Version 25.0: Summer '12

Object Reference for Database.com

Last updated: April 16 2012

Table of Contents

Chapter 1

Standard and Custom Object Basics

In this chapter ...

Generally speaking, API objects represent database tables that contain your organization's information. The term “record” describes a particular occurrence of an object. A record is analogous to a row in a database table.

Objects already created for you by Database.com 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.

Primitive Data Types

The API uses the following primitive data types:

Value	Details
base64	Base 64-encoded binary data.
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. 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 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.
double	Double values. Fields of this type can contain fractional portions (digits to the right of the decimal place). 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. **Caution:** When the user sets the precision in custom fields in the Database.com 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). For integer fields, the digits field specifies the maximum number of digits that an integer can have.

Value	Details
string	Character strings. Fields that are of data type `string` contain text and some have length restrictions depending on the data being stored. **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. 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 Database.com-specific way, which is useful for display formatting and for numeric conversion (adding values of different currencies).

For example, Database.com 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, Database.com 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, Database.com 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
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.
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.
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 Database.com 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 Database.com 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 Database.com.

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 account. 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 Database.com 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 *Database.com 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 Database.com 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 Database.com 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 Database.com 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.

API Data Types and Database.com 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, an object might have 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.

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.

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.

Field	Field Type	Description
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 a workflow process (such as a trigger). SystemModstamp fields have Defaulted on create and Filter access.

If you import data into Database.com and need to set the value for an audit field, contact salesforce.com. 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.

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, Line_Item__c could have a Invoice_Statement__c field that contains a reference to the Invoice_Statement__c parent of the Line_Item__c.
- *Parent_NameId* contains the ID of the parent.

Even if the object can parent itself, these fields occur.

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.
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.

Property	Description
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.

Custom Fields

Database.com administrators can define custom fields for standard or custom objects in their organization using the user interface. During creation, the `custom` flag—a Boolean field in the Field object—is set to true. Client applications cannot define custom fields via the API. Usually, client applications do not need to know whether a field is a standard field or a custom field.

Note that all numeric custom fields are handled as type double.

Use the following topics to understand custom fields:

- Objects That Support Custom Fields
- Naming Conventions for Custom Fields
- External ID Attribute on Custom Fields
- Uniqueness for Custom Fields
- Default Values in Custom Fields

Objects That Support Custom Fields

To identify the standard objects that support custom fields, see the table in Relationships Among Custom Objects.

Naming Conventions for Custom Fields

Custom objects have an associated name field that is defined by your Database.com administrator. Custom fields must have unique names within the same object.

In the API, the names of custom fields are identified by a suffix of two underscores immediately followed by a lowercase “c” character. For example, a custom object labeled “Issue” in the user interface is seen as `Issue__c` in that organization's WSDL. Similarly, a custom field labeled “Hire Date” in the user interface is seen as `Hire_Date__c` in that organization's WSDL. For example, a custom field labeled “Hire Date” in the user interface is seen as `Hire_Date__c` in that organization's WSDL.

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

External ID Attribute on Custom Fields

In the user interface, you can identify one custom field on an object as being an external ID field. The field type must be a text, number, or email field. An external ID contains record IDs from a system outside of Database.com. You can match against this field during import or integration, or when upserting records.

Uniqueness for Custom Fields

In the user interface, you can specify that a custom field on a custom object contain unique values across all the records of that custom object type. The uniqueness can be either case sensitive or case insensitive. In the API, you can find out if a field is unique by issuing a describe call against the custom object and inspecting two attribute values:

- If the `unique` field is set to `true`, the custom field values must be unique across all records of that custom object type in the organization. A value of `false` means the field can have the same value in different records of that custom object type.
- If the `caseSensitive` field is set to `true`, the uniqueness (if enabled) is case sensitive. For example "ABC" and "abc" are considered two unique values. If the value is `false`, then "ABC" and "abc" are considered the same value.

These values cannot be set or modified using API calls. If a custom field on a custom object has `unique` set to `true`, and you try to insert a duplicate value, a `DUPLICATE_VALUE` exception code is returned.

Default Values in Custom Fields

You can set a default value on a custom field using a formula field:

- The user logged in for API activity must be have the "Customize Application" permission.
- The field must have a data type of currency, date, datetime, int, double, percent, string, textarea, email, phone, or url. You cannot use composite fields like Address, Person, Names, nor Fiscal Periods. Note that you can set a checkbox as checked or unchecked by default using the user interface, but you cannot set it using a formula field.
- Default formulas run on fields, and the results are saved, even if the fields are hidden by field-level security.
- Default values are not used for lead conversion, importing, or merging records.

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

Naming Conventions for Custom Objects

Custom objects have an associated name field that is defined by your Database.com 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 Database.com 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 Database.com 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 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 *Database.com SOQL and SOSL Reference Guide*.

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. 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.

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.

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. If a required field does not have a specified or default value, an error with the status code `REQUIRED_FIELD_MISSING` is returned.

Relationships Among Objects

Relationships associate objects with other objects. For example, a relationship can link a custom object called Line_Item__c to another custom object called Invoice_Statement__c.

Note: You can use parent-child relationships in SOQL queries. For more information, see Relationship Queries in the *Database.com 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 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.

  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 bug object that relates to a support case object such that a bug could be related to multiple support cases and a support 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 Database.com 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 Database.com online help for details.

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

REFERENCE

Chapter 2

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.

Fields

Field	Details
`City`	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** Name of the city for the organization's address.
`ComplianceBccEmail`	**Type** email **Properties** Filter, Group, Nillable, Sort **Description** Email address for compliance blind carbon copies. Limit: 80 characters.
`Country`	**Type** string

Field	Details
	Properties Filter, Group, Nillable, Sort **Description** Name of the country for the organization's address.
`DefaultCalendarAccess`	**Type** picklist **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)
`DefaultLocaleSidKey`	**Type** picklist **Properties** Filter, Group, Restricted picklist, Sort, Update **Description** Default locale SID key.
`Division`	**Type** string **Properties** Filter, Nillable, Update **Description** The name of the division for this organization.
`Fax`	**Type** phone **Properties** Filter, Group, Nillable, Sort, Update **Description** Fax number. Limit: 40 characters.

Field	Details
`FiscalYearStartMonth`	**Type** int **Properties** Filter, Group, Nillable, Sort **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.
`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.
`Name`	**Type** string **Properties** Filter, Group, Sort, Update **Description** The name of the organization.
`OrganizationType`	**Type** picklist **Properties** Filter, Group, Nillable, Restricted picklist, Sort **Description** Edition of the organization.

Field	Details
`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.
`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 Database.com (`true`) or not (`false`).

Field	Details
`State`	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** State of the address of the organization. Limit: 20 characters.
`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.

Usage

Query this object to obtain information about an organization's settings. Only one organization object exists per organization.

Chapter 3

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()`, `update()`

Fields

Field	Details
`Description`	**Type** string **Properties** Filter, Group, Nillable, Sort, Update **Description** Description of the profile.
`Name`	**Type** string **Properties** Filter, Group, Sort, Update **Description** The name of the profile.
`Permission`*`PermissionName`*	**Type** boolean **Properties** Filter, Update **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.

Field	Details
`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 Database.com Platform and Database.com Platform One user licenses. Label is **Standard**. • PowerPartner: 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**. • CSPLitePortal: user whose access is limited because he or she is an organization's customer and accesses the application through aCustomer Portal. 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. `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**. • Database.com: Database.com user license. Label is **Salesforce**. • 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**.

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.

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 Database.com 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.

Chapter 4

User

Represents a user in your organization.

Supported Calls

`create()`, `describeSObjects()`, `getDeleted()`, `getUpdated()`, `query()`, `retrieve()`, `search()`, `update()`, `upsert()`

Special Access Rules

- To create or update a User record, you must log in with the “Manage Users” permission.

Fields

Field	Details
AboutMe	**Type** textarea **Properties** Create, Filter, Nillable, Sort, Update **Description** Information about the user, such as areas of interest or skills.
Alias	**Type** string **Properties** Create, Filter, Group, Sort, Update **Description** Required. The user’s alias. For example, `jsmith`.
City	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The city associated with the user.

Field	Details
`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.
`Country`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The country associated with the user.
`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. However, the User object could have currency custom fields stored in a different currency. 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`. . 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 **Properties** Create, Defaulted on create, Filter, Group, Nillable, Restricted picklist, Sort, Update

Field	Details
	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, 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`.
`EmployeeNumber`	**Type** string **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The user's employee number.
`Extension`	**Type** phone

Field	Details
	Properties Create, Filter, Group, Nillable, Sort, Update **Description** The user's phone extension number.
`Fax`	**Type** phone **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The user's fax number.
`FederationIdentifier`	**Type** string **Properties** Create, Filter, 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.
`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.
`LanguageLocaleKey`	**Type** picklist **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.
`LocaleSidKey`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update

Field	Details
	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.
`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 **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The Id of the user who manages this user.
`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.

Field	Details
`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 **Description** The user's phone number.
`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 **Description** Indicates whether the user receives email for administrators from salesforce.com (`true`) or not (`false`).
`ReceivesInfoEmails`	**Type** boolean

Field	Details
	Properties Create, Defaulted on create, Filter, Group, Sort, Update **Description** Indicates whether the user receives informational email from salesforce.com (`true`) or not (`false`).
`SmallPhotoUrl`	**Type** string **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.
`Street`	**Type** textarea **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The street address associated with the User.
`TimeZoneSidKey`	**Type** picklist **Properties** Create, Filter, Group, Restricted picklist, Sort, Update

Field	Details
	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 **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** The user's business title, such as "Vice President."
`Username`	**Type** string **Properties** Create, Filter, Group, 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.
`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`).
`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**.
`UserPermissionsMobileUser`	**Type** boolean **Properties** Create, Filter, Update **Description** Indicates whether the user is allocated one Salesforce Mobile license (`true`) or not (`false`). Label is **Mobile User**. The Salesforce Mobile license grants the user access to the Salesforce Mobile application on supported mobile devices.
`UserPreferencesDisableAllFeedsEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email for all updates to Chatter feeds. 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 they create.
`UserPreferencesDisableBookmarkEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email for all comments made to a Chatter feed item after a 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 for all comments made on a user's change, such as an update to their profile. This field is available in API version 24.0 and later.
`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 for all updates to all the Chatter users currently being followed.This field is available in API version 24.0 and later.
`UserPreferencesDisableLaterCommentEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email for all comments made 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

Field	Details
	Properties Create, Filter, Update **Description** When `false`, the user automatically receives email for all comments made on a feed item after the user has liked the feed item. 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 for all mentions made of the user 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 for all posts made to the user's profile. This field is available in API version 24.0 and later.
`UserPreferencesDisableSharePostEmail`	**Type** boolean **Properties** Create, Filter, Update **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.
`UserPreferencesDisCommentAfterLikeEmail`	**Type** boolean **Properties** Create, Filter, Update

Field	Details
	Description When `false`, the user automatically receives email every time someone makes a comment to 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 for all mentions made of the user 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.
`UserPreferencesDisProfPostCommentEmail`	**Type** boolean **Properties** Create, Filter, Update **Description** When `false`, the user automatically receives email for comments made on user profile posts. This field is available in API version 24.0 and later.
`UserRoleId`	**Type** reference **Properties** Create, Filter, Group, Nillable, Sort, Update **Description** ID of the user's UserRole. Label is **Role ID**.
`UserType`	**Type** picklist

Field	Details
	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 Database.com Platform and Database.com Platform One user licenses. Label is **Standard**. • PowerPartner: 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**. • CSPLitePortal: user whose access is limited because he or she is an organization's customer and accesses the application through aCustomer Portal. 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.

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.

Deactivate Users

You can’t delete a user in the user interface or the API. You can deactivate a user in the user interface. 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 CreatedBy 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.

Index