable state, do not enable single sign-on for your system administrator account. If you do, and then perform a single sign-on integration that fails, you may not be able to log in again to recover.

Chapter 16

Data Replication

In this chapter ...

- API Calls for Data Replication
- Scope of Data Replication
- Data Replication Steps
- Object-Specific Requirements for Data Replication
- Polling for Changes
- Checking for Structural Changes in the Object

The API supports data replication, which allows you to store and maintain a local, separate copy of your organization's pertinent Salesforce data for specialized uses, such as data warehousing, data mining, custom reporting, analytics, and integration with other applications. Data replication provides you with local control and the ability to run large or ad hoc analytical queries across the entire data set without transmitting all that data across the network.

Use the topics in this section to better understand the best practices for data replication.

API Calls for Data Replication

The API supports data replication with the following API calls:

API Call	Description
getUpdated()	Retrieves the list of objects that have been updated (added or changed) during the specified timespan for the specified object.
getDeleted()	Retrieves the list of objects that have been deleted during the specified timespan for the specified object.

Client applications can invoke these API calls to determine which objects in your organization's data have been updated or deleted during a given time period. These API calls return a set of IDs for objects that have been updated (added or changed) or deleted, as well as the timestamp (Coordinated Universal Time (UTC)—not local—timezone) indicating when they were last updated or deleted. It is the responsibility of the client application to process these results and to incorporate the required changes into the local copy of the data.

Scope of Data Replication

This feature provides a mechanism that targets data replication (one-way copying of data). It does not provide data synchronization (two-way copying of data) or data mirroring capabilities.

Data Replication Steps

The following is a typical data replication procedure for an object:

1. Optionally, determine whether the structure of the object has changed since the last replication request, as described in Checking for Structural Changes in the Object.
2. Call `getUpdated()`, passing in the object and timespan for which to retrieve data.

   Note that `getUpdated()` retrieves the IDs for data to which the logged in user has access. Data that is outside of the user's sharing model is not returned. The API returns the ID of every changed object that is visible to you, regardless of what change occurred in the object. For information on IDs, see ID Field Type.
3. Pass in all IDs in an array. For each ID element in the array, call `retrieve()` to obtain the latest information you want from the associated object. You must then take the appropriate action on the local data, such as inserting new rows or updating existing ones with the latest information.
4. Call `getDeleted()`, passing in the object and timespan for which to retrieve data. Like `getUpdated()`, `getDeleted()` retrieves the IDs for data to which the logged-in user has access. Data that is outside of the user's sharing model is not returned. The API returns the ID of every changed object that is visible to you, regardless of what change occurred in the object. For information on IDs, see ID Field Type.
5. Iterate through the returned array of IDs. Your client application must then take the appropriate action on the local data to remove (or flag as deleted) the deleted objects. If your client application cannot match rows in the local data using the retrieved object ID, then the local data rows either were deleted or were never created, in which case there is nothing to do.
6. Optionally, save the request time spans for future reference. You can do this with the `getDeleted()` latestDateCovered value or the `getUpdated()` latestDateCovered value.

Object-Specific Requirements for Data Replication

The API objects have the following requirements for data replication:

- The `getUpdated()` and `getDeleted()` calls filter the results so that the client application receives IDs for only those created or updated objects to which the logged-in user has access. For information on IDs, see ID Field Type.
- Your client application can replicate any objects to which it has sufficient permissions. For example, to replicate all data for your organization, your client application must be logged in with the "View All Data" permission. For more information, see Factors that Affect Data Access.
- The logged-in user must have read access to the object. For more information, see "Setting Your Organization-Wide Sharing Defaults" in the Salesforce online help.
- The object must be configured to be replicateable (`replicateable` is `true`). To determine whether a given object can be replicated, your application can invoke the `describeSObject()` call on the object and inspect the `replicateable` property in the describeSObjectResult.

Polling for Changes

Client applications typically poll for changed data periodically. Polling involves the following considerations:

- The polling frequency depends on business requirements for how quickly changes in your organization's Salesforce data need to be reflected in the local copy. Some client applications might poll once a day to retrieve changes, while other client applications might poll every five minutes to achieve closer accuracy.
- Deleted records are written to a delete log, which `getDeleted()` accesses. A background process that runs every two hours purges records that have been in an organization's delete log for more than two hours if the number of records is above a certain limit. Starting with the oldest records, the process purges delete log entries until the delete log is back below the limit. This is done to protect Salesforce from performance issues related to massive delete logs. The limit is calculated using this formula:

  ```
  5000 * number of licenses in the organization
  ```

  For example, an organization with 1,000 licenses could have up to 5,000,000 (five million) records in the delete log before any purging took place. If purging has been performed before your `getDeleted()` call is executed, an `INVALID_REPLICATION_DATE` error is returned. If you get this exception, you should do a full pull of the table.

- The API truncates the seconds portion of dateTime values. For example, if a client application submits a timespan between 12:30:15 and 12:35:15 (Coordinated Universal Time (UTC) time), then the API retrieves information about items that have changed between 12:30:00 and 12:35:00 (UTC), inclusive.

  Note: Development tools differ in the way that they handle time data. Some development tools report the local time, while others report only the Coordinated Universal Time (UTC) time. To determine how your development tool handles time values, refer to its documentation.

- We recommend polling no more frequently than every five minutes. There are built in controls to prevent errant applications from invoking the data replication API calls too frequently.
- Client applications should save the timespan used in previous data replication API calls so that the application knows the last time period for which data replication was successfully completed.
- To ensure data integrity on the local copy of the data, a client application needs to capture all of the relevant changes during polling—even if it requires processing data redundantly to ensure that there are no gaps. Your client application can contain business logic to skip processing objects that have already been integrated into your local data.

- Gaps can also occur if the client application somehow fails to poll the data as expected (for example, due to a hardware crash or network connection failure). Your client application can contain business logic that determines the last successful replication and polls for the next consecutive timespan.
- If for any reason the local data is compromised, your client application might also provide business logic for rebuilding the local data from scratch.

Note: You can now use Outbound Messaging to trigger actions instead of polling for them.

Checking for Structural Changes in the Object

In the API, data replication only reflects changes made to object records. It does not determine whether changes have been made to the structure of objects (for example, fields added to—or removed from—a custom object). It is the responsibility of the client application to check whether the structure of a given object has changed since the last update. Before replicating data, client applications can call `describeSObjects()` on the object, and then compare the data returned in the DescribeSObjectResult with the data returned and saved from previous `describeSObjects()` invocations.

Chapter 17

Feature-Specific Considerations

In this chapter ...

- Archived Activities
- Person Account Record Types
- Opportunity Forecast Override Business Rules
- Call Centers and the API
- Implementing Salesforce Integrations on Force.com
- Articles
- Data Categories

Some Salesforce features require special consideration when accessed via the API. Use the topics in this section to learn about the special considerations for activities, person accounts, forecast override business rules, the Call Center, and creating your own apps.

Archived Activities

Salesforce archives activities (tasks and events) that are over a year old.

You can use `queryAll()` to query on all Task and Event records, archived or not. You can also filter on the `isArchived` field to find only the archived objects. You cannot use `query()` as it automatically filters out all records where `isArchived` is set to `true`. You can update or delete archived records, though you cannot update the `isArchived` field. If you use the API to insert activities that meet the criteria listed below, the activities will be archived during the next run of the archival background process.

Older Events and Tasks are archived according to the criteria listed below. In the Salesforce user interface, users can view archived activities only in the **Printable View** or by clicking **View All** on the Activity History related list or by doing an advanced search. However, in the API, archived activities can only be queried via `queryAll()`.

Activity archive criteria:

- Events with an `ActivityDateTime` or `ActivityDate` value greater than or equal to 365 days old
- Tasks with an `IsClosed` value of `true` and an `ActivityDate` value greater than or equal to 365 days old
- Tasks with an `IsClosed` value of `true`, a blank `ActivityDate` field, and a create date greater than or equal to 365 days ago

Person Account Record Types

Beginning with API version 8.0, a new family of record types on Account objects is available: “person account” record types. The person account record types enable specialized business-to-consumer functionality for those who sell to or do business with individuals. For example, a doctor, hairdresser, or real estate agent whose clients are individuals. For more information about person accounts, see “What is a Person Account?” and “Person Account Behaviors” in the Salesforce online help.

Record types are person account record types if the Account field `IsPersonAccount` is set to `true`. Salesforce provides one default person account record type, PersonAccount, but an administrator can create additional person account record types. Conversely, record types with the Account field `IsPersonAccount` set to `false` are “business account” record types, which are traditional business-to-business (B2B) Salesforce accounts.

When a person account is created (or an existing business account is changed to a person account), a corresponding contact record is also created. This contact record is referred to as a “person contact.” The person contact enables the person account to function simultaneously as both an account and a contact. This is the one and only contact record that can be associated directly with the person account. Also, the ID of the corresponding person contact record is stored in the `PersonContactId` field on the person account.

Review this list of facts about person account record types before working with them.

- You may need to contact your account representative to enable the person account feature.
- You can use a query similar to the following example to find all records with a person account record type:

```
SELECT Name, SobjectType,IsPersonType
FROM RecordType
WHERE SobjectType='Account' AND IsPersonType=True
```

- If you issue a `query()` call against an account, the results return the root object type in the `SojbectType` field. This means that the value returned will always be Account.
- A person contact can be modified, but cannot be created or deleted. Since these kinds of contacts do not have their own record detail page, clients should redirect users to the corresponding person account (Account) page. SOSL results will

not include any of the contact fields enabled when IsPersonAccount is set to true. The contact ReportsToId field is not visible.

- If you delete the account, the contact is also deleted. You cannot directly delete the contact; you must delete the account.
- You can change the record type of an account across record type families (typically performed when migrating business accounts to person accounts, but the reverse operation is also supported). When you change the record type from a business account to a person account, the person contact is created. When you change the record type from a person account to a business account, the person fields are set to null, and the person contact becomes a regular contact with the same parent account it had before the change.

  Note: You cannot change record types across record type families in the Salesforce user interface.

- If you change the record type of a business account to a person account using either update() or upsert(), you cannot make any other changes to fields in that account in the same call; if attempted, the fault INVALID_FIELD_FOR_INSERT_UPDATE will result. However, you can change record type values from one person account record type to another, or from one business account record type to another, in the same call with other changes.
- When converting a business account to a person account, there must be a one-to-one relationship between each business account record and its corresponding contact record. Furthermore, fields common to both records such as Owner and Currency must have identical values.
- Workflow and validation formulas do not fire during a change in record types from or to person accounts.
- When you change a business account to a person account, valid records will be changed and invalid records will show an error in the results array.
- When you change a person account to a business account, no validation is performed.
- describeLayout() for version 7.0 and below will return the default business account record type as the default record type even if the tab default is a person account record type. In version 8.0 and after, it will always be the tab default.
- describeLayout() for version 7.0 and below will not return any person account record types.
- describeSObject() for version 7.0 and below will show Account objects as not createable if the profile does not have access to any business record types.
- After conversion, the new person accounts will have unique one-to-one relationships with the contact records that formed them. As is true for all person accounts, no other contacts can be associated to a person account.
- After conversion, any existing account field history information remains on the person accounts. Any existing contact field history information is retained on the contact, but is not added to the person accounts field history.

For more information about person accounts, see the Salesforce online help.

Opportunity Forecast Override Business Rules

Customizable forecasting is the preferred way to track revenue data in Salesforce. If you have customizable forecasting enabled for your organization, users with the “Override Forecast” permission can override forecast amounts for themselves and their direct reports. For more information, see “Overriding Customizable Forecasts” in the Salesforce online help.

You can use the following objects to retrieve opportunity forecast override information.

- OpportunityOverride
- LineitemOverride

These read-only objects keep an up-to-date record of the current values related to forecasts, whether those values are inherited directly from the opportunity or reflect an opportunity forecast override. Inherited values are stored in these objects, separate from the opportunity object, to improve performance.

The forecast is computed differently depending on whether or not there is an override record.

- If an override record exists (whether or not a particular value is overridden), the following values are used to compute the forecast for the `OwnerId` specified in the record.
  - ◊ `Amount`
  - ◊ `Quantity`
  - ◊ `Forecast Period`
  - ◊ `Forecast Category`
  - ◊ `Unit Price` on LineitemOverride
  - ◊ OpportunityOverride or LineitemOverride. These objects represent the user's view of the Opportunity or OpportunityLineItem with regard to forecasting, and may reflect overridden values from an opportunity owner or Forecast Manager below the specified user in the Forecast Hierarchy.
- If an override record does not exist, then the values stored in the Opportunity or OpportunityLineItem record are used, and no override information appears to the user.

In the Forecast Hierarchy, one user at each non-leaf level is designated as the Forecast Manager. The Forecast Manager sees the Opportunity Forecast Related List on the Opportunity Detail page when he or she views an opportunity owned by a subordinate user (if that user has the Allow Forecasting permission). A Forecast Manager's forecasts include contributions from those same subordinates. Other users in the same role as a Forecast Manager only see their own opportunities in their forecasts.

Note: Keep in mind that this section discusses both forecast managers and opportunity owners. When we describe traversals from a user and travelling along the forecast hierarchy, it is always in order, unless otherwise specified. For more information about forecast managers and the forecast hierarchy, see the Salesforce online help.

If your organization has territory management enabled, the territory hierarchy drives forecast data. See Territory Management Overview in the Salesforce online help.

OpportunityOverride Lifecycle

OpportunityOverride records are created, updated, or deleted only under certain conditions, and only for relevant users.

Note: Because LineitemOverride records always have a parent record in the OpportunityOverride object with the same `OpportunityId` and `OwnerId`, this section explains the lifecycle of the OpportunityOverride records first, then explains any additional behaviors relevant only to LineitemOverride records.

Creating Opportunity Overrides

Forecast managers can override a forecast-related value (such as `Amount`, `Quantity`, Forecast Period, and `ForecastCategory`) by editing the Opportunity Forecasts related list on an opportunity owned by a subordinate user. Also, the opportunity owner can change Opportunity forecast-related values (such as `ForecastCategory`, `StageName`, `Amount`, and `CloseDate`) on an Opportunity that he or she owns. `ForecastCategory` has additional rules governing it; see Special Case: Forecast Category for more information.

When a user sets the first override for a particular Opportunity record, OpportunityOverride records are created and appropriate values are written for the opportunity owner and every forecast manager above the Opportunity owner in the forecast hierarchy. The values stored in each user's record depend on their location in the hierarchy—users below the user who made an override will have the original values. Other users will have the overrides, plus the Opportunity values for anything not overridden.

When subsequent override values are set, the new values are written to the OpportunityOverride records of the user who set the override and other forecast managers above in the forecast hierarchy, until an existing override value is reached.

Quantity can only be overridden in the Salesforce user interface if Quantity Forecasting is enabled on the Forecast Settings page, and `Amount` can only be overridden if Revenue Forecasting is enabled.

Note: Any changes that an opportunity owner can make on the Opportunity Forecast edit page are applied to the Opportunity record as well as the relevant OpportunityOverride records.

Updating Opportunity Values and Opportunity Override Values

Whenever an opportunity forecast-related value is updated on the Opportunity record itself, the corresponding records in the OpportunityOverride for the Opportunity owner and forecast managers above the owner in the forecast hierarchy are updated, until the first override value is encountered. The Opportunity owner cannot override anything except `ForecastCategory`, but he or she can edit the `CloseDate` and `StageName` on the Opportunity Forecast edit page, in addition to `ForecastCategory`. All three fields correspond to the Opportunity record, and the result is the same whether they make the changes on the Opportunity Forecast edit page or the Opportunity edit page.

Changes to the following Opportunity fields also affect records in the OpportunityOverride.

- `CloseDate` affects the forecast `PeriodId` if the new `CloseDate` falls into a different Forecast Period than the old `CloseDate`. In this case, the OpportunityOverride `PeriodId` is updated on the opportunity override record of the opportunity owner, and all forecast managers above the owner in the forecast hierarchy, until the first overriden `PeriodId` is encountered.
- CurrencyType `IsoCode` changes always cause an update to the opportunity owner's OpportunityOverride. No other object is updated unless `Amount` changes, which is often the case.
- If territory management is enabled, territory changes on the Opportunity affect OpportunityOverride records. An Opportunity owner's OpportunityOverrides are updated with the new territory, and OpportunityOverride records are inserted or updated for forecast managers above the Opportunity owner in the forecast hierarchy. OpportunityOverride records are also deleted for any forecast manager whose territory (stored on the override record) is no longer above the Opportunity owner in the new opportunity territory. That is, even if the Opportunity owner remains a subordinate of a forecast manager before and after the update, if they roll up into a different territory, that forecast manager loses the OpportunityOverrides for the old territory and new records are created for the new territory.
- When an Opportunity is set to `Closed Won` (the `StageName` value is `Closed Won`), the `AmountInherited`, `QuantityInherited`, and `PeriodInherited` fields in OpportunityOverride are set to `true` and the corresponding values are updated to match the Opportunity, if they are different. The OpportunityOverride `ForecastCategoryInherited` flag is also set to `true` and the Opportunity `ForecastCategory` is set to `Closed`, unless it has been overridden to `Omitted`. `Omitted` is the only valid `ForecastCategory` override for a `Closed Won` opportunity.
- When an Opportunity is set to `Closed Lost`, the `Amount` and `PeriodInherited` flags are set to `true`, and the corresponding values are updated to match the Opportunity, if they are different, and the `ForecastCategoryInherited` flag is set to `true` and the `ForecastCategory` is set to `Omitted` if the current value is anything other than `Omitted`.

Deleting Opportunity Overrides

OpportunityOverride records are not deleted unless one of the following triggering events occurs.

- An Opportunity is deleted.
- A change is made in the forecast hierarchy, such that a particular OpportunityOverride owner is no longer above the Opportunity owner or is no longer the forecast manager in the relevant role or territory.
- An Opportunity is transferred to a new owner or territory, individually or as part of an account transfer.
- The “Allow Forecasting” permission is removed from a user.

When an Opportunity is transferred to a new owner, an OpportunityOverride record for the new owner is added as needed, and OpportunityOverride records are deleted for all users that are not forecast managers above the new owner in the forecast hierarchy, possibly including the previous Opportunity owner. Also, OpportunityOverride records are updated or inserted as necessary for all forecast managers above the new owner in the forecast hierarchy. Restrictions on Opportunity owners are strictly enforced during a transfer. If the new owner was in the forecast hierarchy of the previous owner, and had made overrides that are no longer valid as the opportunity owner, such as an `Amount` override, then the corresponding inherited flag is set to `true` and the value is refreshed from the opportunity.

Special Case: Forecast Category

An additional rule applies to `ForecastCategory` values: If you set the `ForecastCategory` in a new opportunity that is not the default forecast category for the stage selected, or update an existing `ForecastCategory` on an opportunity, then OpportunityOverride records are created for relevant users as described above. Since only Opportunity owners can update the `ForecastCategory` on an opportunity record directly, this scenario is treated as a `ForecastCategory` override by the opportunity owner (the `ForecastCategoryInherited` flag on that user's OpportunityOverride record is `false`).

LineitemOverride Object Lifecycle

Additional rules apply to line item overrides, as explained in the rest of this section.

Creating Line Item Overrides

The full set of LineitemOverride records for each OpportunityOverride record is always created to mirror the line item records that exist for the Opportunity record. Whether a user makes an override at the opportunity level or line item level, or even just edits Forecast Category on the Opportunity Edit (only opportunity owners can do this), the end result is the same in terms of the Opportunity and LineitemOverride records that are created.

Updating Line Item Overrides

All information from Updating Opportunity Values and Opportunity Override Values applies to updating LineitemOverride records, with the following changes.

- Forecast Category overrides at the opportunity level are cascaded down into child LineitemOverride records, if the Forecast Category has not been overridden for a particular line item by that user or a subordinate user. That is, when a user establishes an opportunity-level Forecast Category override, any line item-level Forecast Category overrides by a subordinate forecast manager will take precedence when updating line item Forecast Category values for this user and any superiors. The converse is not true. Forecast Category on an OpportunityOverride record is never updated in response to a Forecast Category override on a LineitemOverride record by any user. When a user sets a Forecast Category override at the line item level, the override value is written to the LineitemOverride records of the user who set the override and other forecast managers above that user in the forecast hierarchy, until an existing override value is reached.
- Opportunity line item values for `Unit Price`, `Total Price` and `Quantity`, if changed, are saved on the LineitemOverride records for the opportunity owner and above in the forecast hierarchy, until the first overriden value is encountered for the field. `Quantity` can only be overridden in the Salesforce user interface if Quantity Forecasting is enabled on the Forecast Settings page. `Unit Price` and `Total Price` can only be overridden if Revenue Forecasting is enabled.

When `Quantity` or `Unit Price` are overridden, the `Total Price` is computed and written to the relevant LineitemOverride record. When `Total Price` is overridden, but not `Unit Price`, the `Unit Price` is computed and written to the relevant LineitemOverride records. If both are overridden, no computation occurs. The computation applies to the LineitemOverride record that contains the override value and related LineitemOverride records that inherit the overridden value. That is, for a given LineitemOverride record, we compute `Total Price` or `Unit Price` according to the above rules, if an override is made on the current record or if the record inherited an override made by a subordinate forecast manager.

Note that if you have selected the value **Schedule Date** from Setup at **Customize** > **Forecasts** > **Forecast Settings** > **Forecast Date** picklist, you cannot override the following values.

- If forecasting by Revenue, or Revenue and Quantity, and the OpportunityLineItem record has a Revenue schedule, you cannot override `Unit Price` and `Total Price`.
- If forecasting by Quantity, or Revenue and Quantity, and the OpportunityLineItem records have Quantity schedules, you cannot override Quantity.

Deleting Line Item Overrides

Whenever a line item is deleted, LineitemOverride records relating to that line item are deleted for all users, similar to opportunities. Also, opportunity transfers and forecast hierarchy changes affect line item overrides the same way they affect opportunity overrides. That is, if an OpportunityOverride record is deleted, all the child LineitemOverride records, matching on `OpportunityId` and `OwnerId`, are also deleted.

Call Centers and the API

The API provides access to information about computer–telephony integration (CTI) call centers with the `describeSoftphoneLayout()` call. You must have the CTI feature enabled for your organization. Contact your account representative for assistance.

The API supports limited access to call center-related objects, including being able to create call centers, and create or modify additional numbers for the call center.

Topic	Description
CallCenter	Call Center object description, including fields and usage.
AdditionalNumber	Configuration settings that allow you to add an additional number if it cannot easily be categorized as a user, contact, lead, account, or any other object. Examples include phone queues or conference rooms.

In addition, several fields have been added to existing objects to support call centers. The following fields provide configuration settings for operation of a call center.

Object Name	Field Name	Field Type	Field Properties	Description
OpenActivity ActivityHistory Task	`CallDisposition`	string	Create (Task only) Filter Nillable Update (Task only)	Represents the result of a given call, for example, “we'll call back,” or “call unsuccessful.” Limit is 255 characters. For the Task object, corresponds to the Salesforce user interface label **Call Result**. You can also create and update values for this field in Task.
OpenActivity ActivityHistory Task	`CallDurationInSeconds`	int	Create (Task only) Filter Nillable Update (Task only)	Duration of the call in seconds. For Task, you can also create and update values for this field.
OpenActivity ActivityHistory	`CallObject`	string	Filter Nillable	Name of a call center. Limit is 255 characters. For Task, you can also create and update values for this field.

Object Name	Field Name	Field Type	Field Properties	Description
Task			Update (Task only)	
OpenActivity ActivityHistory Task	`CallType`	picklist	Create (Task only) Filter Nillable Restricted picklist Update	The type of call being answered: Inbound, Internal, or Outbound. For Task, you can also create and update values for this field.
User	`CallCenterId`	reference	Create Filter Nillable Update	The unique identifier for the call center associated with this user.
User	`UserPermissionsCallCenterAutoLogin`	boolean	Create Update	Indicates whether a user will be automatically logged in to a call center when logging in to the Salesforce application (`true`) or not (`false`).

Implementing Salesforce Integrations on Force.com

You can implement your Salesforce integrations or other client applications, on the Force.com platform by creating a Force.com AppExchange app.

1. Create a WebLink that passes the user session ID and the API server URL to an external site:

   ```
   https://www.your_tool.com/test.jsp?sessionid={!API_Session_ID}&url={!API_Partner_Server_URL_80}
   ```

   Use `https` to ensure your session ID cannot be detected.
2. The page pointed to in the step above takes the session ID and uses it to call back to the API. Use `getUserInfo()` to return the `userID` associated with the session and related information. If needed, you can also use retrieve on the User object to retrieve any additional information you need about the user.
3. Maintain a cross-reference between the `UserId` or username and the corresponding user ID in your system, which you can do using a WebLink that is executed when the user clicks on a tab, or a WebLink on the page layout.
4. Package and upload this app using the instructions in the Salesforce online help topic “Preparing Your Apps for Distribution.”

Accessing Salesforce Data Using the API and OAuth

Salesforce supports OAuth versions 1.0A and 2.0 for SOAP API requests.

For OAuth version 1.0.A, after a third-party has successfully negotiated a session with Salesforce (using an already defined connected app and the OAuth protocol) and has a valid AccessToken, a client application can request to access Salesforce data using the API.

The following contains more detailed steps regarding accessing data for developers who are using a connected app to request Salesforce data.

1. The consumer makes an HTTPS POST request to Salesforce.

- The URL must have the following format:
  `https://login.salesforce.com/services/OAuth/`*`type`*`/`*`api-version`*.

  `type` must have one of the following values.

  ◊ `u`—Partner WSDL
  ◊ `c`—Enterprise WSDL

  `api-version` must be a valid API version.

- The authorization header must have the following parameters.

  ◊ `oauth_consumer_key`
  ◊ `oauth_token`
  ◊ `oauth_signature_method`
  ◊ `oauth_signature`
  ◊ `oauth_timestamp`
  ◊ `oauth_nonce`
  ◊ `oauth_version` (optional, must be "1.0" if included)

2. Salesforce validates the request and sends a valid session ID to the consumer. The response header includes the following.

```
<response>
    <metadataServerUrl>https://na1.salesforce.com/services/Soap/m/17.0/00D300000006qjK
    </metadataServerUrl>
    <sandbox>false</sandbox>
    <serverUrl>https://na1.salesforce.com/services/Soap/u/17.0/00D300000006qjK
    </serverUrl>
    <sessionId>00D300000006qrN!AQoAQJTMzwTa67tGgQck1ng_xgMSuWVBpFwZ1xUq2kLjMYg6Zq
       GTS8Ezu_C3w0pdT1DMyHiJgB6fbhhEPxKjGqlYnlROIUs1</sessionId>
</response>
```

After a consumer using OAuth version 2.0 has an access token, the method of using the token depends on the API being used.

- For the REST API, use an HTTP authorization header with the following format `Authorization: Bearer` ***`Access_Token`***.
- For the SOAP API, the access token is placed in the SessionHeader SOAP authentication header.
- For the identity URL, use either an HTTP authorization header (as with the REST API) or use as an HTTP parameter `oauth_token`.

For more information, see Authenticating Apps with OAuth in the Salesforce online help.

Partners, who wish to get an OAuth consumer Id for authentication, can contact salesforce.com

Articles

Articles capture information about your company's products and services that you want to make available in your knowledge base. Articles in the knowledge base can be classified by using one or more data categories to make it easy for users to find the articles they need. Administrators can use data categories to control access to articles.

Articles are based on article types, which rely on:

- Article-type layouts to organize the content in sections.
- Article-type templates to render articles.

Every article is managed in a publishing cycle.

Article Type

All articles in Salesforce Knowledge are assigned to an *article type*. An article's type determines the type of content it contains, its appearance, and which users can access it. For example, a simple FAQ article type might have two custom fields, `Question` and `Answer`, where article managers enter data when creating or updating FAQ articles. A more complex article type may require dozens of fields organized into several sections. Using layouts and templates, administrators can structure the article type in the most effective way for its particular content. User access to article types is controlled by permissions. For each article type, an administrator can grant "Create," "Read," "Edit," or "Delete" permissions to users. For example, the article manager might want to allow internal users to read, create, and edit FAQ article types, but let partner users only read FAQs.

Article-Type Layout

An *article-type layout* enables administrators to create sections that organize the fields on an article, as well as choose which fields users can view and edit. One layout is available per article type. Administrators modify the layout from the article-type detail page.

Article-Type Template

An *article-type template* specifies how the sections in the article-type layout are rendered. An article type can have a different template for each of its four channels. For example, if the Customer Portal channel on the FAQ article-type is assigned to the Tab template, the sections in the FAQ's layout appear as tabs when customers view an FAQ article. For the Table of Contents template, the sections defined in the layout appear on a single page (with hyperlinks) when the article is viewed. Salesforce provides two standard article-type templates, Tab and Table of Contents. Custom templates can be created with Visualforce.

Channel

A channel refers to the medium by which an article is available. Salesforce Knowledge offers four channels where you can make articles available.

- Internal App: Salesforce users can access articles in the Articles tab depending on their role visibility.
- Customer: Customers can access articles if the Articles tab is available in the portal. Customer Portal users inherit the role visibility of the manager on the account.
- Partner: Partners can access articles if the Articles tab is available in the portal. Partner portal users inherit the role visibility of the manager on the account.
- Public Knowledge Base: Articles can be made available to anonymous users by creating a public knowledge base using the *Sample Public Knowledge Base for Salesforce Knowledge* app from the AppExchange. Creating a public knowledge base requires Sites and Visualforce.
- Your own website. Articles can be made available to users through your company website.

Publishing Cycle

Salesforce Knowledge Articles move through a publishing cycle from their creation to their deletion. The publishing cycle includes three different statuses: `Draft` is the stage when a new article is being created or an existing one is being updated. Articles with the `Online` status are draft articles that have been published and are now available to their different channels. Eventually, when a published article is at the end of its life, it can be moved to the `Archived` status or sent back to `Draft` to be updated in a subsequent version.

Working with Articles in the API

Articles are available through the KnowledgeArticleVersion and KnowledgeArticle objects in the API. They both represent an article but provide different capabilities.

KnowledgeArticleVersion

Every new draft article in Salesforce Knowledge has a version number. When an article is published and you want to update it, you can create a new `Draft` with a distinct version number. Each version has its own ID. Once the updated version is ready to be published, it replaces the former one and updates the version number. You can access the content of an article version using the KnowledgeArticleVersion object and filter on its Draft or Online status. For example, the following query returns the title of the Draft version of all the articles across all article types in United States English:

```
SELECT Title
FROM KnowledgeArticleVersion
WHERE PublishStatus='Draft'
AND language ='en_US'
```

You can change the language to return any other language your knowledge base supports, see "Setting up a Multilingual Knowledge Base" in the Salesforce online help.

Articles are also auto-assigned an Article Number, which is not a unique identifier to an individual article, but an identifier to a master article and all of its available translations.

Note: Both the master version (the Knowledge article with IsMasterLanguage = 1) and the translations are KnowledgeArticleVersion objects.

KnowledgeArticle

Unlike KnowledgeArticleVersion, the ID of a KnowledgeArticle record is identical irrespective of the article's version (status). However, KnowledgeArticle doesn't give access to the article's fields and can't be used in a SOQL clause. Where the KnowledgeArticleVersion object provides API access to an article's custom field values, the KnowledgeArticle object provides API access to an article's metadata fields.

The article record is the parent container of all versions of an article, whatever the publishing status (draft, published, archived) and the language. While KnowledgeArticle and KnowledgeArticleVersion represent any article in the Knowledge Base, <Article Type>__ka and <Article Type>__kav are the concrete representation of respectively KnowledgeArticle ("__ka" suffix) and KnowledgeArticleVersion ("__kav" suffix) for a specific article type. For example, Offer__kav gives access to every Offer articles. KnowledgeArticle and <Article Type>__ka give access to an article independent of its version. KnowledgeArticleVersion and <ArticleType>__kav enables you to retrieve a specific article version and all its standard (KnowledgeArticleVersion) and custom (<ArticleType>__kav) fields. The following query returns the title for all the published offers in United States English:

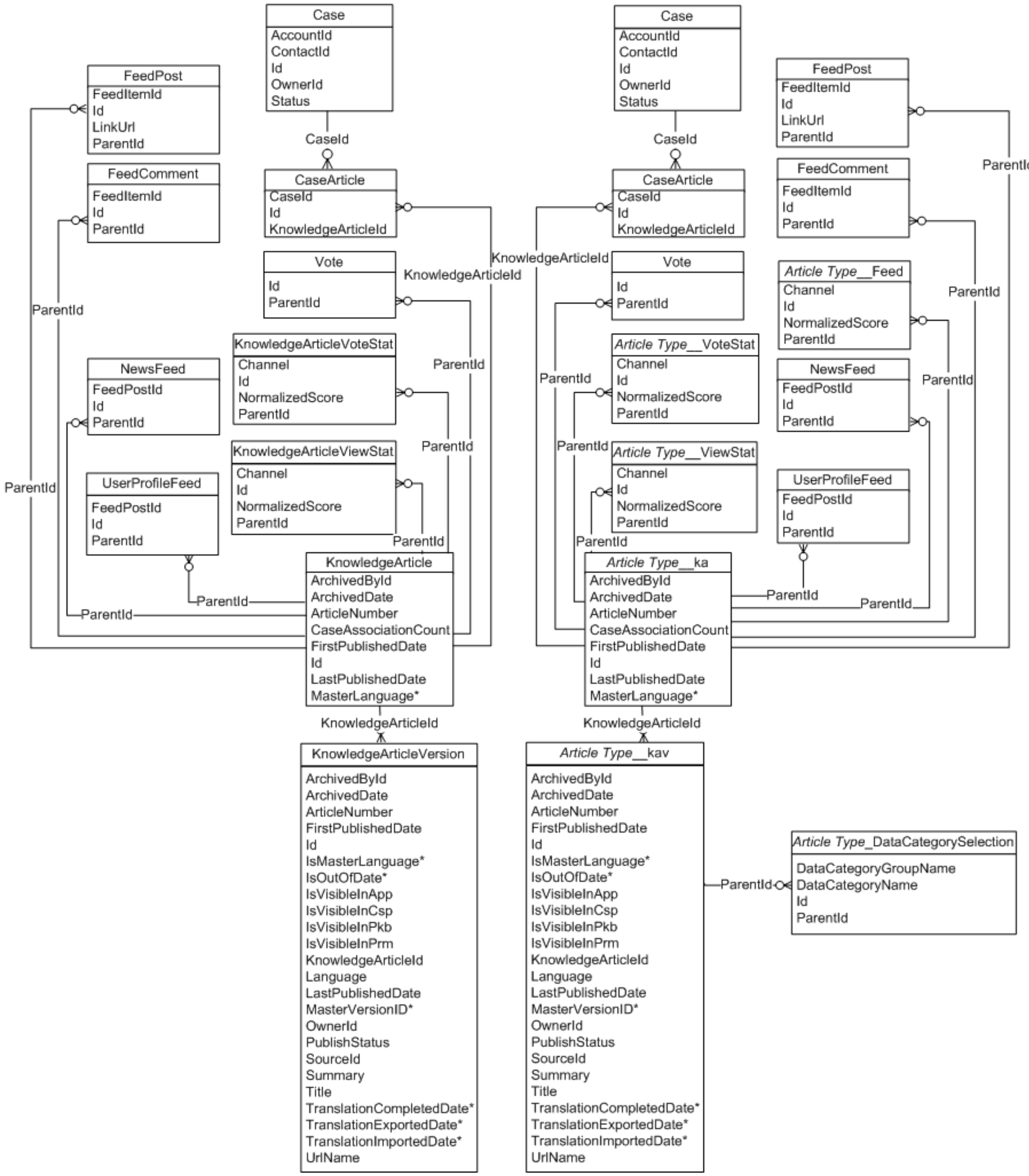
```
SELECT Title
FROM Offer__kav
WHERE PublishStatus='online'
AND language ='en_US'
```

The following table lists API resources for working with articles:

Name	Type	Description
Article Type__DataCategorySelection	Object	A data category selection represents a data category that classifies an article.
Article Type__ka	Object	Gives access to an article from a specific article type independent of its version. This object is read–only and can't be used in a SOQL clause or in a WITH DATA CATEGORY *DataCategorySpec* SOSL clause. For more information, see KnowledgeArticle.
Article Type__kav	Object	Gives access to all articles from a specific article type depending on their version. This object gives access to the fields available in KnowledgeArticleVersion. For more information, see KnowledgeArticleVersion.
Article Type__Feed	Object	Represents a single feed item in the feed displayed on the detail page for an article.
Article Type__ViewStat	Object	Provides statistics on the number of views for an article from a specific article type. For more information, see KnowledgeArticleViewStat.
Article Type__VoteStat	Object	Provides the weighted rating for an article from a specific article type on a scale of 1 to 5. For more information, see KnowledgeArticleVoteStat.
CaseArticle	Object	Represents the association between a Case and a KnowledgeArticle.
FeedComment	Object	Represents a comment added to a feed by a user. Represents a comment added to a feed by a user.
FeedItem	Object	FeedItem represents an entry in the feed, such as changes in a record feed, including text posts, link posts, and content posts.
KnowledgeArticle	Object	Gives access to an article independent of its version. This object is read–only and can't be used in a SOQL clause or in a WITH DATA CATEGORY *DataCategorySpec* SOSL clause.
KnowledgeArticleVersion	Object	Provides a global view of standard article fields across all article types depending on their version.
KnowledgeArticleViewStat	Object	Provides statistics on the number of views for the specified article across all article types.
KnowledgeArticleVoteStat	Object	Provides the weighted rating for the specified article on a scale of 1 to 5 across all article types.
NewsFeed	Object	Represents a single feed item on a user's home tab. A Chatter feed shows recent changes to records that the user is following.
UserProfileFeed	Object	Represents a user profile feed, which tracks all actions by a user on records that can be tracked in a feed. This feed is displayed on the user profile page.
WITH DATA CATEGORY ***filteringExpression***	SOQL clause	Filters articles depending on their status in the publishing cycle and their data categories. For more information, see the *Salesforce SOQL and SOSL Reference Guide*.
WITH DATA CATEGORY ***DataCategorySpec***	SOSL clause	Finds articles based on their categorization. For more information, see the *Salesforce SOQL and SOSL Reference Guide*.

Salesforce Knowledge Objects

This entity relationship diagram (ERD) illustrates relationships between the Salesforce Knowledge objects.



For field definitions, see its object's page within this guide. Fields with an asterisk (*) are only accessible if your knowledge base supports multiple languages, see “Setting up a Multilingual Knowledge Base” in the Salesforce online help.

Data Categories

Data categories are organized by category group and let:

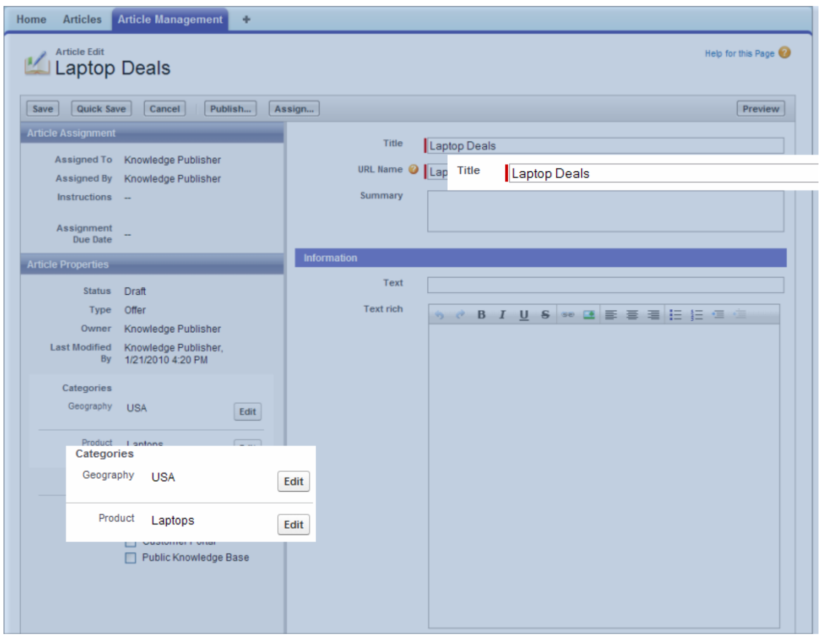
- Users classify and find records.
- Administrators control access to records.

Data categories can be used by Salesforce Knowledge (articles) and answers communities (questions).

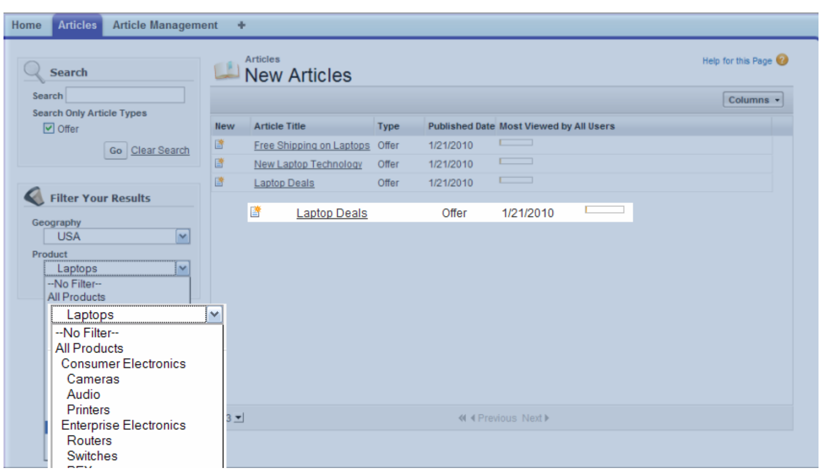
Data Categories and Articles

Salesforce Knowledge supports a five-level hierarchy of data categories within each category group. You can classify articles in the knowledge base according to multiple categories that make it easy for users to find the articles they need. For example, to classify articles by sales regions and business units, create two category groups, Sales Regions and Business Units. The Sales Regions category group could consist of a geographical hierarchy, such as All Sales Regions as the top level, North America, Europe, and Asia at the second level, and so on.

The figure below shows a knowledge base administrator's view of an article about laptop deals. Using the article edit page, the administrator has classified the article with `Laptops` in the `Product` category group, and `USA` in the `Geography` category group.

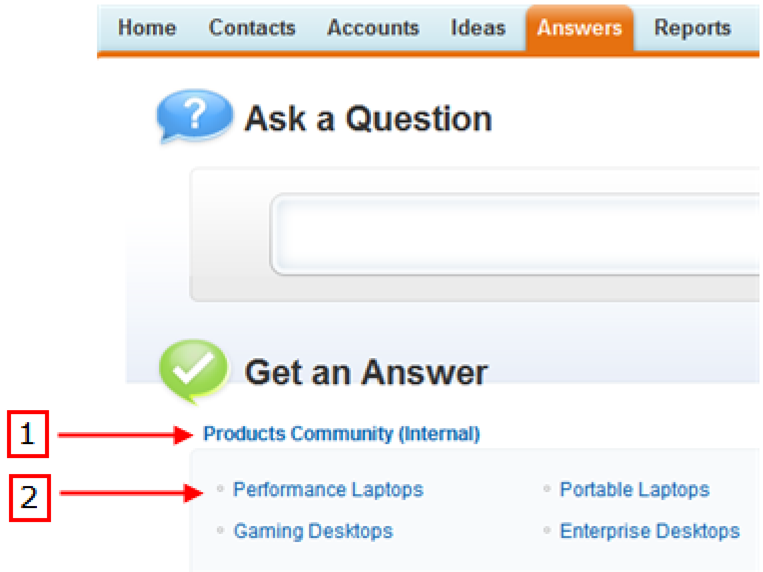


The next figure illustrates an agent finding that same article published on the Articles tab. The agent selects `Laptops` and `USA` in the `Products` and `Geography` drop-down lists respectively.



Data Categories and Answers

In an answers zone, data categories help organize questions for easy browsing. Each answers zone supports one category group. For example, if you're a computer manufacturer you might create a Products category group that has four sibling categories: Performance Laptops, Portable Laptops, Gaming Desktops, and Enterprise Desktops. On the Answers tab, zone members can assign one of the four categories to each question and then browse these categories for answers to specific questions.



Working with Data Categories in the API

The following table lists API resources for working with data categories.

Name	Type	Description
Article Type`__DataCategorySelection`	Object	Gives access to article categorization.
`QuestionDataCategorySelection`	Object	Gives access to question categorization.
`WITH DATA CATEGORY` ***`filteringExpression`***	SOQL clause	Filters articles depending on their status in the publishing cycle and their data categories. For more information, see the *Salesforce SOQL and SOSL Reference Guide*.
`WITH DATA CATEGORY` ***`DataCategorySpec`***	SOSL clause	Finds articles based on their categorization. For more information, see the *Salesforce SOQL and SOSL Reference Guide*.
`describeDataCategoryGroups()`	Call	Retrieves available category groups for objects specified in the request.
`describeDataCategoryGroupStructures()`	Call	Retrieves available category groups along with their data category structure for objects specified in the request.
`describeDataCategoryGroups`	Apex method	Returns a list of the category groups associated with the specified objects. See the *Force.com Apex Code Developer's Guide*.
`describeDataCategoryGroupStructures`	Apex method	Returns available category groups along with their data category structure for objects specified in the request. See the *Force.com Apex Code Developer's Guide*.

Glossary

A |B |C |D |E |F |G |H |I |J |K |L |M |N |O |P |Q |R |S |T |U |V |W |X |Y |Z

A

AJAX Toolkit

A JavaScript wrapper around the API that allows you to execute any API call and access any object you have permission to view from within JavaScript code. For more information, see the *AJAX Toolkit Developer's Guide*.

Anonymous Block, Apex

Apex code that does not get stored in Salesforce, but that can be compiled and executed through the use of the `ExecuteAnonymousResult()` API call, or the equivalent in the AJAX Toolkit.

Anti-Join

An anti-join is a subquery on another object in a `NOT IN` clause in a SOQL query. You can use anti-joins to create advanced queries, such as getting all accounts that do not have any open opportunities. See also Semi-Join.

Apex

Apex is a strongly typed, object-oriented programming language that allows developers to execute flow and transaction control statements on the Force.com platform server in conjunction with calls to the Force.com API. Using syntax that looks like Java and acts like database stored procedures, Apex enables developers to add business logic to most system events, including button clicks, related record updates, and Visualforce pages. Apex code can be initiated by Web service requests and from triggers on objects.

Apex-Managed Sharing

Enables developers to programmatically manipulate sharing to support their application's behavior. Apex-managed sharing is only available for custom objects.

App

Short for "application." A collection of components such as tabs, reports, dashboards, and Visualforce pages that address a specific business need. Salesforce provides standard apps such as Sales and Call Center. You can customize the standard apps to match the way you work. In addition, you can package an app and upload it to the AppExchange along with related components such as custom fields, custom tabs, and custom objects. Then, you can make the app available to other Salesforce users from the AppExchange.

AppExchange

The AppExchange is a sharing interface from salesforce.com that allows you to browse and share apps and services for the Force.com platform.

AppExchange Upgrades

Upgrading an app is the process of installing a newer version.

Application Programming Interface (API)

The interface that a computer system, library, or application provides to allow other computer programs to request services from it and exchange data.

B

Boolean Operators

You can use Boolean operators in report filters to specify the logical relationship between two values. For example, the AND operator between two values yields search results that include both values. Likewise, the OR operator between two values yields search results that include either value.

Bulk API

The REST-based Bulk API is optimized for processing large sets of data. It allows you to query, insert, update, upsert, or delete a large number of records asynchronously by submitting a number of batches which are processed in the background by Salesforce. See also SOAP API.

C

Callout, Apex

An Apex callout enables you to tightly integrate your Apex with an external service by making a call to an external Web service or sending a HTTP request from Apex code and then receiving the response.

Child Relationship

A relationship that has been defined on an sObject that references another sObject as the “one” side of a one-to-many relationship. For example, contacts, opportunities, and tasks have child relationships with accounts.

See also sObject.

Class, Apex

A template or blueprint from which Apex objects are created. Classes consist of other classes, user-defined methods, variables, exception types, and static initialization code. In most cases, Apex classes are modeled on their counterparts in Java.

Client App

An app that runs outside the Salesforce user interface and uses only the Force.com API or Bulk API. It typically runs on a desktop or mobile device. These apps treat the platform as a data source, using the development model of whatever tool and platform for which they are designed.

Component, Visualforce

Something that can be added to a Visualforce page with a set of tags, for example, `<apex:detail>`. Visualforce includes a number of standard components, or you can create your own custom components.

Component Reference, Visualforce

A description of the standard and custom Visualforce components that are available in your organization. You can access the component library from the development footer of any Visualforce page or the *Visualforce Developer's Guide*.

Controller, Visualforce

An Apex class that provides a Visualforce page with the data and business logic it needs to run. Visualforce pages can use the standard controllers that come by default with every standard or custom object, or they can use custom controllers.

Controlling Field

Any standard or custom picklist or checkbox field whose values control the available values in one or more corresponding dependent fields.

Custom App

See App.

Custom Field

A field that can be added in addition to the standard fields to customize Salesforce for your organization's needs.

Custom Help

Custom text administrators create to provide users with on-screen information specific to a standard field, custom field, or custom object.

Custom Links

Custom links are URLs defined by administrators to integrate your Salesforce data with external websites and back-office systems. Formerly known as Web links.

Custom Object

Custom records that allow you to store information unique to your organization.

Custom S-Control

Note: S-controls have been superseded by Visualforce pages. After March 2010 organizations that have never created s-controls, as well as new organizations, won't be allowed to create them. Existing s-controls will remain unaffected, and can still be edited.

Custom Web content for use in custom links. Custom s-controls can contain any type of content that you can display in a browser, for example a Java applet, an Active-X control, an Excel file, or a custom HTML Web form.

D

Database

An organized collection of information. The underlying architecture of the Force.com platform includes a database where your data is stored.

Database Table

A list of information, presented with rows and columns, about the person, thing, or concept you want to track. See also Object.

Data Loader

A Force.com platform tool used to import and export data from your Salesforce organization.

Data Manipulation Language (DML)

An Apex method or operation that inserts, updates, or deletes records from the Force.com platform database.

Date Literal

A keyword in a SOQL or SOSL query that represents a relative range of time such as `last month` or `next year`.

Decimal Places

Parameter for number, currency, and percent custom fields that indicates the total number of digits you can enter to the right of a decimal point, for example, 4.98 for an entry of 2. Note that the system rounds the decimal numbers you enter, if necessary. For example, if you enter 4.986 in a field with `Decimal Places` of 2, the number rounds to 4.99. Salesforce uses the round half-up rounding algorithm. Half-way values are always rounded up. For example, 1.45 is rounded to 1.5. –1.45 is rounded to –1.5.

Delegated Authentication

A security process where an external authority is used to authenticate Force.com platform users.

Dependent Field

Any custom picklist or multi-select picklist field that displays available values based on the value selected in its corresponding controlling field.

Developer Edition

A free, fully-functional Salesforce organization designed for developers to extend, integrate, and develop with the Force.com platform. Developer Edition accounts are available on developer.salesforce.com.

Developer Force

The Developer Force website at developer.salesforce.com provides a full range of resources for platform developers, including sample code, toolkits, an online developer community, and the ability to obtain limited Force.com platform environments.

Document Library

A place to store documents without attaching them to accounts, contacts, opportunities, or other records.

E

Email Alert

Email alerts are workflow and approval actions that are generated using an email template by a workflow rule or approval process and sent to designated recipients, either Salesforce users or others.

Email Template

A form email that communicates a standard message, such as a welcome letter to new employees or an acknowledgement that a customer service request has been received. Email templates can be personalized with merge fields, and can be written in text, HTML, or custom format.

Enterprise Edition

A Salesforce edition designed for larger, more complex businesses.

Enterprise WSDL

A strongly-typed WSDL for customers who want to build an integration with their Salesforce organization only, or for partners who are using tools like Tibco or webMethods to build integrations that require strong typecasting. The downside of the Enterprise WSDL is that it only works with the schema of a single Salesforce organization because it is bound to all of the unique objects and fields that exist in that organization's data model.

Entity Relationship Diagram (ERD)

A data modeling tool that helps you organize your data into entities (or objects, as they are called in the Force.com platform) and define the relationships between them. ERD diagrams for key Salesforce objects are published in the *SOAP API Developer's Guide*.

F

Field

A part of an object that holds a specific piece of information, such as a text or currency value.

Field-Level Security

Settings that determine whether fields are hidden, visible, read only, or editable for users. Available in Enterprise, Unlimited, Performance, and Developer Editions only.

Filter Condition/Criteria

Condition on particular fields that qualifies items to be included in a list view or report, such as “State equals California.”

Force.com

The salesforce.com platform for building applications in the cloud. Force.com combines a powerful user interface, operating system, and database to allow you to customize and deploy applications in the cloud for your entire enterprise.

Foreign Key

A field whose value is the same as the primary key of another table. You can think of a foreign key as a copy of a primary key from another table. A relationship is made between two tables by matching the values of the foreign key in one table with the values of the primary key in another.

Formula Field

A type of custom field. Formula fields automatically calculate their values based on the values of merge fields, expressions, or other values.

Function

Built-in formulas that you can customize with input parameters. For example, the DATE function creates a date field type from a given year, month, and day.

G

Gregorian Year

A calendar based on a 12-month structure used throughout much of the world.

Group Edition

A product designed for small businesses and workgroups with a limited number of users.

H

HTTP Debugger

An application that can be used to identify and inspect SOAP requests that are sent from the AJAX Toolkit. They behave as proxy servers running on your local machine and allow you to inspect and author individual requests.

I

ID

See Salesforce Record ID.

Inline S-Control

Note: S-controls have been superseded by Visualforce pages. After March 2010 organizations that have never created s-controls, as well as new organizations, won't be allowed to create them. Existing s-controls will remain unaffected, and can still be edited.

An s-control that displays within a record detail page or dashboard, rather than on its own page.

Instance

The cluster of software and hardware represented as a single logical server that hosts an organization's data and runs their applications. The Force.com platform runs on multiple instances, but data for any single organization is always consolidated on a single instance.

Integration User

A Salesforce user defined solely for client apps or integrations. Also referred to as the logged-in user in a SOAP API context.

ISO Code

The International Organization for Standardization country code, which represents each country by two letters.

J

Junction Object

A custom object with two master-detail relationships. Using a custom junction object, you can model a “many-to-many” relationship between two objects. For example, you may have a custom object called “Bug” that relates to the standard case object such that a bug could be related to multiple cases and a case could also be related to multiple bugs.

K

No Glossary items for this entry.

L

License Management Application (LMA)

A free AppExchange app that allows you to track sales leads and accounts for every user who downloads your managed package (app) from the AppExchange.

License Management Organization (LMO)

The Salesforce organization that you use to track all the Salesforce users who install your package. A license management organization must have the License Management Application (LMA) installed. It automatically receives notification every time your package is installed or uninstalled so that you can easily notify users of upgrades. You can specify any Enterprise, Unlimited, Performance, or Developer Edition organization as your license management organization. For more information, go to http://www.salesforce.com/docs/en/lma/index.htm.

List View

A list display of items (for example, accounts or contacts) based on specific criteria. Salesforce provides some predefined views.

In the Agent console, the list view is the top frame that displays a list view of records based on specific criteria. The list views you can select to display in the console are the same list views defined on the tabs of other objects. You cannot create a list view within the console.

Locale

The country or geographic region in which the user is located. The setting affects the format of date and number fields, for example, dates in the English (United States) locale display as 06/30/2000 and as 30/06/2000 in the English (United Kingdom) locale.

In Professional, Enterprise, Unlimited, Performance, and Developer Edition organizations, a user’s individual `Locale` setting overrides the organization’s `Default Locale` setting. In Personal and Group Editions, the organization-level locale field is called `Locale`, not `Default Locale`.

Logged-in User

In a SOAP API context, the username used to log into Salesforce. Client applications run with the permissions and sharing of the logged-in user. Also referred to as an integration user.

M

Managed Package

A collection of application components that is posted as a unit on the AppExchange and associated with a namespace and possibly a License Management Organization. To support upgrades, a package must be managed. An organization can create a single managed package that can be downloaded and installed by many different organizations. Managed packages differ from unmanaged packages by having some locked components, allowing the managed package to be upgraded later. Unmanaged packages do not include locked components and cannot be upgraded. In addition, managed packages obfuscate certain components (like Apex) on subscribing organizations to protect the intellectual property of the developer.

Manual Sharing

Record-level access rules that allow record owners to give read and edit permissions to other users who might not have access to the record any other way.

Many-to-Many Relationship

A relationship where each side of the relationship can have many children on the other side. Many-to-many relationships are implemented through the use of junction objects.

Master-Detail Relationship

A relationship between two different types of records that associates the records with each other. For example, accounts have a master-detail relationship with opportunities. This type of relationship affects record deletion, security, and makes the lookup relationship field required on the page layout.

Master Picklist

A complete list of picklist values available for a record type or business process.

Metadata

Information about the structure, appearance, and functionality of an organization and any of its parts. Force.com uses XML to describe metadata.

Metadata WSDL

A WSDL for users who want to use the Force.com Metadata API calls.

Multitenancy

An application model where all users and apps share a single, common infrastructure and code base.

N

Namespace

In a packaging context, a one- to 15-character alphanumeric identifier that distinguishes your package and its contents from packages of other developers onAppExchange, similar to a domain name. Salesforce automatically prepends your namespace prefix, followed by two underscores ("__"), to all unique component names in your Salesforce organization.

Native App

An app that is built exclusively with setup (metadata) configuration on Force.com. Native apps do not require any external services or infrastructure.

O

Object

An object allows you to store information in your Salesforce organization. The object is the overall definition of the type of information you are storing. For example, the case object allow you to store information regarding customer inquiries. For each object, your organization will have multiple records that store the information about specific instances of that type of data. For example, you might have a case record to store the information about Joe Smith's training inquiry and another case record to store the information about Mary Johnson's configuration issue.

Object-Level Help

Custom help text that you can provide for any custom object. It displays on custom object record home (overview), detail, and edit pages, as well as list views and related lists.

Object-Level Security

Settings that allow an administrator to hide whole objects from users so that they don't know that type of data exists. Object-level security is specified with object permissions.

onClick JavaScript

JavaScript code that executes when a button or link is clicked.

One-to-Many Relationship

A relationship in which a single object is related to many other objects. For example, an account may have one or more related contacts.

Organization-Wide Defaults

Settings that allow you to specify the baseline level of data access that a user has in your organization. For example, you can set organization-wide defaults so that any user can see any record of a particular object that is enabled via their object permissions, but they need extra permissions to edit one.

Outbound Call

Any call that originates from a user to a number outside of a call center in Salesforce CRM Call Center.

Outbound Message

An outbound message is a workflow, approval, or milestone action that sends the information you specify to an endpoint you designate, such as an external service. An outbound message sends the data in the specified fields in the form of a SOAP message to the endpoint. Outbound messaging is configured in the Salesforce setup menu. Then you must configure the external endpoint. You can create a listener for the messages using the SOAP API.

Overlay

An overlay displays additional information when you hover your mouse over certain user interface elements. Depending on the overlay, it will close when you move your mouse away, click outside of the overlay, or click a close button.

Owner

Individual user to which a record (for example, a contact or case) is assigned.

P

PaaS

See Platform as a Service.

Package

A group of Force.com components and applications that are made available to other organizations through the AppExchange. You use packages to bundle an app along with any related components so that you can upload them to AppExchange together.

Package Dependency

This is created when one component references another component, permission, or preference that is required for the component to be valid. Components can include but are not limited to:

- Standard or custom fields
- Standard or custom objects
- Visualforce pages
- Apex code

Permissions and preferences can include but are not limited to:

- Divisions
- Multicurrency
- Record types

Package Installation

Installation incorporates the contents of a package into your Salesforce organization. A package on the AppExchange can include an app, a component, or a combination of the two. After you install a package, you may need to deploy components in the package to make it generally available to the users in your organization.

Package Publication

Publishing your package makes it publicly available on the AppExchange.

Package Version

A package version is a number that identifies the set of components uploaded in a package. The version number has the format `majorNumber.minorNumber.patchNumber` (for example, 2.1.3). The major and minor numbers increase to a chosen value during every major release. The `patchNumber` is generated and updated only for a patch release.

Unmanaged packages are not upgradeable, so each package version is simply a set of components for distribution. A package version has more significance for managed packages. Packages can exhibit different behavior for different versions. Publishers can use package versions to evolve the components in their managed packages gracefully by releasing subsequent package versions without breaking existing customer integrations using the package. See also Patch and Patch Development Organization.

Parent Account

An organization or company that an account is affiliated. By specifying a parent for an account, you can get a global view of all parent/subsidiary relationships using the **View Hierarchy** link.

Partner WSDL

A loosely-typed WSDL for customers, partners, and ISVs who want to build an integration or an AppExchange app that can work across multiple Salesforce organizations. With this WSDL, the developer is responsible for marshaling data in the correct object representation, which typically involves editing the XML. However, the developer is also freed from being dependent on any particular data model or Salesforce organization. Contrast this with the Enterprise WSDL, which is strongly typed.

Patch

A patch enables a developer to change the functionality of existing components in a managed package, while ensuring subscribing organizations that there are no visible behavior changes to the package. For example, you can add new variables or change the body of an Apex class, but you may not add, deprecate, or remove any of its methods. Patches are tracked by a `patchNumber` appended to every package version. See also Patch Development Organization and Package Version.

Patch Development Organization

The organization where patch versions are developed, maintained, and uploaded. Patch development organizations are created automatically for a developer organization when they request to create a patch. See also Patch and Package Version.

Personal Edition

Product designed for individual sales representatives and single users.

Personal Information

User information including personal contact information, quotas, personal group information, and default opportunity team.

Picklist

Selection list of options available for specific fields in a Salesforce object, for example, the `Industry` field for accounts. Users can choose a single value from a list of options rather than make an entry directly in the field. See also Master Picklist.

Picklist (Multi-Select)

Selection list of options available for specific fields in a Salesforce object. Multi-select picklists allow users to choose one or more values. Users can choose a value by double clicking on it, or choose additional values from a scrolling list by holding down the CTRL key while clicking a value and using the arrow icon to move them to the selected box.

Picklist Values

Selections displayed in drop-down lists for particular fields. Some values come predefined, and other values can be changed or defined by an administrator.

Platform as a Service (PaaS)

An environment where developers use programming tools offered by a service provider to create applications and deploy them in a cloud. The application is hosted as a service and provided to customers via the Internet. The PaaS vendor provides an API for creating and extending specialized applications. The PaaS vendor also takes responsibility for the daily maintenance, operation, and support of the deployed application and each customer's data. The service alleviates the need for programmers to install, configure, and maintain the applications on their own hardware, software, and related IT resources. Services can be delivered using the PaaS environment to any market segment.

Platform Edition

A Salesforce edition based on Enterprise, Unlimited, or Performance Edition that does not include any of the standard Salesforce CRM apps, such as Sales or Service & Support.

Primary Key

A relational database concept. Each table in a relational database has a field in which the data value uniquely identifies the record. This field is called the primary key. The relationship is made between two tables by matching the values of the foreign key in one table with the values of the primary key in another.

Production Organization

A Salesforce organization that has live users accessing data.

Professional Edition

A Salesforce edition designed for businesses who need full-featured CRM functionality.

Q

Queue

A holding area for items before they are processed. Salesforce uses queues in a number of different features and technologies.

Query Locator

A parameter returned from the `query()` or `queryMore()` API call that specifies the index of the last result record that was returned.

Query String Parameter

A name-value pair that's included in a URL, typically after a '?' character. For example:

```
http://na1.salesforce.com/001/e?name=value
```

R

Record

A single instance of a Salesforce object. For example, “John Jones” might be the name of a contact record.

Record Name

A standard field on all Salesforce objects. Whenever a record name is displayed in a Force.com application, the value is represented as a link to a detail view of the record. A record name can be either free-form text or an autonumber field. `Record Name` does not have to be a unique value.

Record Type

A record type is a field available for certain records that can include some or all of the standard and custom picklist values for that record. You can associate record types with profiles to make only the included picklist values available to users with that profile.

Record-Level Security

A method of controlling data in which you can allow a particular user to view and edit an object, but then restrict the records that the user is allowed to see.

Recycle Bin

A page that lets you view and restore deleted information. Access the Recycle Bin by using the link in the sidebar.

Related Object

Objects chosen by an administrator to display in the Agent console's mini view when records of a particular type are shown in the console's detail view. For example, when a case is in the detail view, an administrator can choose to display an associated account, contact, or asset in the mini view.

Relationship

A connection between two objects, used to create related lists in page layouts and detail levels in reports. Matching values in a specified field in both objects are used to link related data; for example, if one object stores data about companies and another object stores data about people, a relationship allows you to find out which people work at the company.

Relationship Query

In a SOQL context, a query that traverses the relationships between objects to identify and return results. Parent-to-child and child-to-parent syntax differs in SOQL queries.

Report Type

A *report type* defines the set of records and fields available to a report based on the relationships between a primary object and its related objects. Reports display only records that meet the criteria defined in the report type. Salesforce provides a set of pre-defined standard report types; administrators can create custom report types as well.

Role Hierarchy

A record-level security setting that defines different levels of users such that users at higher levels can view and edit information owned by or shared with users beneath them in the role hierarchy, regardless of the organization-wide sharing model settings.

Roll-Up Summary Field

A field type that automatically provides aggregate values from child records in a master-detail relationship.

Running User

Each dashboard has a *running user*, whose security settings determine which data to display in a dashboard. If the running user is a specific user, all dashboard viewers see data based on the security settings of that user—regardless of their own personal security settings. For dynamic dashboards, you can set the running user to be the logged-in user, so that each user sees the dashboard according to his or her own access level.

S

SaaS

See Software as a Service (SaaS).

S-Control

> **Note:** S-controls have been superseded by Visualforce pages. After March 2010 organizations that have never created s-controls, as well as new organizations, won't be allowed to create them. Existing s-controls will remain unaffected, and can still be edited.

Custom Web content for use in custom links. Custom s-controls can contain any type of content that you can display in a browser, for example a Java applet, an Active-X control, an Excel file, or a custom HTML Web form.

Salesforce Record ID

A unique 15- or 18-character alphanumeric string that identifies a single record in Salesforce.

Salesforce SOA (Service-Oriented Architecture)

A powerful capability of Force.com that allows you to make calls to external Web services from within Apex.

Sandbox Organization

A nearly identical copy of a Salesforce production organization. You can create multiple sandboxes in separate environments for a variety of purposes, such as testing and training, without compromising the data and applications in your production environment.

Search Layout

The organization of fields included in search results, in lookup dialogs, and in the key lists on tab home pages.

Search Phrase

Search phrases are queries that users enter when searching on www.google.com.

Semi-Join

A semi-join is a subquery on another object in an `IN` clause in a SOQL query. You can use semi-joins to create advanced queries, such as getting all contacts for accounts that have an opportunity with a particular record type. See also Anti-Join.

Session ID

An authentication token that is returned when a user successfully logs in to Salesforce. The Session ID prevents a user from having to log in again every time he or she wants to perform another action in Salesforce. Different from a record ID or Salesforce ID, which are terms for the unique ID of a Salesforce record.

Session Timeout

The period of time after login before a user is automatically logged out. Sessions expire automatically after a predetermined length of inactivity, which can be configured in Salesforce from Setup by clicking **Security Controls**. The default is 120 minutes (two hours). The inactivity timer is reset to zero if a user takes an action in the Web interface or makes an API call.

Setup

A menu where administrators can customize and define organization settings and Force.com apps. Depending on your organization's user interface settings, Setup may be a link in the user interface header or in the drop-down list under your name.

Sharing

Allowing other users to view or edit information you own. There are different ways to share data:

- Sharing Model—defines the default organization-wide access levels that users have to each other's information and whether to use the hierarchies when determining access to data.
- Role Hierarchy—defines different levels of users such that users at higher levels can view and edit information owned by or shared with users beneath them in the role hierarchy, regardless of the organization-wide sharing model settings.
- Sharing Rules—allow an administrator to specify that all information created by users within a given group or role is automatically shared to the members of another group or role.

- Manual Sharing—allows individual users to share records with other users or groups.
- Apex-Managed Sharing—enables developers to programmatically manipulate sharing to support their application's behavior. See Apex-Managed Sharing.

Sharing Model

Behavior defined by your administrator that determines default access by users to different types of records.

Sharing Rule

Type of default sharing created by administrators. Allows users in a specified group or role to have access to all information created by users within a given group or role.

Sites

Force.com Sites enables you to create public websites and applications that are directly integrated with your Salesforce organization—without requiring users to log in with a username and password.

Snippet

Note: S-controls have been superseded by Visualforce pages. After March 2010 organizations that have never created s-controls, as well as new organizations, won't be allowed to create them. Existing s-controls will remain unaffected, and can still be edited.

A type of s-control that is designed to be included in other s-controls. Similar to a helper method that is used by other methods in a piece of code, a snippet allows you to maintain a single copy of HTML or JavaScript that you can reuse in multiple s-controls.

SOAP (Simple Object Access Protocol)

A protocol that defines a uniform way of passing XML-encoded data.

SOAP API

A SOAP-based Web services application programming interface that provides access to your Salesforce organization's information.

sObject

Any object that can be stored in the Force.com platform.

Software as a Service (SaaS)

A delivery model where a software application is hosted as a service and provided to customers via the Internet. The SaaS vendor takes responsibility for the daily maintenance, operation, and support of the application and each customer's data. The service alleviates the need for customers to install, configure, and maintain applications with their own hardware, software, and related IT resources. Services can be delivered using the SaaS model to any market segment.

SOQL (Salesforce Object Query Language)

A query language that allows you to construct simple but powerful query strings and to specify the criteria that should be used to select data from the Force.com database.

SOSL (Salesforce Object Search Language)

A query language that allows you to perform text-based searches using the Force.com API.

Standard Object

A built-in object included with the Force.com platform. You can also build custom objects to store information that is unique to your app.

Syndication Feeds

Give users the ability to subscribe to changes within Force.com sites and receive updates in external news readers.

System Log

Part of the Developer Console, a separate window console that can be used for debugging code snippets. Enter the code you want to test at the bottom of the window and click Execute. The body of the System Log displays system resource information, such as how long a line took to execute or how many database calls were made. If the code did not run to completion, the console also displays debugging information.

T

Test Method

An Apex class method that verifies whether a particular piece of code is working properly. Test methods take no arguments, commit no data to the database, and can be executed by the `runTests()` system method either through the command line or in an Apex IDE, such as the Force.com IDE.

Translation Workbench

The Translation Workbench lets you specify languages you want to translate, assign translators to languages, create translations for customizations you've made to your Salesforce organization, and override labels and translations from managed packages. Everything from custom picklist values to custom fields can be translated so your global users can use all of Salesforce in their language.

Trigger

A piece of Apex that executes before or after records of a particular type are inserted, updated, or deleted from the database. Every trigger runs with a set of context variables that provide access to the records that caused the trigger to fire, and all triggers run in bulk mode—that is, they process several records at once, rather than just one record at a time.

Trigger Context Variable

Default variables that provide access to information about the trigger and the records that caused it to fire.

U

Unit Test

A unit is the smallest testable part of an application, usually a method. A unit test operates on that piece of code to make sure it works correctly. See also Test Method.

Unlimited Edition

Unlimited Edition is salesforce.com's solution for maximizing CRM success and extending that success across the entire enterprise through the Force.com platform.

Unmanaged Package

A package that cannot be upgraded or controlled by its developer.

URL (Uniform Resource Locator)

The global address of a website, document, or other resource on the Internet. For example, http://www.salesforce.com.

URL S-Control

Note: S-controls have been superseded by Visualforce pages. After March 2010 organizations that have never created s-controls, as well as new organizations, won't be allowed to create them. Existing s-controls will remain unaffected, and can still be edited.

An s-control that contains an external URL that hosts the HTML that should be rendered on a page. When saved this way, the HTML is hosted and run by an external website. URL s-controls are also called Web controls.

V

Validation Rule

A rule that prevents a record from being saved if it does not meet the standards that are specified.

Visualforce

A simple, tag-based markup language that allows developers to easily define custom pages and components for apps built on the platform. Each tag corresponds to a coarse or fine-grained component, such as a section of a page, a related list, or a field. The components can either be controlled by the same logic that is used in standard Salesforce pages, or developers can associate their own logic with a controller written in Apex.

W

Web Control

See URL S-Control.

Web Links

See Custom Links.

Web Service

A mechanism by which two applications can easily exchange data over the Internet, even if they run on different platforms, are written in different languages, or are geographically remote from each other.

Web Services API

A Web services application programming interface that provides access to your Salesforce organization's information. See also SOAP API and Bulk API.

WebService Method

An Apex class method or variable that can be used by external systems, like a mash-up with a third-party application. Web service methods must be defined in a global class.

Web Tab

A custom tab that allows your users to use external websites from within the application.

Workflow and Approval Actions

Workflow and approval actions, such as email alerts, tasks, field updates, and outbound messages, can be triggered by a workflow rule or approval process.

Workflow Action

A workflow action, such as an email alert, field update, outbound message, or task, fires when the conditions of a workflow rule are met.

Workflow Email Alert

A workflow action that sends an email when a workflow rule is triggered. Unlike workflow tasks, which can only be assigned to application users, workflow alerts can be sent to any user or contact, as long as they have a valid email address.

Workflow Field Update

A workflow action that changes the value of a particular field on a record when a workflow rule is triggered.

Workflow Outbound Message

A workflow action that sends data to an external Web service, such as another cloud computing application. Outbound messages are used primarily with composite apps.

Workflow Queue

A list of workflow actions that are scheduled to fire based on workflow rules that have one or more time-dependent workflow actions.

Workflow Rule

A workflow rule sets workflow actions into motion when its designated conditions are met. You can configure workflow actions to execute immediately when a record meets the conditions in your workflow rule, or set time triggers that execute the workflow actions on a specific day.

Workflow Task

A workflow action that assigns a task to an application user when a workflow rule is triggered.

Wrapper Class

A class that abstracts common functions such as logging in, managing sessions, and querying and batching records. A wrapper class makes an integration more straightforward to develop and maintain, keeps program logic in one place, and affords easy reuse across components. Examples of wrapper classes in Salesforce include theAJAX Toolkit, which is a JavaScript wrapper around the Salesforce SOAP API, wrapper classes such as `CCritical Section` in the CTI Adapter for Salesforce CRM Call Center, or wrapper classes created as part of a client integration application that accesses Salesforce using the SOAP API.

WSC (Web Service Connector)

An XML-based Web service framework that consists of a Java implementation of a SOAP server. With WSC, developers can develop client applications in Java by using Java classes generated from Salesforce Enterprise WSDL or Partner WSDL.

WSDL (Web Services Description Language) File

An XML file that describes the format of messages you send and receive from a Web service. Your development environment's SOAP client uses the Salesforce Enterprise WSDL or Partner WSDL to communicate with Salesforce using the SOAP API.

X

No Glossary items for this entry.

Y

No Glossary items for this entry.

Z

No Glossary items for this entry.

DRC 190.9

Index

.NET handling of non-string fields 1640

A

AcceptedEventRelation object 128
Account object 130
AccountCleanInfo object 151
AccountContactRole object 173
AccountFeed object 174
AccountHistory object 181
AccountOwnerSharingRule object 183
AccountPartner object 186
AccountShare object 188
AccountTag object 191
AccountTeamMember object 192
AccountTerritoryAssignmentRule object 194
AccountTerritoryAssignmentRuleItem object 196
ActivityHistory 200
AdditionalNumber object 208
address 33–34, 36
AggregateResult object 1498
AllOrNoneHeader header 1618
AllowedEmailDomain object 210
AllowFieldTruncationHeader header 1619
Announcement object 210
ApexClass object 211
ApexComponent object 214
ApexLog object 217
ApexPage object 219
ApexTestQueueItem object 222
ApexTestResult object 224
ApexTrigger object 226
API
    Bulk API 1651
    choosing 1651
API usage metering 1636
AppMenuItem object 231
Approval object 235
Article
    DataCategorySelection 237
Article Type__Feed object 239
Asset object 245
AssetFeed object 248
AssetTag object 254
AssignmentRule object 256
AssignmentRuleHeader header 1620
AsyncApexJob object 257
AttachedContentDocument object 259
Attachment object 261
Authentication 70
AuthSession object 265

B

Bookmark object 267
BrandTemplate 268
BusinessProcess object 274

C

Call centers and the API 1664
CallCenter object 276
CallOptions header 1621
Calls
    convertLead() 1451
    create() 1456
    delete() 1465
    describeAllTabs() 1527
    describeAppMenu() 1528
    describeApprovalLayout() 1530
    describeAvailableQuickActions() 1531
    describeCompactLayouts() 1533
    describeDataCategoryGroups() 1536
    describeDataCategoryGroupStructures() 1538
    describeFlexiPages() 1542
    describeGlobal() 1544
    describeGlobalTheme() 1547
    describeLayout() 1549
    describePrimaryCompactLayouts() 1564
    describeQuickActions() 1565
    describeSearchLayouts() 1571
    describeSearchScopeOrder() 1570
    describeSObject() 1573
    describeSObjects() 1575–1576
    describeSoftphoneLayout() 1589
    describeTabs() 1592
    describeTheme() 1597
    emptyRecycleBin() 1468
    getDeleted() 1470
    getServerTimestamp() 1599
    getUpdated() 1474
    getUserInfo() 1601
    invalidateSessions() 1478
    list of core calls 1450
    list of describe calls 1526
    list of utility calls 1599
    login() 1480
    logout() 1484
    merge() 1485
    performQuickActions() 1490
    process() 1491
    query() 1494
    queryAll() 1500
    queryMore() 1503
    resetPassword() 1604
    retrieve() 1506

Calls *(continued)*
search() 1508
sendEmail() 1605
sendEmailMessage 1613
setPassword() 1615
undelete() 1511
update() 1514
upsert() 1520
Campaign object 278
CampaignFeed object 286
CampaignMember object 293
CampaignMemberStatus object 296
CampaignOwnerSharingRule object 298
CampaignShare object 300
CampaignTag object 301
Cascading deletes 58
Case object 303
CaseArticle object 312
CaseComment object 313
CaseContactRole object 316
CaseFeed object 317
CaseHistory object 324
CaseMilestone object 326
CaseOwnerSharingRule object 329
CaseShare object 331
CaseSolution object 333
CaseStatus object 334
CaseTag object 336
CaseTeamMember object 337
CaseTeamRole object 338
CaseTeamTemplate object 339
CaseTeamTemplateMember object 339
CaseTeamTemplateRecord object 340
CategoryData object 341
CategoryNode object 342
CategoryNodeLocalization object 344
Changes 5
Chatter
objects 105
ChatterActivity object 347
ChatterAnswersActivity object 349
ChatterAnswersReputationLevel object 354
ChatterConversation object 355
ChatterConversationMember object 355
ChatterMessage object 356
Choosing an API 1651
Chunking of HTTP 1639
Cipher key length 70
Client certificate download 1645
Client parameter in CallOptions header 1621
Client parameter in OwnerChangeOptions header 1629
ClientBrowser object 357
CollaborationGroup object 359
CollaborationGroupFeed object 364
CollaborationGroupMember object 370
CollaborationGroupMemberRequest object 372
CollaborationInvitation object 374
CombinedAttachment object 377
Community (Zone) object 379
compound fields 33–34, 36
Compression 1638
Connections, HTTP persistent 1639
Contact object 380
ContactCleanInfo object 392
ContactFeed object 402
ContactHistory object 409
ContactOwnerSharingRule object 410
ContactShare object 412
ContactTag object 414
Content type requirements 1636
ContentDocument object 415
ContentDocumentFeed object 421
ContentDocumentHistory object 427
ContentDocumentLink object 429
ContentVersion object 431
ContentVersionHistory object 440
ContentWorkspace object 442
ContentWorkspaceDoc object 444
Contract object 446
ContractContactRole object 455
ContractFeed object 456
ContractHistory object 463
ContractLineItem object 465
ContractLineItemHistory object 468
ContractStatus object 470
ContractTag object 471
convertLead() call 1451
create() call 1456
CreatedById fields 55
CreatedDate fields 55
CronJobDetail object 473
CronTrigger object 473
CSS for Salesforce look and feel 79
CurrencyType object 476
Custom Object__Feed object 481
Custom objects 59
CustomBrand object 478
CustomBrandAsset object 479
CustomPermission object 487

D

DandBCompany object 490
Dashboard object 509
DashboardComponent object 515
DashboardComponentFeed object 515
DashboardFeed object 522
DashboardTag object 528
Data types, mapping API to Salesforce field types 37
Database server instances 1636
DataCategorySelection article 237
DataCategorySelection question 1084

DatacloudCompany object 530
DatacloudContact object 536
DatacloudDandBCompany object 541
DatacloudOwnedEntity object 561
DatacloudPurchaseUsage object 563
DatacloudSocialHandle object 565
DatedConversionRate object 566
DeclinedEventRelation object 567
delete() call 1465
Deleting and cascading deletes 58
describeAllTabs() call 1527
describeAppMenu() call 1528
describeApprovalLayout()
    DescribeApprovalLayoutResult 1531
describeApprovalLayout() call 1530
DescribeApprovalLayoutResult 1531
DescribeAvailableQuickActionResult 1532
describeAvailableQuickActions() call 1531
DescribeColor 1595
describeCompactLayouts()
    DescribeCompactLayoutsResult 1535
describeCompactLayouts() call 1533
DescribeCompactLayoutsResult 1535
describeDataCategoryGroups() call 1536
describeDataCategoryGroupStructures() call 1538
describeFlexiPages() call 1542
describeGlobal() call 1544
DescribeGlobalTheme 1549
describeGlobalTheme() call 1547
DescribeIcon 1595
describeLayout() call 1549
describePrimaryCompactLayouts() call 1564
DescribeQuickActionResult 1566
describeQuickActions()
    DescribeAvailableQuickActionResult 1532
    DescribeQuickActionResult 1566
describeQuickActions() call 1565
DescribeSearchLayoutResult 1571–1572
describeSearchLayouts()
    DescribeSearchLayoutResult 1572
describeSearchLayouts() call 1571
describeSearchScopeOrder() call 1570
describeSObject() call 1573
describeSObjects() call 1575–1576
describeSoftphoneLayout() call 1589
describeTabs() call 1592
describeTheme() call 1597
DescribeThemeItem 1598
DescribeThemeResult 1598
DisableFeedTrackingHeader header 1622
Division object 569
DivisionLocalization object 570
Document object 572
DocumentAttachmentMap 577
DocumentTag object 578
Domain object 579
DomainSite object 580

E

Effective dated currency 566
EmailHeader header 1624
EmailMessage object 581
EmailServicesAddress object 585
EmailServicesFunction object 587
EmailStatus 592
EmailTemplate object 577, 594
emptyRecycleBin() call 1468
Enterprise WSDL and .NET 1640
Entitlement object 598
EntitlementContact object 602
EntitlementFeed object 603
EntitlementHistory object 610
EntitlementTemplate object 611
EntityHistory object 613
EntitySubscription object 615
EnvironmentHubMember object 617
Error handling 67
Event object 620
EventFeed object 632
EventRelation object 639
EventTag object 643
EventWhoRelation object 645
Expiration of session ID 68
External ID fields 1520
ExternalDataSource object 646
ExternalDataUserAuth object 648

F

FeedComment object 649
FeedItem object 652
FeedLike object 660
FeedPollChoice object 661
FeedPollVote object 662
FeedPost object 663
Feeds
    custom 667
    post 663, 667
    standard 667
FeedTrackedChange object 667
Field types 29, 33–34, 36
FieldHistoryArchive object 670
FieldPermissions object 671
Fields
    .NET and the enterprise WSDL handling of non-string fields 1640
    external ID fields 1520
    system fields 55
FiscalYearSettings object 674
Folder object 677
ForecastingAdjustment object 680

ForecastingFact object 683
ForecastingItem object 686
ForecastingQuota object 691
ForecastingType object 694
ForecastShare object 695

G

geolocation 33–34, 36
getDeleted() call 1470
getServerTimestamp() call 1599
getUpdated() call 1474
getUserInfo() call 1601
GoogleDoc object 697
GROUP BY 1498
Group object 698
GroupMember object 701

H

HashtagDefinition object 702
Headers
  AllOrNoneHeader 1618
  AllowFieldTruncationHeader 1619
  AssignmentRuleHeader 1620
  CallOptions 1621
  DisableFeedTrackingHeader 1622
  EmailHeader 1624
  LimitInfoHeader 1625
  LocaleOptions 1626
  LoginScopeHeader 1627
  MruHeader 1628
  OwnerChangeOptions 1629
  PackageVersionHeader 1630
  QueryOptions 1631
  SessionHeader 1632
  SOAP 1617
  UserTerritoryDeleteHeader 1632
Holiday object 703
HTTP chunking 1639
HTTP persistent connections 1639

I

ID fields 55
Idea object 707
IdeaComment object 714
IdeaReputation 716
IdeaReputationLevel object 719
IdeaTheme object 720
Implicit security restrictions for objects and field 72
Internationalization 1639
invalidateSessions() call 1478
ISO-8859-1 1639

K

KnowledgeableUser object 722
KnowledgeArticle object 723
KnowledgeArticleVersion object 725
KnowledgeArticleVersionHistory object 732
KnowledgeArticleViewStat object 735
KnowledgeArticleVoteStat object 737

L

LastModifiedById fields 55
LastModifiedDate fields 55
Lead object 738
LeadCleanInfo object 755
LeadFeed object 769
LeadHistory object 776
LeadOwnerSharingRule object 778
LeadShare object 780
LeadStatus object 781
LeadTag object 783
LimitAllocationPerApp object 784
LimitInfoHeader header 1625
Limiting API requests (API usage metering) 1636
LineItemOverride object 786
LiveAgentSession object 789
LiveAgentSessionHistory object 793
LiveAgentSessionOwnerSharingRule object 794
LiveAgentSessionShare object 795
LiveChatButton object 797
LiveChatButtonDeployment object 805
LiveChatButtonSkill object 806
LiveChatDeployment object 807
LiveChatTranscript object 809
LiveChatTranscriptEvent object 817
LiveChatTranscriptHistory object 819
LiveChatTranscriptOwnerSharingRule 820
LiveChatTranscriptOwnerSharingRule object
  LiveChatTranscriptOwnerSharingRule 820
LiveChatTranscriptShare object 821
LiveChatTranscriptSkill object 823
LiveChatUserConfig object 824
LiveChatUserConfigProfile object 829
LiveChatUserConfigUser object 830
LiveChatVisitor object 831
LocaleOptions header 1626
Localization and character sets 1639
location 34, 36
login 70
login server URL 1634
login() call
  username 1634
LoginHistory object 833
LoginIp object 832
LoginScopeHeader 1627
logout() call 1484

Lookup relationships 58

M

MailmergeTemplate object 836
Managed packages
    version settings 65
map 34, 36
Mapping API data types to Salesforce field types 37
Master-detail relationships 58
merge() call 1485
Metering API usage 1636
MetricsDataFile object 838
MilestoneType object 841
mobile 34, 36
MruHeader header 1628

N

Network object 846
NetworkActivityAudit object 852
NetworkMember object 854
NetworkMemberGroup object 860
NetworkModeration object 861
NewsFeed object 862
Note object 869
NoteAndAttachment object 871
NoteTag object 873
Notifications 1643
notifications() call in outbound messaging WSDL 1646

O

Object[PushTopic] 1067
Object[StreamingChannel] 1221
ObjectPermissions object 874
Objects
    AcceptedEventRelation 128
    Account 130
    AccountCleanInfo 151
    AccountContractRole 173
    AccountFeed 174
    AccountHistory 181
    AccountOwnerSharingRule 183
    AccountPartner 186
    AccountShare 188
    AccountTag 191
    AccountTeamMember 192
    AccountTerritoryAssignmentRule 194
    AccountTerritoryAssignmentRuleItem 196
    ActivityHistory 200
    AdditionalNumber 208
    AggregateResult 1498
    AllowedEmailDomain 210
    Announcement 210
    ApexClass 211

Objects *(continued)*
    ApexComponent 214
    ApexLog 217
    ApexPage 219
    ApexTestQueueItem 222
    ApexTestResult 224
    ApexTrigger 226
    AppMenuItem 231
    Approval 235
    Article Type__Feed 239
    Asset 245
    AssetFeed 248
    AssetTag 254
    AssignmentRule 256
    AsyncApexJob 257
    AttachedContentDocument 259
    Attachment 261
    AuthSession 265
    Bookmark 267
    BrandTemplate 268
    BusinessProcess 274
    CallCenter 276
    Campaign 278
    CampaignFeed 286
    CampaignMember 293
    CampaignMemberStatus 296
    CampaignOwnerSharingRule 298
    CampaignShare 300
    CampaignTag 301
    Case 303
    CaseArticle 312
    CaseComment 313
    CaseContactRole 316
    CaseFeed 317
    CaseHistory 324
    CaseMilestone 326
    CaseOwnerSharingRule 329
    CaseShare 331
    CaseSolution 333
    CaseStatus 334
    CaseTag 336
    CaseTeamMember 337
    CaseTeamRole 338
    CaseTeamTemplate 339
    CaseTeamTemplateMember 339
    CaseTeamTemplateRecord 340
    CategoryData 341
    CategoryNode 342
    CategoryNodeLocalization 344
    ChatterActivity 347
    ChatterAnswersActivity 349
    ChatterAnswersReputationLevel 354
    ChatterConversation 355
    ChatterConversationMember 355
    ChatterMessage 356
    ClientBrowser 357

Objects *(continued)*
CollaborationGroup 359
CollaborationGroupFeed 364
CollaborationGroupMember 370
CollaborationGroupMemberRequest 372
CollaborationInvitation 374
CombinedAttachment 377
Community (Zone) 379
Contact 380
ContactCleanInfo 392
ContactFeed 402
ContactHistory 409
ContactOwnerSharingRule 410
ContactShare 412
ContactTag 414
ContentDocument 415
ContentDocumentFeed 421
ContentDocumentHistory 427
ContentDocumentLink 429
ContentVersion 431
ContentVersionHistory 440
ContentWorkspace 442
ContentWorkspaceDoc 444
Contract 446
ContractContactRole 455
ContractFeed 456
ContractHistory 463
ContractLineItem 465
ContractLineItemHistory 468
ContractStatus 470
ContractTag 471
CronJobDetail 473
CronTrigger 473
CurrencyType 476
Custom Object__Feed 481
custom objects 59
CustomBrand 478
CustomBrandAsset 479
CustomPermission 487
DandBCompany 490
Dashboard 509
DashboardComponent 515
DashboardComponentFeed 515
DashboardFeed 522
DashboardTag 528
DatacloudCompany 530
DatacloudContact 536
DatacloudDandBCompany 541
DatacloudOwnedEntity 561
DatacloudPurchaseUsage 563
DatacloudSocialHandle 565
DatedConversionRate 566
DeclinedEventRelation 567
Division 569
DivisionLocalization 570
Document 572

Objects *(continued)*
DocumentAttachmentMap 577
DocumentTag 578
Domain 579
DomainSite 580
EmailMessage 581
EmailServicesAddress 585
EmailServicesFunction 587
EmailStatus 592
EmailTemplate 594
Entitlement 598
EntitlementContact 602
EntitlementFeed 603
EntitlementHistory 610
EntitlementTemplate 611
EntityHistory 613
EntitySubscription 615
EnvironmentHubMember 617
Event 620
EventFeed 632
EventRelation 639
EventTag 643
EventWhoRelation 645
ExternalDataSource 646
ExternalDataUserAuth 648
FeedComment 649
FeedItem 652
FeedLike 660
FeedPollChoice 661
FeedPollVote 662
FeedPost 663
FeedTrackedChange 667
FieldHistoryArchive 670
FieldPermissions 671
FiscalYearSettings 674
Folder 677
ForecastingAdjustment 680
ForecastingFact 683
ForecastingItem 686
ForecastingQuota 691
ForecastingType 694
ForecastShare 695
GoogleDoc 697
Group 698
GroupMember 701
HashtagDefinition 702
Holiday 703
Idea 707
idea comment 714
IdeaReputation 716
IdeaReputationLevel 719
IdeaTheme 720
KnowledgeableUser 722
KnowledgeArticle 723
KnowledgeArticleVersion 725
KnowledgeArticleVersionHistory 732

Objects *(continued)*
KnowledgeArticleViewStat 735
KnowledgeArticleVoteStat 737
Lead 738
LeadCleanInfo 755
LeadFeed 769
LeadHistory 776
LeadOwnerSharingRule 778
LeadShare 780
LeadStatus 781
LeadTag 783
LimitAllocationPerApp 784
LineItemOverride 786
LiveAgentSession 789
LiveAgentSessionHistory 793
LiveAgentSessionOwnerSharingRule 794
LiveAgentSessionShare 795
LiveChatButton 797
LiveChatButtonDeployment 805
LiveChatButtonSkill 806
LiveChatDeployment 807
LiveChatTranscript 809
LiveChatTranscriptEvent 817
LiveChatTranscriptHistory 819
LiveChatTranscriptShare 821
LiveChatTranscriptSkill 823
LiveChatUserConfig 824
LiveChatUserConfigProfile 829
LiveChatUserConfigUser 830
LiveChatVisitor 831
LoginHistory 833
LoginIp 832
MailmergeTemplate 836
MetricsDataFile 838
MilestoneType 841
Network 846
NetworkActivityAudit 852
NetworkMember 854
NetworkMemberGroup 860
NetworkModeration 861
NewsFeed 862
Note 869
NoteAndAttachment 871
NoteTag 873
ObjectPermissions 874
ObjectTerritory2AssignmentRule 877
ObjectTerritory2AssignmentRuleItem 879
ObjectTerritory2Association 881
OpenActivity 882
Opportunity 889
OpportunityCompetitor 899
OpportunityContactRole 901
OpportunityFeed 902
OpportunityFieldHistory 909
OpportunityHistory 910
OpportunityLineItem 913

Objects *(continued)*
OpportunityLineItemSchedule 919
OpportunityOverride 923
OpportunityOwnerSharingRule 926
OpportunityPartner 928
OpportunityShare 929
OpportunitySplit 931
OpportunitySplitType 933
OpportunityStage 935
OpportunityTag 938
OpportunityTeamMember 939
Order 941
OrderFeed 952
OrderHistory 957
OrderItem 959
OrderItemFeed 962
OrderItemHistory 967
Organization 969
OrgWideEmailAddress 988
OwnedContentDocument 989
PackageLicense 991
Partner 993
PartnerNetworkConnection 995
PartnerNetworkRecordConnection 997
PartnerRole 1001
Period 1002
PermissionSet 1004
PermissionSetAssignment 1009
PermissionSetLicense 1011
PermissionSetLicenseAssign 1014
Pricebook2 1015
Pricebook2History 1018
PricebookEntry 1020
ProcessDefinition 1022
ProcessInstance 1024
ProcessInstanceHistory 1027
ProcessInstanceNode 1032
ProcessInstanceStep 1030
ProcessInstanceWorkitem 1035
ProcessNode 1037
Product2 1038
Product2Feed 1043
ProductEntitlementTemplate 1050
Profile 1051
ProfileSkill 1054
ProfileSkillEndorsement 1056
ProfileSkillEndorsementHistory 1057
ProfileSkillFeed 1058
ProfileSkillHistory 1062
ProfileSkillShare 1063
ProfileSkillUser 1065
ProfileSkillUserHistory 1066
QuantityForecast 1069
QuantityForecastHistory 1077
Question 1079
QuestionReportAbuse 1086

Objects *(continued)*
QuestionSubscription 1087
QueueSobject 1089
QuickText 1089
QuickTextHistory 1091
QuickTextOwnerSharingRule 1092
QuickTextShare 1094
Quote 1095
QuoteDocument 1109
QuoteLineItem 1111
RecentlyViewed 1115
RecordType 1119
RecordTypeLocalization 1122
Reply 1124
ReplyReportAbuse 1126
Report 1127
ReportFeed 1130
ReportTag 1137
RevenueForecast 1138
RevenueForecastHistory 1147
RuleTerritory2Association 1150
Scontrol 1151
ScontrolLocalization 1154
SelfServiceUser 1158
ServiceContract 1161
ServiceContractFeed 1170
ServiceContractHistory 1177
ServiceContractOwnerSharingRule 1178
ServiceContractShare 1180
SetupEntityAccess 1182
SignupRequest 1184
Site 1190
SiteDomain 1195
SiteHistory 1196
Skill 1198
SkillProfile 1199
SkillUser 1200
SlaProcess 1200
Solution 1203
SolutionFeed 1207
SolutionHistory 1214
SolutionStatus 1215
SolutionTag 1217
standard objects 109
StaticResource 1218
TagDefinition 1222
Task 1224
TaskFeed 1233
TaskPriority 1240
TaskRelation 1241
TaskStatus 1243
TaskTag 1245
TaskWhoRelation 1246
Territory 1247
Territory2 1250
Territory2Model 1253

Objects *(continued)*
Territory2ModelHistory 1255
Territory2Type 1256
Topic 1258
TopicAssignment 1259
TopicFeed 1260
UndecidedEventRelation 1266
User 1268
UserAccountTeamMember 1306
UserConfigTransferButton 1309
UserConfigTransferSkill 1309
UserFeed 1310
UserLicense 1317
UserLogin 1320
UserMembershipSharingRule 1321
UserPackageLicense 1322
UserPreference 1323
UserProfile 1324
UserProfileFeed 1343
UserRecordAccess 1350
UserRole 1352
UserShare 1356
UserTeamMember 1358
UserTerritory 1359
UserTerritory2Association 1361
vote 1362
WebLink 1363
WebLinkLocalization 1369
WorkAccess 1373
WorkAccessShare 1374
WorkBadge 1375
WorkBadgeDefinition 1376
WorkBadgeDefinitionHistory 1379
WorkBadgeDefinitionShare 1380
WorkCoaching 1381
WorkCoachingFeed 1383
WorkCoachingHistory 1388
WorkCoachingShare 1389
WorkFeedback 1391
WorkFeedbackQuestion 1392
WorkFeedbackQuestionHistory 1395
WorkFeedbackQuestionSet 1396
WorkFeedbackQuestionSetShare 1398
WorkFeedbackQuestionShare 1399
WorkFeedbackRequest 1400
WorkFeedbackRequestHistory 1404
WorkFeedbackRequestShare 1405
WorkFeedbackShare 1406
WorkGoal 1408
WorkGoalCollaborator 1414
WorkGoalCollaboratorHistory 1415
WorkGoalFeed 1416
WorkGoalHistory 1422
WorkGoalLink 1423
WorkGoalShare 1424
WorkPerformanceCycle 1425

Objects *(continued)*
WorkPerformanceCycleFeed 1427
WorkPerformanceCycleHistory 1432
WorkPerformanceCycleShare 1433
WorkReward 1435
WorkRewardFund 1436
WorkRewardFundHistory 1438
WorkRewardFundShare 1439
WorkRewardFundType 1440
WorkRewardFundTypeHistory 1443
WorkRewardFundTypeShare 1444
WorkRewardHistory 1445
WorkRewardShare 1446
WorkThanks 1447
WorkThanksShare 1448
ObjectTerritory2AssignmentRule object 877
ObjectTerritory2AssignmentRuleItem object 879
ObjectTerritory2Association object 881
OpenActivity object 882
Opportunity object 889
OpportunityCompetitor object 899
OpportunityContactRole object 901
OpportunityFeed object 902
OpportunityFieldHistory object 909
OpportunityHistory object 910
OpportunityLineItem object 913
OpportunityLineItemSchedule object 919
OpportunityOverride object 923
OpportunityOwnerSharingRule object 926
OpportunityPartner object 928
OpportunityShare object 929
OpportunitySplit object 931
OpportunitySplitType object 933
OpportunityStage object 935
OpportunityTag object 938
OpportunityTeamMember object 939
Order object 941
OrderFeed object 952
OrderHistory object 957
OrderItem object 959
OrderItemFeed object 962
OrderItemHistory object 967
Organization object 969
OrgWideEmailAddress object 988
Outbound messaging
and notifications 1643
building a listener 1648
configuring up user profiles 1644
defining 1644
introduction 1641
notification calls in WSDL 1646
port restrictions 1644
security 1646
setting up 1643
tracking 1646
understanding 1642
Outbound messaging *(continued)*
viewing 1645
OwnedContentDocument object 989
OwnerChangeOptions header 1629

P

Package API access 72
PackageLicense object 991
PackageVersionHeader headers 1630
Partner object 993
PartnerNetworkConnection object 995
PartnerNetworkRecordConnection object 997
PartnerRole object 1001
PerformQuickActionResult 1491
performQuickActions()
PerformQuickActionResult 1491
performQuickActions() call 1490
Period object 1002
Permission sets configuration 70
PermissionSet object 1004
PermissionSetAssignment object 1009
PermissionSetLicense object 1011
PermissionSetLicenseAssign object 1014
Persistent connections 1639
Picklists
multi-select picklists 1579
picklistValues 1579
restrictedPicklist 1579
single-select picklists 1579
Port restrictions for outbound ports 74
Posts 663, 667
Pricebook2 object 1015
Pricebook2History object 1018
PricebookEntry object 1020
process() call 1491
ProcessDefinition object 1022
ProcessInstance object 1024
ProcessInstanceHistory object 1027
ProcessInstanceNode object 1032
ProcessInstanceStep object 1030
ProcessInstanceWorkitem 1035
ProcessNode object 1037
Product2 object 1038
Product2Feed object 1043
ProductEntitlementTemplate object 1050
Profile object 1051
ProfileSkill object 1054
ProfileSkillEndorsement object 1056
ProfileSkillEndorsementHistory object 1057
ProfileSkillFeed object 1058
ProfileSkillHistory object 1062
ProfileSkillShare object 1063
ProfileSkillUser object 1065
ProfileSkillUserHistory object 1066
PushTopic object 1067

Q

QuantityForecast object 1069
QuantityForecastHistory object 1077
query() call 1494
queryAll() call 1500
queryMore() call 1503
QueryOptions headers 1631
Question object 1079
QuestionDataCategorySelection 1084
QuestionReportAbuse object 1086
QuestionSubscription object 1087
QueueSobject object 1089
Quickstart
    Generate the WSDL 6
    Import the WSDL 7
    Obtain a Developer Edition Organization 6
    Walk through code 8
QuickText object 1089
QuickTextHistory object 1091
QuickTextOwnerSharingRule object 1092
QuickTextShare object 1094
Quote object 1095
QuoteDocument object 1109
QuoteLineItem object 1111

R

Recent items list and update call 1628
RecentlyViewedobject 1115
RecordType object 1119
RecordTypeLocalization object 1122
Recycle Bin 1511
Relationships
    parent reference fields in child object 56
Reply object 1124
ReplyReportAbuse object 1126
Report object 1127
ReportFeed object 1130
ReportTag object 1137
Requirements for content type 1636
resetPassword() call 1604
retrieve() call 1506
RevenueForecast object 1138
RevenueForecastHistory object 1147
Role, see UserRole 1352
RuleTerritory2Association object 1150

S

Salesforce Sandbox 1635
salesforce.com instances 1636
Sandbox 1635
Scontrol object 1151
ScontrolLocalization object 1154
search() call 1508
Security
    and packages 72
    implicit restrictions for objects and fields 72
    outbound messaging 1646
    outbound port restrictions 74
    sharing 71
    user profile and permission sets configuration 70
Security token 70
SelfServiceUser object 1158
sendEmail() call 1605
sendEmailMessage call 1613
ServiceContract object 1161
ServiceContractFeed object 1170
ServiceContractHistory object 1177
ServiceContractOwnerSharingRule object 1178
ServiceContractShare object 1180
Session ID expiration 68
SessionHeader header 1632
setPassword() call 1615
SetupEntityAccess object 1182
Sharing 71
SignupRequest object 1184
Site object 1190
SiteDomain object 1195
SiteHistory object 1196
Skill object 1198
SkillProfile object 1199
SkillUser object 1200
SlaProcess object 1200
SOAP headers 1617
Solution object 1203
SolutionFeed object 1207
SolutionHistory object 1214
SolutionStatus object 1215
SolutionTag object 1217
SSL requirement 70
Standard objects 109
StaticResource object 1218
StreamingChannel object 1221
Stylesheets for Salesforce look and feel 79
Support policy
    API 4
    Backward compatibility 4
    Deprecated API versions 4
    End-of-life notifications 4
System fields 55
SystemModstamp fields 55

T

TagDefinition object 1222
Task object 1224
TaskFeed object 1233
TaskPriority object 1240
TaskRelation object 1241
TaskStatus object 1243

TaskTag object 1245
TaskWhoRelation object 1246
Territory object 1247
Territory2 object 1250
Territory2Model object 1253
Territory2ModelHistory object 1255
Territory2Type object 1256
TLS requirement 70
Topic object 1258
TopicAssignment object 1259
TopicFeed object 1260
Types of fields 29, 33–34, 36

U

UndecidedEventRelation object 1266
undelete() 1511
Unicode 1639
update() call
    and recent items list 1628
updateMru parameter in MruHeader header 1628
upsert() call 1520
User authentication 70
User interface themes 79
User object 1268
User profile configuration 70
User profiles
    configuring up for outbound messaging 1644
UserAccountTeamMember object 1306
UserConfigTransferButton object 1309
UserConfigTransferSkill object 1309
UserFeed object 1310
UserLicense object 1317
UserLogin object 1320
UserMembershipSharingRule object 1321
UserPackageLicense object 1322
UserPreference object 1323
UserProfile object 1324
UserProfileFeed object 1343
UserRecordAccess object 1350
UserRole object 1352
UserShare object 1356
UserTeamMember object 1358
UserTerritory object 1359
UserTerritory2Association object 1361
UserTerritoryDeleteHeader header 1632
UTF-8 1639

V

Version settings
    managed packages 65
Vote object 1362

W

WebLink object 1363
WebLinkLocalization object 1369
What's New 5
WorkAccess object 1373
WorkAccessShare object 1374
WorkBadge object 1375
WorkBadgeDefinition object 1376
WorkBadgeDefinitionHistory object 1379
WorkBadgeDefinitionShare object 1380
WorkCoaching object 1381
WorkCoachingFeed object 1383
WorkCoachingHistory object 1388
WorkCoachingShare object 1389
WorkFeedback object 1391
WorkFeedbackQuestion object 1392
WorkFeedbackQuestionHistory object 1395
WorkFeedbackQuestionSet object 1396
WorkFeedbackQuestionSetShare object 1398
WorkFeedbackQuestionShare object 1399
WorkFeedbackRequest object 1400
WorkFeedbackRequestHistory object 1404
WorkFeedbackRequestShare object 1405
WorkFeedbackShare object 1406
WorkGoal object 1408
WorkGoalCollaborator object 1414
WorkGoalCollaboratorHistory object 1415
WorkGoalFeed object 1416
WorkGoalHistory object 1422
WorkGoalLink object 1423
WorkGoalShare object 1424
WorkPerformanceCycle object 1425
WorkPerformanceCycleFeed object 1427
WorkPerformanceCycleHistory object 1432
WorkPerformanceCycleShare object 1433
WorkReward object 1435
WorkRewardFund object 1436
WorkRewardFundHistory object 1438
WorkRewardFundShare object 1439
WorkRewardFundType object 1440
WorkRewardFundTypeHistory object 1443
WorkRewardFundTypeShare object 1444
WorkRewardHistory object 1445
WorkRewardShare object 1446
WorkThanks object 1447
WorkThanksShare object 1448
WSDL
    version settings 65

X

XML compliance 1639