

Oracle APEX - The Tailor Fit

Rajan Chandru
Sathya Jegan N
Pandimuneeswaran K
Karthik S

DOYENSYS
Technology Drives, We Lead



INDIA • SINGAPORE • MALAYSIA



Notion Press

Old No. 38, New No. 6
McNichols Road, Chetpet
Chennai - 600 031

First Published by Notion Press 2019

Copyright © Doyensys 2019

All Rights Reserved.

ISBN 978-1-64760-813-2

This book has been published with all reasonable efforts taken to make the material error-free after the consent of the author. No part of this book shall be used, reproduced in any manner whatsoever without written permission from the author, except in the case of brief quotations embodied in critical articles and reviews.

While every effort has been made to avoid any mistake or omission, this publication is being sold on the condition and understanding that neither the author nor the publishers or printers would be liable in any manner to any person by reason of any mistake or omission in this publication or for any action taken or omitted to be taken or advice rendered or accepted on the basis of this work. For any defect in printing or binding the publishers will be liable only to replace the defective copy by another copy of this work then available.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION	9
CHAPTER 2: ARCHITECTURE.....	11
ORDS	11
CHAPTER 3: SKILLS REQUIRED	13
Mandatory Skills.....	13
Optional Skills	13
CHAPTER 4: PROS AND CONS OF ORACLE APEX.....	15
Advantages	15
Disadvantages.....	16
CHAPTER 5: ROAD MAP OF APEX.....	17
CHAPTER 6: QUICK TOUR TO APEX.....	19
Application Builder	19
SQL Workshop.....	22
Object Browser	22
SQL Commands	23
SQL Scripts.....	23
Utilities.....	24
ORDS RESTful Services.....	25
Team Development	26
App Gallery	27

CONTENTS

CHAPTER 7: APEX PAGE COMPONENTS	29
Regions	29
Creating Breadcrumb Manually.....	29
Creating Breadcrumb while creating a Page	35
Creating Calendars.....	37
About Supported Calendars.....	37
Calendar	38
Legacy Calendar	39
Adding a Calendar to a New Page Using a Local Database	39
Adding a Calendar on a New Page Using Remote Database References	41
Creating a Calendar in Page Designer.....	43
Managing Calendar Attributes.....	44
Editing Calendar Attributes in the Property Editor	44
Editing an Existing Calendar to Include Add and Edit Functionality	45
Adding Calendar Initialization JavaScript Code.....	47
Enabling the Dragging and Dropping of Data in an Existing Calendar.....	48
About Dynamic Action Support for Calendar	49
Charts.....	50
Classic Report.....	56
Column Toggle Report.....	59
Forms	61
Help Text.....	67
Interactive Grid.....	71
Interactive Report.....	76

List.....	87
List View	90
Map Chart	93
Plsql Dynamic Content.....	97
Reflow Report.....	101
Region Display Selector	103
Static Content	105
Tree.....	121
Items.....	127
Checkbox.....	127
Display Image	130
File Browse	132
Hidden	141
List Manager	146
Number Field.....	149
Password.....	153
Radio Group	158
Rich Text Editor	163
SELECT LIST.....	166
Shuttle	171
TEXT FIELD.....	174
CUSTOMIZATION.....	175
Text field with Auto Complete.....	177
Page Designer item attributes.....	178

CONTENTS

Working with Textarea	180
Customization in Textarea	181
Buttons	182
CHAPTER 8: PAGE DESIGNER.....	187
Centre Pane	189
CHAPTER 9: SHARED COMPONENTS.....	191
Application Logic	191
Application Definition Attributes.....	191
Security	194
Application Items.....	198
Application Process.....	199
Application Process.....	201
Application Settings.....	203
Build Options	203
Security.....	205
Other Components.....	208
List of Values	208
Plug-ins.....	210
Component Settings	211
Shortcuts.....	212
Email Templates	213
Navigation	214
Lists.....	214
Navigation Menu.....	215

Breadcrumbs.....	216
Navigation Bar List.....	217
User Interface	217
User Interface Attributes.....	217
Themes.....	219
Templates	222
FILES	224
Static Application Files	224
Static Workspace Files	225
REPORTS	227
Report Queries	227
Report Layout.....	228
Globalization	229
Globalization Attributes	229
Text Messages.....	232
Translate Application.....	233
CHAPTER 10: INTEGRATION WITH OTHER DATABASE	
WITHOUT DBLINK	235
ABOUT DOYENSYS	241
ABOUT THE AUTHORS.....	245

INTRODUCTION

Oracle Application Express which is otherwise referred to as APEX and pronounced as **ā'pěks** is a web based Rapid application with low-code development which can be scalable and more secured. The developer who develops application need not be notorious in web development technologies since the APEX does it for you. All you need to do is just drag and drop the objects to make a visually appealing application faster. This feature of drag and drop enables the developer to concentrate on the business problems rather than pondering on coding.

This is a no cost feature of oracle which means you need not spend a penny over the license cost of oracle database. This can run on any oracle database like XE, EE or SE. Oracle Apex has many built-in features such as user interface themes, navigation controls, form handlers, and reports. Oracle Application Express accelerates the application development process to a greater extent.

As an end user or a business user you only need a web browser to access the application built on APEX. The popularity of the web based application is humongous as we need not install a number of software to client system, and as and when we need an update or enhancement we can do in the server itself. This makes the web based application a major go to solution.

You can create application using the inbuilt wizards creating components like pages, items, buttons etc. These data entered in each wizard will be saved and archived in

ORACLE APEX

Oracle table as Oracle Apex is built using SQL and PLSQL. Developer can use SQL and PLSQL for server side, and HTML, CSS, and JAVASCRIPT for client side condition.

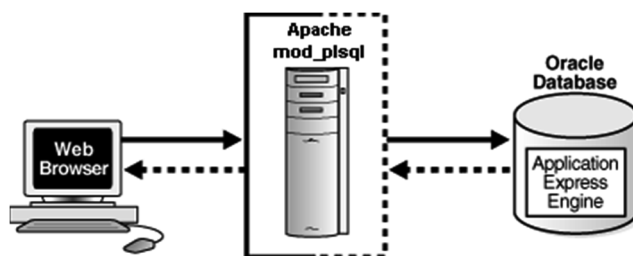
Oracle Apex enables the developer to customize the inbuilt components based on the customer's requirement, making it a tailor fit for them which we will be covering in this book.

ARCHITECTURE

Oracle Apex is a simple 3 tier architecture, which consists of the following—Browser (client side), ORDS (Oracle Rest Data Service), Oracle Apex engine. The request from the client browser is sent to the oracle database as PL/SQL call via ORDS and the request is processed in APEX engine; in other terms oracle database then sends it back to the browser for the client.

ORDS

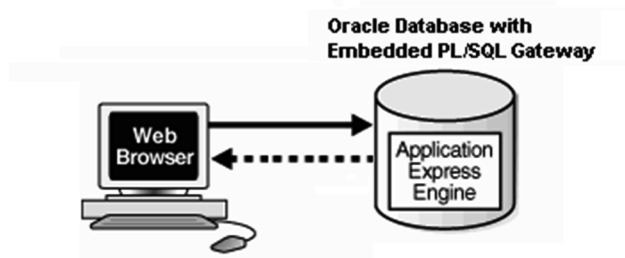
ORDS is the abbreviation of Oracle Rest Data Service. It is a Java program that allows the user to interact with the oracle database via REST service. The SQL or PL/SQL call from the browser is transmitted to the oracle database and the result set from the Oracle Database is sent back to the ORDS which transforms the output to JSON format which will be sent back to the browser.



https://docs.oracle.com/cd/E11882_01/appdev.112/e11947/start_arch.htm#HTMDB25001

ORACLE APEX

In Oracle database 10g and 11g there was 2 tier architecture where the mod_pls gate way is replaced as embedded PL_SQL gateway.



https://docs.oracle.com/cd/E11882_01/appdev.112/e11947/start_arch.htm#HTMDB25001

SKILLS REQUIRED

Oracle Apex developer is someone who is proficient in using oracle database as a powerful RAD platform. The Developer must possess the following skills and we can further drill down to mandatory and optional skills, where the mandatory skills are ones without which one cannot develop the application, and the optional skills are needed only for making your application visually appealing and for customizing the in-built functionality.

MANDATORY SKILLS

- ☐ Oracle Application Express
- ☐ Oracle SQL
- ☐ Oracle PL/SQL
- ☐ Oracle Data Modelling
- ☐ Oracle Database performance and tuning

OPTIONAL SKILLS

- ☐ HTML5
- ☐ CSS
- ☐ Web design
- ☐ Data security
- ☐ Web security

ORACLE APEX

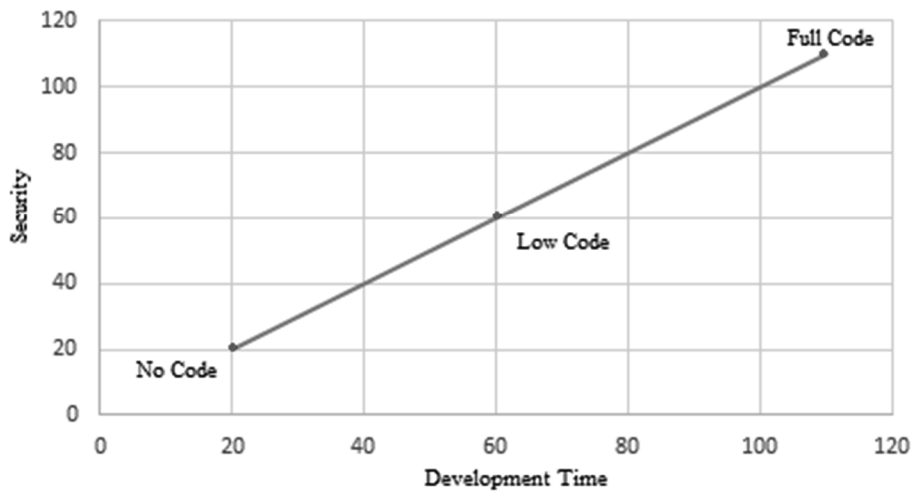
Someone can be called as Oracle APEX developer only if he or she is proficient in at least three of the mandatory skills. If you excel in some of the optional skills, you can be a pleasing developer of oracle APEX.

Note: At Doyensys, we hire people who are experts in Oracle Apex and possess any one of the above skills and hands on experience in them.

PROS AND CONS OF ORACLE APEX

ADVANTAGES

- ❑ It's FREEE!!! Probably we would say it's a no cost feature given by Oracle. As you pay for Oracle database, this comes as a value added service. This comes with all the versions as XE, EE, etc.
- ❑ It's a RAD (Rapid Application Development) tool which helps the developer to develop an application in a fast and secure manner.
- ❑ It's a low-code development environment, where we can write the code, and the tool will take off some of the development process. Hence we will have both secured and fast development. In the other scenario, if the tool generates the whole code and if the developer is not allowed to change it, then security will be a major concern, even though the development timing will be greatly reduced. But if the developer needs to do full coding, which in turn will be a time consuming process, it will greatly increase the development time. The below graph shows the advantages of low-code development.



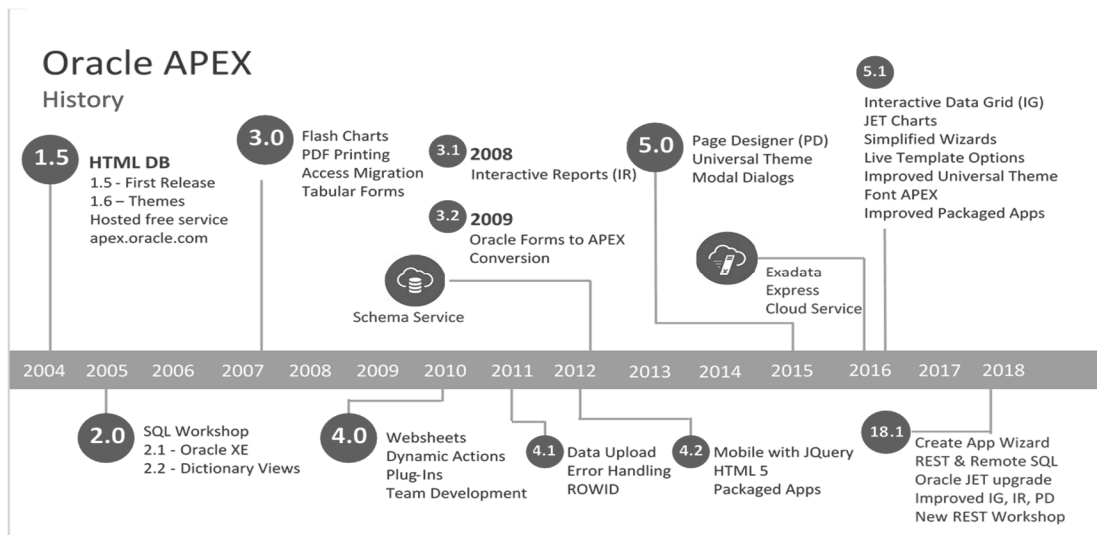
DISADVANTAGES

- There is no version management in Oracle APEX which takes rework if changes are needed to the previous versions.

ROAD MAP OF APEX

Oracle APEX was developed way back in 2004 when it was called HDML Db, which was the first release. In each version Oracle added a functionality to make it a good product. Oracle concentrated on each technology at each release, like in version 2, it was added with SQL workshop, where it acts as GUI for executing query, which has query builder as inbuilt for the ease of developers. Then they moved on in improving the user interface where they have built charts and reports. Later they have added migration guide and support for Oracle Forms to APEX, which enable all the Oracle forms users to switch to latest technology, and helped them to speed up their development. Without some energy the application will not feel lively, hence they added dynamic action and plugins which were handled by the developers using JavaScript. Inclusion of team development added an advantage for the team of members to track the progress of the development.

The concentration of Oracle turned to mobile application as the world has shrunk to individuals' palms, and hence it's time for technology to adapt itself for the same. Hence they added a responsive theme which will cater to this requirement of changing the display based on the screen resolution.



Source: Oracle

<https://www.oracle.com/technetwork/developer-tools/apex/overview/apex-overview-otn-4491378.pdf>

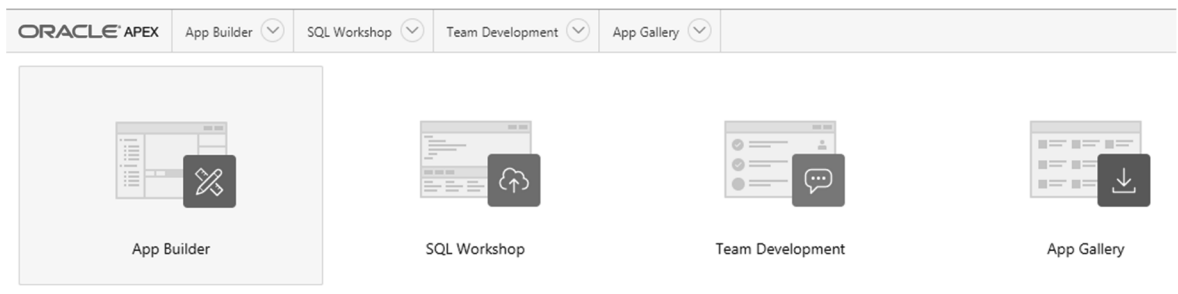
QUICK TOUR TO APEX

In this chapter we will take you through the different modules in Oracle Apex, their functionalities and uses. The following are the four major modules in Oracle Apex.

- ☐ Application Builder
- ☐ SQL Workshop
- ☐ Team Development
- ☐ App Gallery

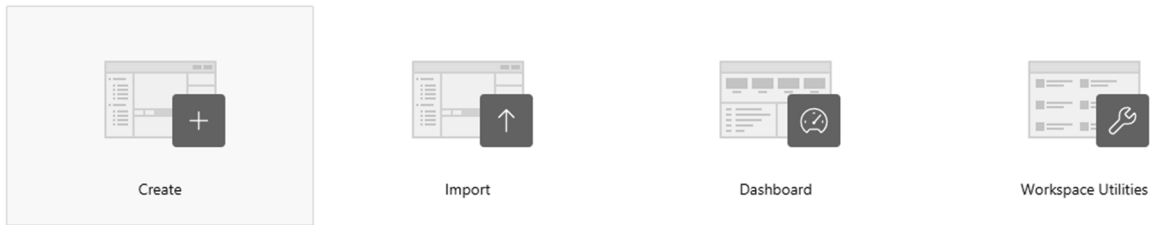
APPLICATION BUILDER

Application builder is the place where the developer can create applications which are composed of several HTML pages based on the Database objects



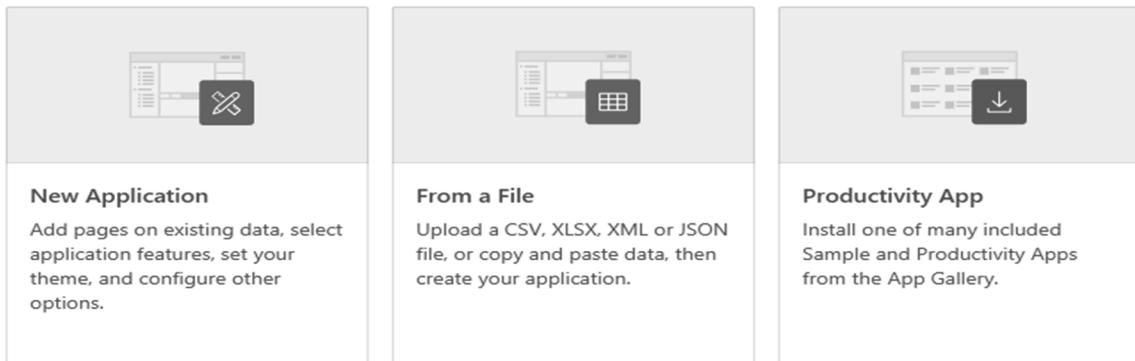
Application builder is further classified into create, import, dashboard, and utilities, where using create option developer can create database application or spread sheet

application or he/she may choose from the preinstalled application which is provided by Oracle.



Create – used to create database application or spreadsheet application or to install productive application

Create an Application



Import – this is the option where the developers can import the existing application which is exported as .sql file. This option also enables developers to import and use other objects like plugin themes; images can also be imported from other workspaces to the current one. This is how we deploy the application to the production. We can export the application from UAT environment and import it to production.

Import

Select the file you wish to import to the export repository. Once imported, you can install your file.

If the imported file is a productivity or sample application export, the installation wizard will allow you to run the app installation scripts after installing the application definition.

* Import file Choose File No file chosen ?

* File Type: ☒ **Database Application, Page or Component Export** ?

- ☐ Worksheet Application Export
- ☐ Plug-in
- ☐ Theme Export
- ☐ User Interface Defaults
- ☐ Team Development Feedback
- ☐ CSS Export [Deprecated]
- ☐ Image Export [Deprecated]
- ☐ File Export [Deprecated]

File Character Set Unicode UTF-8 ?

Cancel Help Next >

Dashboard – It's the combination of different reports and charts about the activities of the user and the activities done over the application.

Workspace Utilities – These utilities are the ones we have in the workspace details app builder; defaults is for setting the default, and manage news is to display news about the application, and workspace themes is to change and add themes, rest enabled SQL services is to manage endpoint urls, and the export option is used for exporting the workspace components

Workspace Utilities	
 App Builder Defaults Define application settings, themes and globalization attributes.	 REST Enabled SQL Services Manage references to external REST Enabled SQL services.
 Manage News Manage news displayed on the Application Express home page and Team Development.	 Remote Servers Manage Remote Server objects used for Web Source Modules and Authentications.
 Application Groups Organize applications into logical groups.	 Web Credentials Manage secure credentials to connect to REST Enabled SQL or other REST services.
 Workspace Themes Manage workspace themes that can be utilized by any application within the workspace.	 Export ... Export workspace components.
 Application Express Views Query the various views against Application Express metadata.	

SQL WORKSHOP

SQL workshop provides you a GUI (Graphical User Interface) for managing database objects such as tables, views, functions, procedures, packages, etc. This is further modularized as the following:

ORACLE APEX

App Builder

SQL Workshop

Team Development

App Gallery


Object Browser


SQL Commands


SQL Scripts


Utilities


RESTful Services

Recently Created Tables		Recent SQL Commands	
DEPT	48 hours ago	select * from EMP	4 hours ago
EMP	48 hours ago		

- ☐ Object Browser.
- ☐ SQL Commands.
- ☐ SQL Scripts.
- ☐ Utilities.
- ☐ RESTful Services.

OBJECT BROWSER

This page has two panes—one is the Selection pane and the other is the Details pane. The Selection pane enables the developer to browse through the database objects. He/she may also use the search bar to narrow down the results. The developer can view the details of the object by using the tabs in the details pane for the selected object.

The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'ORACLE APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'App Gallery'. The user is logged in as 'Rajan'. The 'Object Selection Pane' on the left shows the 'EMP' table selected under the 'DEPT' schema. The 'Object Detail Pane' on the right displays the 'EMP' table details, including columns, data types, nullability, and primary keys.

Column Name	Data Type	Nullable	Default	Primary Key
EMPNO	NUMBER(4,0)	No	-	1
ENAME	VARCHAR2(10)	Yes	-	-
JOB	VARCHAR2(9)	Yes	-	-
MGR	NUMBER(4,0)	Yes	-	-
HIREDATE	DATE	Yes	-	-
SAL	NUMBER(7,2)	Yes	-	-
COMM	NUMBER(7,2)	Yes	-	-
DEPTNO	NUMBER(2,0)	Yes	-	-

SQL COMMANDS

This is the work place where the developer can execute the SQL scripts or queries. The results of the query can be viewed in the result tab. We can also see the explain plan for the query for tuning the performance of the query. You can save the query if needed for future reference. If you forget to save, the history tab will help you to find the series of executed queries.

The screenshot shows the Oracle APEX SQL Commands interface. The top navigation bar is the same as the previous screenshot. The 'SQL Commands' section is active, showing a query: `select * from EMP`. The 'Results' tab is selected, displaying the query results in a table format.

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT	-	11/17/1981	5000	-	10
7698	BLAKE	MANAGER	7839	05/01/1981	2850	-	30
7782	CLARK	MANAGER	7839	06/09/1981	2450	-	10

SQL SCRIPTS

This module is for managing the scripts which are needed for the application. Say Doyensys has created an application for managing Item master in EBS suite Oracle Apex, and we have a client asking for the same. Hence we load all the scripts for database object creation; master data insertion can be added and executed after the installation.

ORACLE APEX

App Builder

SQL Workshop

Team Development

App Gallery

Q

Rajan

SQL Scripts

Q

Go

Actions

Delete Checked

Upload

Quick SQL

Create

Tasks

Edit

Owner

Name

Created

Updated By

Updated

Bytes

Results

Run

CAPPANDRAJ@GMAIL.COM

-

47 hours ago

-

47 hours ago

17

0

1 - 1

Manage Results

Show Quotas

Export

Import

UTILITIES

There are different utilities in SQL workshop as follows:

Data Workshop -- Load or unload data using CSV, XLSX, XML or JSON files.

Query Builder -- Build queries graphically by adding tables into a panel, and selecting columns to return.

Quick SQL -- Generate SQL using shorthand syntax.

Sample Datasets -- Install, refresh, or remove sample datasets.

Generate DDL -- Generate scripts for all or selected database objects within a schema.

User Interface Defaults -- Specify layout properties to consistently generate items / columns across pages and applications.

Schema Comparison -- Show differences between database objects in two different schemas.

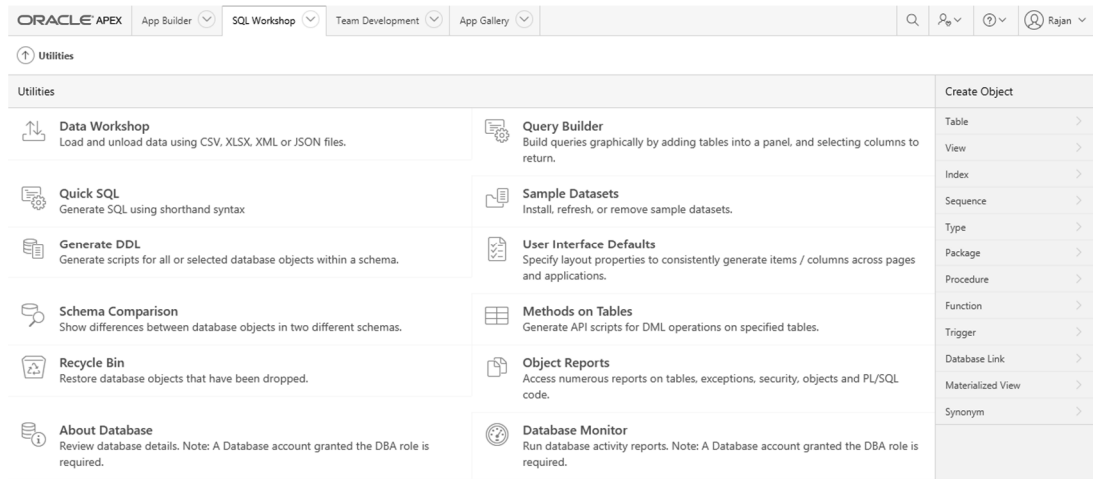
Methods on Tables -- Generate API scripts for DML operations on specified tables.

Recycle Bin -- Restore database objects that have been dropped.

Object Reports -- Access numerous reports on tables, exceptions, security, objects and PL/SQL code.

About Database -- Review database details. For this schema needs DBA role.

Database Monitor -- Run database activity reports. For this schema needs DBA role is required.



ORDS RESTful Services

ORDS, or Oracle RESTful Data Service is the webservice for electronic data transfer from one database to another. You can enable ORDS for the database using ORDS API. The ORDS can be enabled for the whole schema or for each table based on the requirement. This enhances the security of the database.

```
BEGIN
```

```
  ORDS.enable_schema (
    p_enabled          => TRUE,
    p_schema           => 'TESTUSER1',
    p_url_mapping_type  => 'BASE_PATH',
    p_url_mapping_pattern => 'testuser1',
    p_auto_rest_auth   => FALSE
  );
```

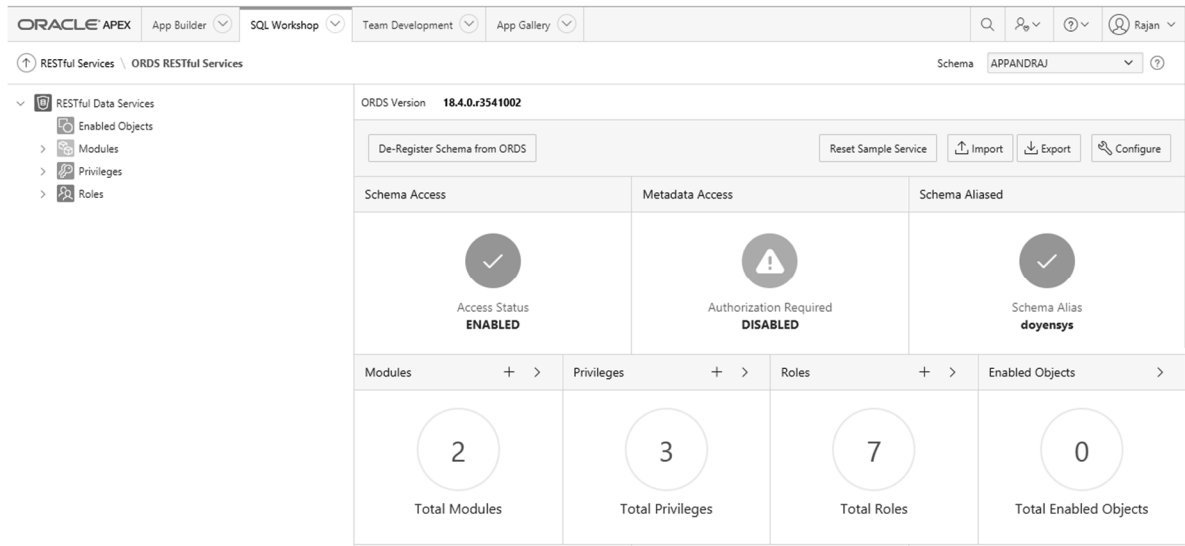
```
END;
```

And we have different API for creation services and methods like GET, PUT, and DELETE for the data manipulation and review.

```
BEGIN
```

ORACLE APEX

```
ORDS.define_service(  
    p_module_name      => 'testmodule1',  
    p_base_path        => 'testmodule1/',  
    p_pattern          => 'emp/',  
    p_method           => 'GET',  
    p_source_type      => ORDS.source_type_collection_feed,  
    p_source            => 'SELECT * FROM emp',  
    p_items_per_page   => 0);  
  
END;
```

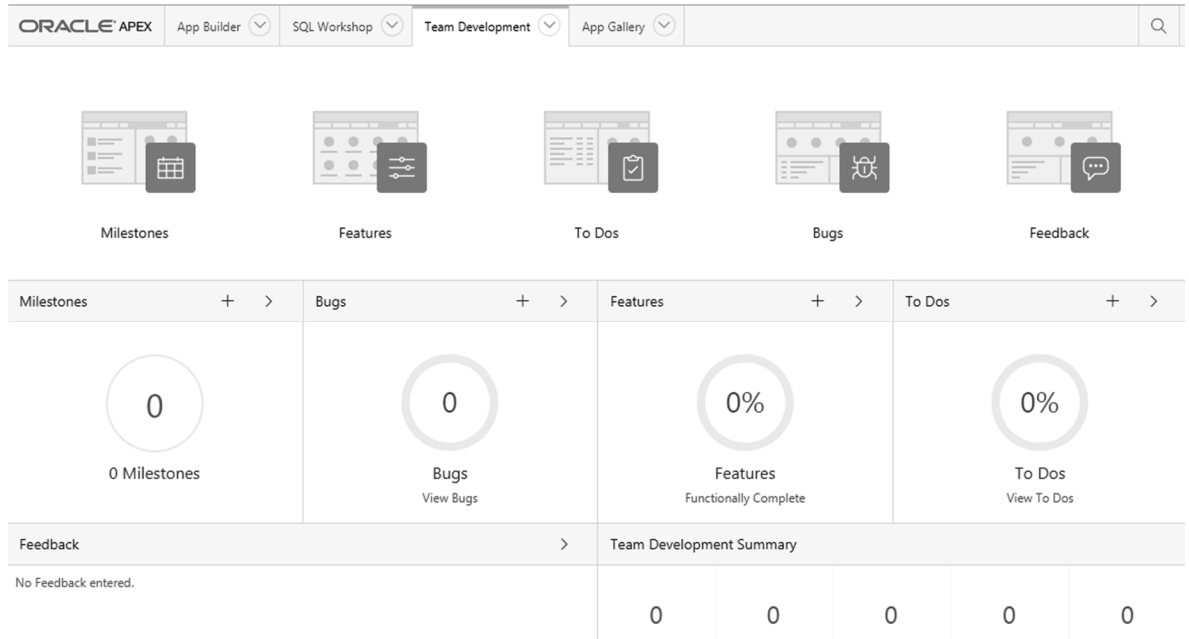


Schema Access	Metadata Access	Schema Aliased
 Access Status ENABLED	 Authorization Required DISABLED	 Schema Alias doyensys

Modules	Privileges	Roles	Enabled Objects
 2 Total Modules	 3 Total Privileges	 7 Total Roles	 0 Total Enabled Objects

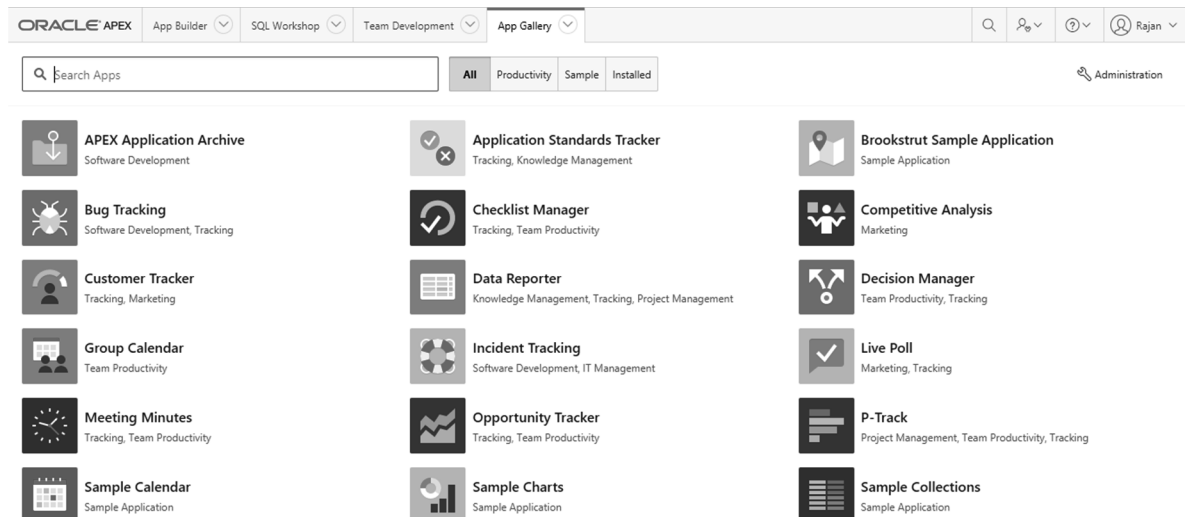
Team Development

This module is for the managing and tracking the development activities of the application. We can split the application to modules and add to the milestone and to-do list. The feedback is the place where the end user's feedback will be displayed to the developers.



App Gallery

Application gallery is the place where Oracle has loaded the prebuilt application



APEX PAGE COMPONENTS

Regions

Breadcrumb

Breadcrumb provides users with hierarchical navigation view. You can display a breadcrumb as a list of links or as a breadcrumb path.

Breadcrumb Creation

You can create Breadcrumb by using two methods:

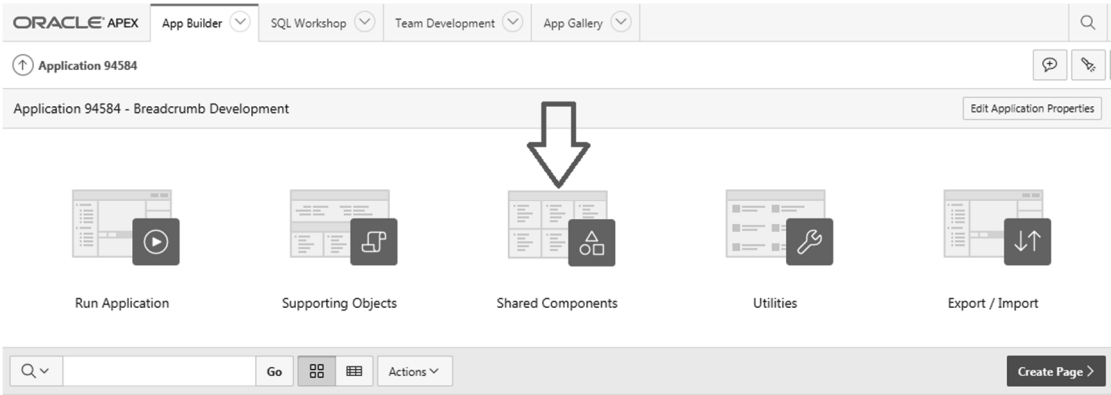
- ☐ Creating Breadcrumb manually
- ☐ Creating Breadcrumb while creating a Page

Creating Breadcrumb Manually

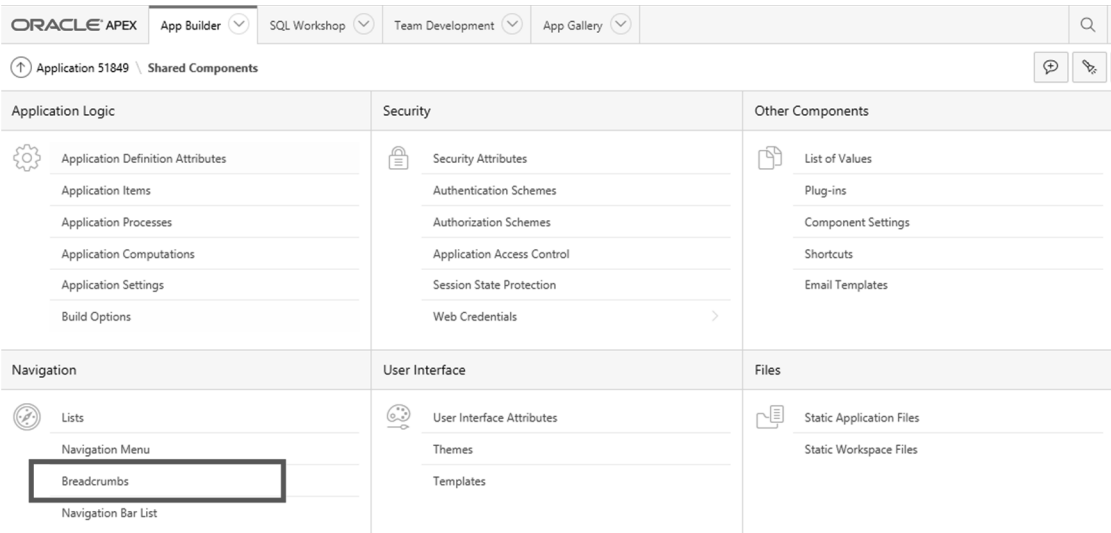
We can create breadcrumb manually in Shared Components and Include the Breadcrumbs in our Page Region

You can create a Breadcrumb manually by using the below steps

Step 1: In App Builder, Select Shared Components



Step 2: In Navigation, Select Breadcrumbs menu



Step 3: Create a new Breadcrumb, by clicking the “Create Breadcrumb” Button and give the Breadcrumb name

Create / Edit Breadcrumb ×

Breadcrumb

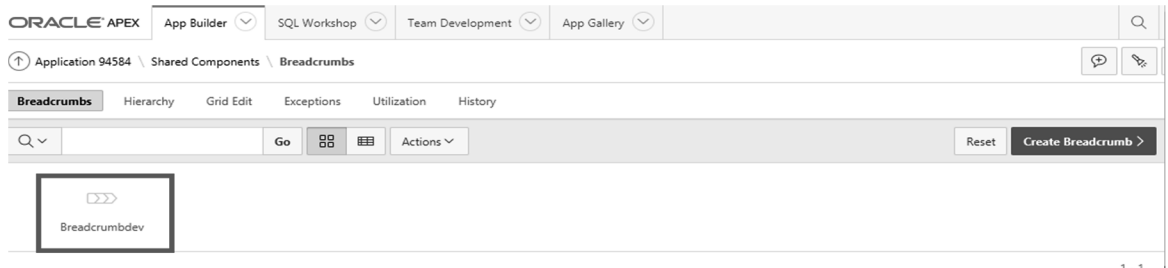
A Breadcrumb is a hierarchical list of links that can be included as a region. Each Breadcrumb is a container for Breadcrumb Entries. Once a Breadcrumb is created you can define breadcrumb entries associated with pages. Each entry can optionally have a parent entry.

Application: **94584 Breadcrumb Development** ?

* Breadcrumb Name ?

Cancel Create

Step 4: Now the breadcrumb has been created with a name “Breadcrumbdev”.



Step 5: Click “Breadcrumbdev” to include the breadcrumb list Entry.

Step 6: Click Create Breadcrumb Entry

Application 94584 \ Shared Components \ Breadcrumbs \ Entries

Breadcrumb

Breadcrumbbdev

?

Name or Target

?

Edit Breadcrumb Name

Create Breadcrumb Entry >

Page

?

Set

Q

Go

Actions

No data found.

Step 7: Select the Page No and Short name to display in a Navigation List as a Parent tab.

Application 94584 \ Shared Components \ Breadcrumbs \ Entries \ Create / Edit

Breadcrumb Entry:

Cancel

Create Breadcrumb Entry

Show All

Breadcrumb

Entry

Target

Conditions

Authorization

Configuration

Breadcrumb

Breadcrumbbdev

?

* Page

2

?

[3]

?

Entry

Sequence

10

?

Parent Entry

- Select Parent -

?

* Short Name

Breadcrumb

?

Long Name

?

Target

Target is a

Page in this Application

?

Step 8: Parent list has been created.

✓ Action processed. ✕

Breadcrumb: Breadcrumbbdev ⓘ Name or Target: ⓘ

Page: ⓘ Set Edit Breadcrumb Name Create Breadcrumb Entry >

Q ▾ Go Actions ▾

Name	Sequence	Page	Parent
Breadcrumbb	10	2	(null)

1 - 1

Here we are setting the Parent Entry as Breadcrumb which we have created above.

Step 9: For Page 3, we have given a breadcrumb short name as “Breadcrumb Sub page”.

Show All Breadcrumb Entry Target Conditions Authorization Configuration

Breadcrumb

Breadcrumb: Breadcrumbbdev ⓘ

* Page: 3 ⓘ [3] ⓘ

Entry

Sequence: 10 ⓘ

Parent Entry: Breadcrumb (Page 2) ⓘ

* Short Name: Breadcrumb sub page ⓘ

Long Name: ⓘ

Target

Target is a: Page in this Application ⓘ

Page: 3 ⓘ

☐ reset pagination for this page ⓘ

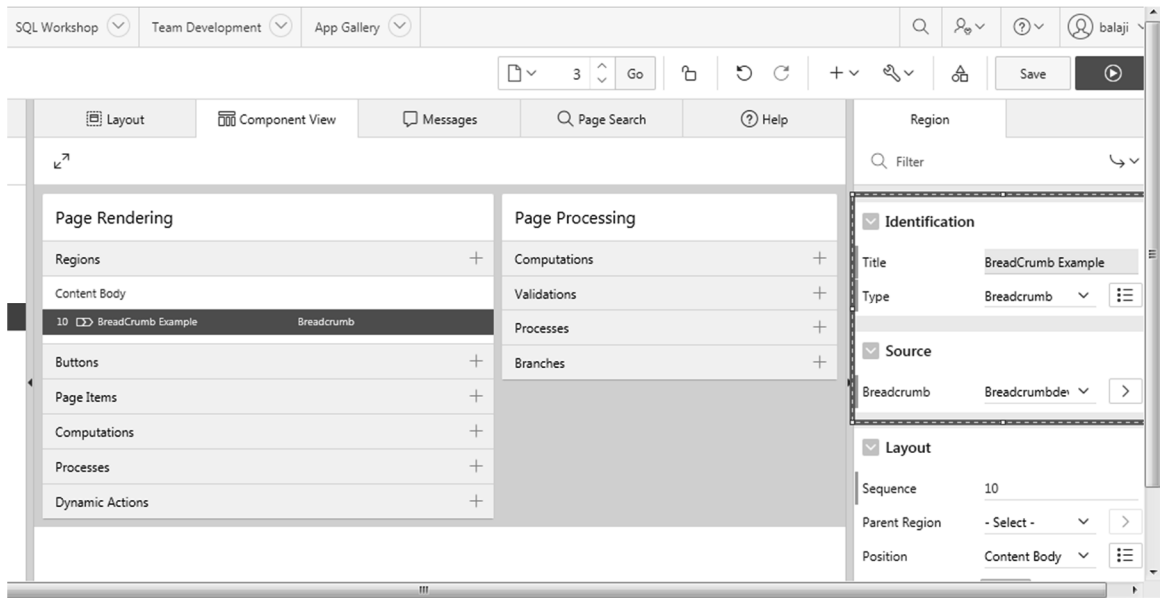
Step 10: Now the Breadcrumb has been Created and below are the steps to include the Breadcrumb in our Page Region

Step 11: Create a new Region in Page 3, with title “BreadCrumb Example”

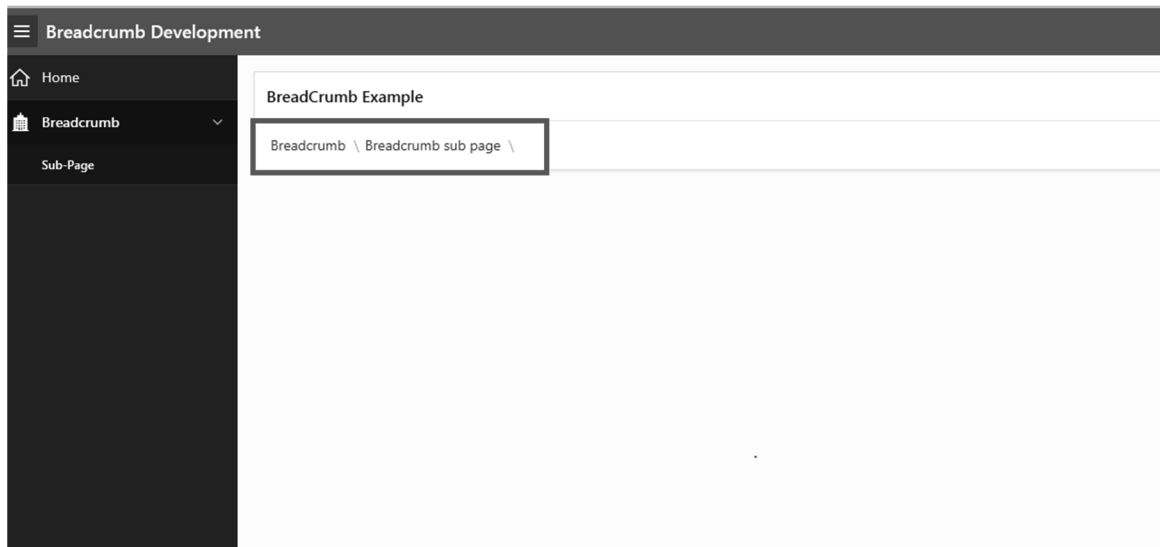
Step 12: In Type, Select “Breadcrumb”

Step 13: In Source, Select Breadcrumb name that we created on Shared Component “Breadcrumbbdev”.

ORACLE APEX



Step 14: Now “Save & Run” the Page. We are able to view the Breadcrumb Region with List type



Step 15: It will display as a hierarchical list; you can select the “Breadcrumb” list, to move for Page 2.

Creating Breadcrumb while creating a Page

Step 1: In Component view, Click [+] to Create new Blank page.

Step 2: In APEX Page creation, we have an option to choose the Breadcrumb.

Step 3: Select the Breadcrumb name and its Parent Entry and create a new page

Create a Blank Page

Page Attributes

* Page Number ?

* Name ?

* Page Mode **Normal** Modal Dialog Non-Modal Dialog ?

Page Group ?

Breadcrumb ?

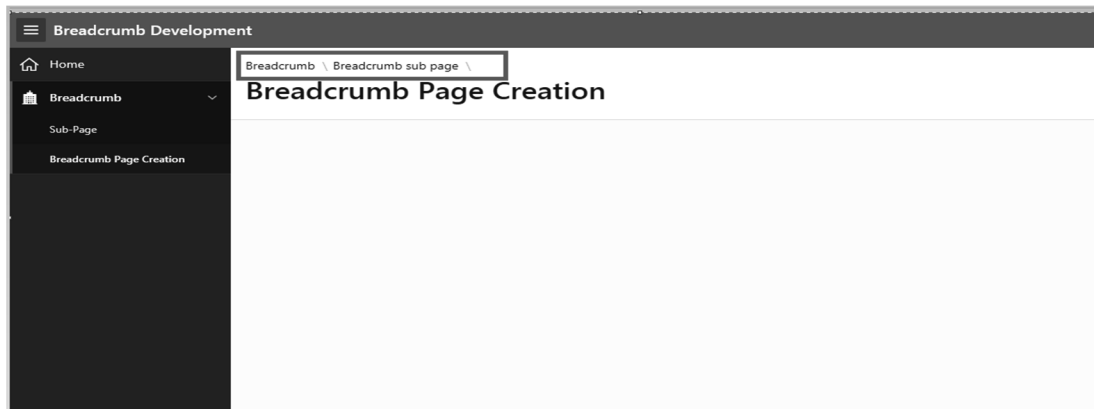
Parent Entry ?

Entry Name ?

> **Optional Static Content Regions**

< Cancel Next >

Step 4: Now the Page has been created with Breadcrumb Region.

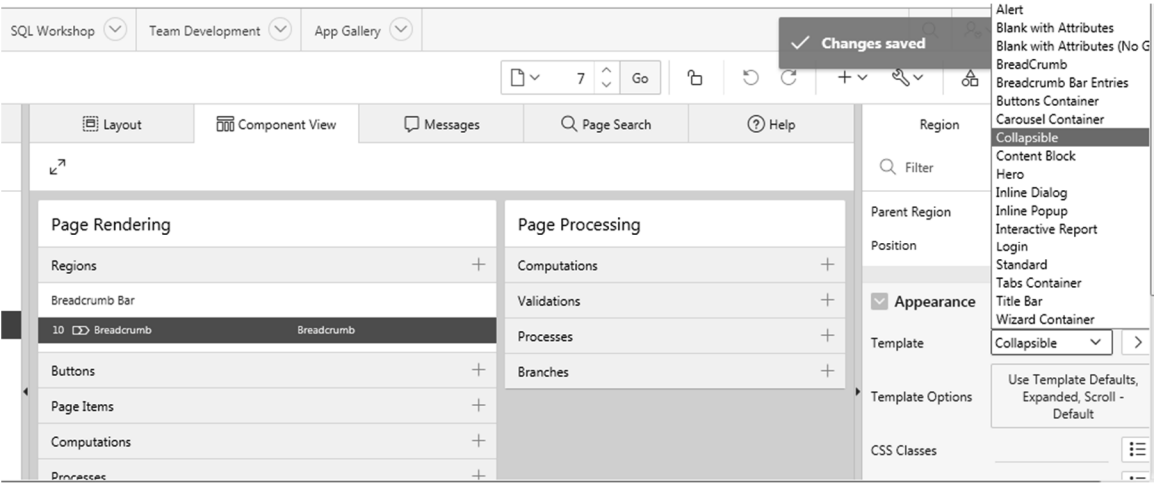


Step 5: You can select the Breadcrumb list to navigate to the corresponding Pages.

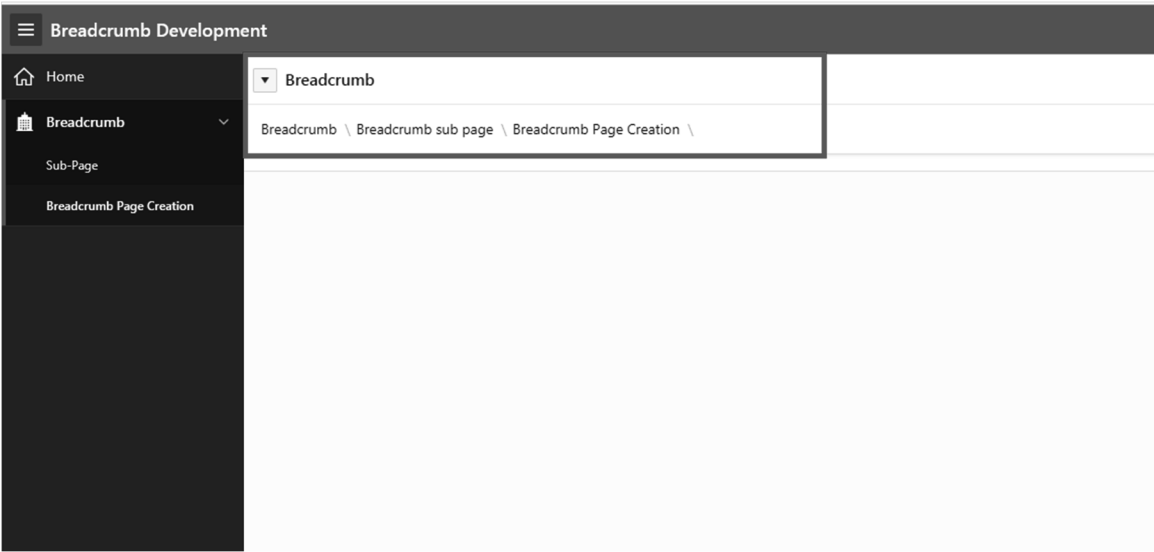
Changing the Appearance of Breadcrumb

By Default, Template for breadcrumb is “Title Bar”.

We can change the template of Breadcrumb in below mentioned types



In this we have chosen “Collapsible”. Now the Page 7 will appear in the below format.



Creating Calendars

App Builder includes support for creating a calendar with monthly, weekly, daily, and list views. Once you specify the table on which the calendar is based, you can create drill-down links to information stored in specific columns and enable drag and drop capability.

App Builder supports two types of calendars: Calendar and Legacy Calendar.

Sample Calendar – You can view a sample calendar by installing the sample application, *Sample Database Application* or the *Sample Calendar* application.

- Adding a Calendar to a New Page Using a Local Database
 - Run the Create Page Wizard to create a new page with a calendar using a local database.
- Adding a Calendar on a New Page Using Remote Database References.
 - Run the Create Page Wizard to add a new page with a calendar that uses a remote database reference.
- Creating a Calendar in Page Designer
 - Manually create a calendar in Page Designer by providing the SQL query.
- Managing Calendar Attributes
 - Once you create a calendar, you can alter the display by editing attributes in the Property Editor. Each calendar has two sets of editable attributes: Region Attributes and Calendar Attributes.
- About Dynamic Action Support for Calendar
 - Dynamic actions provide a way to define complex client-side behaviour declaratively without the need for JavaScript. Using the Dynamic Action Create wizard, you specify an action that is performed when a defined set of conditions occur. You can also specify which elements are affected by the action, and when and how they are affected.

About Supported Calendars

App Builder supports two types of calendars: Calendar and Legacy Calendar.

Calendar is based on the FullCalendar jQuery library, and can only be customized through CSS. *Legacy Calendar* is template-based and may be deprecated in the future release.

Both Calendar and Legacy Calendar enable you to create a calendar based on a table or SQL query you provide. During the creation process, you are prompted to select a date column and display column. Because it offers a better user interface and numerous additional features, Oracle recommends converting all legacy calendars to the latest calendar.

Calendar

- ☐ Rendering of duration and non-duration based events.
- ☐ Ability to modify the start and end dates by dragging and dropping events on different dates. Drag and drop is only supported for local data sources, that is, database objects in the referenced database schema and not on external data sources such as a Google calendar.
- ☐ Ability to change the duration by resizing the length of the events, by changing the end date.
- ☐ Ability to Edit or Add new events on calendar using forms by clicking either on events or empty calendar cell.
- ☐ Rendering events from external sources using web service calls or Google Calendar feeds.
- ☐ Ability to render events titles on single or multiple lines.
- ☐ Using different CSS classes, developer can choose different styles for different types of events.
- ☐ Ability to download events in PDF printable format.
- ☐ Support of multiple formats for sharing events (iCal, CSV, XML).
- ☐ Ability to add tooltip on Month/Week/Day views to make it easier for users to have a quick look at details of each event.

Legacy Calendar

Features of this calendar include:

- ☐ Previous – Navigates to the previous month.
- ☐ Next – Navigates to the next month.
- ☐ Month – Displays a monthly view of all orders.
- ☐ List – Displays a list of all orders.

Adding a Calendar to a New Page Using a Local Database

Run the Create Page Wizard to create a new page with a calendar using a local database.

To create a calendar on a new page using a local database:

- ☐ Run the Create Page Wizard:
 - Navigate to the Workspace home page.
 - Click the **App Builder** icon.
 - Select an application.
 - On the Application home page, click **Create Page**. The Create Page Wizard appears.
- ☐ For Creating a Page:
 - User Interface – Select a user interface for the page (optional). This attribute only displays applications using older themes and for which Desktop and Mobile User Interfaces have been defined.
 - Select a page type – Select **Calendar**.
- ☐ For Page Attributes:
 - Page Number – If you identify a new page number, the wizard creates a new page. If you identify an existing page number, the wizard adds the component to that page.
 - Page Name – Enter a title for the page to be created.
 - Page Mode – Select a page mode.
 - Page Group – Identify the name of the page group you would like to associate with this page.

- Breadcrumb – Select whether to use a breadcrumb navigation control on your page, and which breadcrumb navigation control you want to use.
 - Click **Next**.
- For Navigation Menu, specify the type of navigation to include on this page and click **Next**. The navigation options that display depend upon the current application theme.
- For Data Source, select **Local Database**.
- For Source Type, do one of the following:
 - If the source is a table:
 - Source Type – Select **Table**.
 - Table/View Owner – Select the owner of the table on which you are building a calendar.
 - Table/View Name – Select the table or view on which the calendar is based.
 - Select Columns – Select one or more columns to be included in the calendar.
 - Click **Next**.
 - If the Calendar source is an SQL Query:
 - Source Type – Select **SQL Query**.
 - Enter Region Source – Enter the SQL SELECT statement to be used for the calendar. The SQL SELECT statement must include at least two columns, of which one must be a DATE column.
 - Click **Next**.
- For Settings:
 - Display Column – Select the column which holds the text displayed for events on this calendar.
 - Start Date Column – Select the column which holds the start date for events displayed on this calendar.
 - End Date Column – Select the column which holds the end date for events displayed on this calendar. If this attribute is specified, then the calendar displays duration-based events.

- Show Time – Select the time portion of the date display.
- Click **Create**.

Adding a Calendar on a New Page Using Remote Database References

Run the Create Page Wizard to add a new page with a calendar that uses a remote database reference.

To create a calendar on a new page:

- ☐ Run the Create Page Wizard:
 - Navigate to the Workspace home page.
 - Click the **App Builder** icon.
 - Select an application.
 - On the Application home page, click **Create Page**. The Create Page Wizard appears.
- ☐ For Create a Page:
 - (Optional) User Interface – Select a user interface for the page. This attribute only displays for applications using older themes and for which Desktop and Mobile User Interfaces have been defined.
 - Page Type – Select **Calendar**.
- ☐ For Page Attributes:
 - Page Number – If you identify a new page number, the wizard creates a new page. If you identify an existing page number, the wizard adds the component to that page.
 - Page Name – Enter a title for the page to be created.
 - Page Mode – Select a page mode.
 - Page Group – Identify the name of the page group you would like to associate with this page.
 - Breadcrumb – Select whether to use a breadcrumb navigation control on your page, and which breadcrumb navigation control you want to use.
 - Click **Next**.

- ☐ For Navigation Menu, specify the type of navigation to include on this page and click **Next**. The navigation options that display depend upon the current application theme.
- ☐ For Data Source, select and then configure a remote database reference:
 - For a REST Enabled SQL Service reference:
 - Data Source – Select **REST Enabled SQL Service**.
 - REST Enabled SQL Service – Select a REST Enabled SQL reference.
 - For a Web source:
 - Location – Select **Web Source**.
 - Web Source Module – Select a Web Source Module.
- ☐ For Source Type, do one of the following:
 - If the Calendar source is a Table:
 - Source Type – Select **Table**.
 - Table/View Owner – Select the owner of the table on which you are building a calendar.
 - Table/View Name – Select the table or view on which the calendar is based.
 - Select Columns – Select one or more columns to be included in the calendar.
 - Click **Next**.
 - If the Calendar source is an SQL Query:
 - Source Type – Select **SQL Query**.
 - Enter Region Source – Enter the SQL SELECT statement to be used for the calendar. The SQL SELECT statement must include at least two columns, of which one must be a DATE column.
 - Click **Next**.
- ☐ For Settings:
 - Display Column – Select the column which holds the text displayed for events on this calendar.
 - Start Date Column – Select the column which holds the start date for events displayed on this calendar.

- End Date Column – Select the column which holds the end date for events displayed on this calendar. If this attribute is specified, then the calendar displays duration-based events.
- Show Time – Select the time portion of the date display.
- Click **Create**.

Creating a Calendar in Page Designer

Manually create a calendar in Page Designer by providing the SQL query.

To create a calendar in Page Designer:

- ☐ View the page in Page Designer:
 - On the Workspace home page, click the **App Builder** icon.
 - Select an application.
 - Select a page. Page Designer appears.
- ☐ If necessary, click the Layout tab in the central pane.
- ☐ In the Gallery, click the **Regions** tab.
- ☐ Right-click **Calendar**, click **Add To**, and select the appropriate location.
- ☐ The Property Editor displays Region attributes. Attributes are organized in groups.
- ☐ To find a group or attribute:
 - **Search for the group or attribute** – Enter keywords in the Filter Properties field. The Property Editor displays the group or attributes. To return to the default display, delete the keywords.
 - **Use Go to Group** – Click **Go to Group** and select the group. To return to the default display, click **Go to Group** again and select **Expand All**.
- ☐ The Messages tab displays a red or yellow badge to identify messages you need to address. Select a message to view the associated attribute in the Property Editor. You must address red error message before you can save.
- ☐ In the Property Editor, edit the appropriate Region attributes: Identification, Title – Enter a region title. Source, Location – Select the location of the database which contains column which holds the CSS Class to style the events displayed on this calendar. Options include:
 - **Local Database** – Data is sourced from the local database.

- **Remote Database** – Data is sourced from a remote database, where the connection is defined using REST Enabled SQL.
- **Web Source** – Data is sourced from a RESTful web service defined using Web Source Modules.
- What displays next depends upon Source, Location you select.
- Edit the calendar Attributes.
 - In the Rendering tab, locate the Calendar region and click **Attributes**. The Property Editor displays the calendar Attributes.
 - Edit the calendar Attributes:
 - Settings, Display Column – Select the column which holds the text displayed for events on this calendar.
 - Settings, Start Date – Select the column which holds the start date for events displayed on this calendar.
- Click **Save**.

Managing Calendar Attributes

Once you create a calendar, you can alter the display by editing attributes in the Property Editor. Each calendar has two sets of editable attributes: region attributes and calendar attributes.

Region attributes define the area of the page which functions as a container for the calendar. Calendar Attributes specify the template, date columns, and general calendar formatting.

- Editing Calendar Attributes in the Property Editor
- Editing an Existing Calendar to Include Add and Edit Functionality
- Adding Calendar Initialization JavaScript Code
- Enabling the Dragging and Dropping of Data In an Existing Calendar

Editing Calendar Attributes in the Property Editor

To edit calendar Attributes:

- View the page in Page Designer:
 - On the Workspace home page, click the **App Builder** icon.

- Select an application.
- Select a page. Page Designer appears.
- In the Rendering tab, locate and select the calendar. The Property Editor displays the region attributes.
- In the Rendering tab, locate the calendar and select the **Attributes** node. The Property Editor displays the calendar attributes. Attributes are organized in groups.
- To find a group or attribute:
 - **Search for the group or attribute** – Enter keywords in the Filter Properties field. The Property Editor displays the group or attributes. To return to the default display, delete the keywords.
 - **Use Go to Group** – Click **Go to Group** and select the group. To return to the default display, click **Go to Group** again and select **Expand All**.
- Edit the calendar attributes.
- To save your changes click **Save**. To save and run the page, click **Save and Run Page**.

Editing an Existing Calendar to Include Add and Edit Functionality

Developers can edit an existing calendar to include add and edit functionality.

To edit an existing calendar to include add and edit functionality:

- View the page in Page Designer:
 - On the Workspace home page, click the **App Builder** icon.
 - Select an application.
 - Select a page. Page Designer appears.
- In the Rendering tab, locate and select the calendar. The Property Editor displays the region attributes.
- In the Rendering tab, locate the calendar and select the **Attributes** node. The Property Editor displays the calendar attributes. Attributes are organized in groups.
 - **Search for the group or attribute** – Enter keywords in the Filter Properties field. The Property Editor displays the group or attributes. To return to the default display, delete the keywords.

- **Use Go to Group** – Click **Go to Group** and select the group. To return to the default display, click **Go to Group** again and select **Expand All**.
- **Edit Settings, Create Link** to specify a target page to call when the user clicks an empty cell or an existing calendar entry.
 - Settings, Create Link – Click **No Link Defined**. The Link Builder - Create Link dialog appears.
 - Type – Select the type of link target.
 - Application – If you select **Page in a different application**, enter the application ID.
 - Page – Specify the target page number.
 - Set Items – Configure the values to be passed from the calendar to the target page. The **Create Link** is called after a date or date range has been selected in the calendar. Therefore, the selected start and end dates must be passed to the Create Page.
 - To pass the selected date or the start date of a selected date range:
 - Name – Specify the name of the target page item containing the start date of the event. If a form, this should be a date picker element.
 - Value – &APEX\$NEW_START_DATE.
 - When a date range has been selected, pass the end date as follows:
 - Name – Specify the name of the target page item containing the end date of the event. For a form, this should be a date picker element.
 - Value – &APEX\$NEW_END_DATE.
 - You can pass additional values in the same way as any other link to an application page.
 - Click **OK**.
- **Edit Settings, Edit Link** to define a target page to be called when the user clicks an existing entry.

Tip: To create an Edit Link, you must pass a primary key value to the target page in order for the form to look up the table row.

- Settings, Edit Link – Click **No Link Defined**. The Link Builder - Create Link dialog appears.

- ☐ **Type** – Select the type of link target.
- ☐ **Application** – If you select **Page in a different application**, enter the application ID.
- ☐ **Page** – Specify the target page number.
- ☐ **Set Items** – Configure the values to be passed from the calendar to the target page. The Create Link is called after a date or date range has been selected in the calendar. Therefore, the selected start and end dates must be passed to the Create Page.
 - **Name** – Specify the name of the target page item containing the primary key value; typically a hidden item, for example: PX_ID.
 - **Value** – Specify the table or SQL query column containing the primary key value, for example: &ID.
 - **Click OK.**
- ☐ **Click Save.**

Adding Calendar Initialization JavaScript Code

You can add a JavaScript function to override the standard jQuery FullCalendar initialization attributes by editing the Initialization JavaScript Code attribute. The function must return a JavaScript Object containing the calendar initialization attributes as parameters.

To add initialization JavaScript code:

- ☐ **View the page in Page Designer:**
 - On the Workspace home page, click the **App Builder** icon.
 - Select an application.
 - Select a page. Page Designer appears.
- ☐ In the Rendering tab, under the calendar title, select the **Attributes** node. The Property Editor displays the calendar attributes. Attributes are organized in groups.
- ☐ To find a group or attribute:
 - **Search for the group or attribute** – Enter keywords in the Filter Properties field. The Property Editor displays the group or attributes. To return to the default display, delete the keywords.
 - **Use Go to Group** – Click **Go to Group** and select the group. To return to the default display, click **Go to Group** again and select **Expand All**.

- ☐ Find **Advanced**.
- ☐ For Initialization JavaScript Code, enter a JavaScript function to override the standard jQuery FullCalendar initialization attributes.
- ☐ Click **Save**.

Enabling the Dragging and Dropping of Data in an Existing Calendar

The Calendar Region supports moving or changing calendar events with drag and drop. To enable support for drag and drop for a calendar region:

- ☐ View the page in Page Designer:
 - On the Workspace home page, click the **App Builder** icon.
 - Select an application.
 - Select a page. Page Designer appears.
- ☐ In the Rendering tab under the calendar title, select the **Attributes** node. The Property Editor displays the attributes for the calendar. Attributes are organized in groups.
- ☐ To find a group or attribute:
 - **Search for the group or attribute** – Enter keywords in the Filter Properties field. The Property Editor displays the group or attributes. To return to the default display, delete the keywords.
 - **Use Go to Group** – Click **Go to Group** and select the group. To return to the default display, click **Go to Group** again and select **Expand All**.
- ☐ Find **Settings** and edit the following attributes:
 - Settings, Primary Key Column – Select a table or SQL Query column. The Drag and Drop field appears in the Property Editor, below Additional Calendar Views.
 - Settings, Drag and Drop – Select **Yes**. Drag and Drop PL/SQL Code appears.
 - Settings, Drag and Drop PL/SQL Code – Enter the PL/SQL code to execute when the end user finishes a drag and drop for an event. Within the PL/SQL code, you can reference the ID of the dragged event, the new start, and end dates as: `:APEX$PK_VALUE`, `:APEX$NEW_START_DATE` and `:APEX$NEW_END_DATE`

- For example, the following code changes the event row in the PROJECTS table after the end user has dragged the event to a new date.


```
begin update projects set start_date =
to_date(:APEX$NEW_START_DATE, 'YYYYMMDDHH24MISS'),
end_date = to_date(:APEX$NEW_END_DATE, 'YYYYMMDDHH24MISS')
where id = :APEX$PK_VALUE; end;
```
 - To learn more, click the **Help** tab in the central pane.
- ☐ Click **Save**.

About Dynamic Action Support for Calendar

Dynamic actions provide a way to define complex client-side behaviour declaratively without the need for JavaScript. Using the Dynamic Action Create wizard, you specify an action that is performed when a defined set of conditions occur. You can also specify which elements are affected by the action, and when and how they are affected.

When working with dynamic actions, you should be mindful of the fact that the more dynamic actions you add to a page, the greater your overall page size. This is because the dynamic action framework emits additional code to the client for each dynamic action defined, which then also must be downloaded and executed by the framework in the client.

Dynamic actions contain a condition that can be specified to control the action. Oracle Application Express components such as reports or charts can react to specific calendar events. For example, if you view a calendar and click a button to view a subsequent month, you can create a dynamic action to automatically refresh an associated report.

To use this feature the developer creates a new dynamic action on the Calendar page in Page Designer. Select the new dynamic action in the Dynamic Action tab in the left pane of Page Designer. Once created, the new dynamic action displays in red with the label **New**. Then, the developer edits the dynamic action attributes in the Property Editor. In Property Editor, locate and expand **When** attribute. Under the **When** attribute, for **Event**, select one of the following events:

Date Selected [Calendar]

This event fires when the user selects an empty date or date range either with the mouse or with the keyboard. If the developer provides a "create link" in the calendar attributes, no event fires.

Information about the current view is being passed as the "data" object:

```
{ "newStartDate"::: "newEndDate"::: } { "newStartDate": {first day of
selection} (YYYYMMDDHH24MISS), "newEndDate": {last day of selection}
YYYYMMDDHH24MISS }
```

Event Selected [Calendar]

This event fires when the user clicks an existing event with the mouse or selects it with the keyboard. When the developer provided an "edit link" in the Calendar attributes, no event is being fired. The "data" object contains the jQuery FullCalendar JSON representation of the event.

View Changed [Calendar]

This event fires when either the view type changes (day, week, month, list) or when the view changes to its previous or next view page. The "data" object contains the following information:

```
{ "viewType": {view type, day, week, month or list}, "startDate": {first
day of view} (YYYYMMDDHH24MISS), "endDate": {last day of view}
(YYYYMMDDHH24MISS) }
```

Note: To view examples of this functionality, run the Sample Calendar sample application. From the navigation menu, select the **Calendar and Dynamic Actions** and select the calendar event. See "Installing a Packaged Application."

Charts

Oracle JET (JavaScript Extension Toolkit) is a collection of open source JavaScript libraries with a set of Oracle contributed JavaScript libraries which helps us to create Interactive charts easily. We can create a chart by referring the below URL.

<https://docs.oracle.com/en/database/oracle/application-express/19.1/htmldb/creating-charts.html#GUID-010717AD-51E9-49A2-8026-1DD472260B45>

In this tutorial we are going to see how it is possible to dynamically change the chart type in oracle apex.

In Oracle JET there are many types of charts you can create, but most used types are:

- ☐ Pie Chart
- ☐ Line Chart
- ☐ Donut Chart
- ☐ Bar Chart

You can easily understand if you have a little knowledge in jQuery and JavaScript.

As a First Step, we are going to Create two different chart types with one select list on page:

- ☐ Create static region and enter the region name as *Dynamic Chart I/P*.
- ☐ Create a select list item in the name of *P8_CHART_TYPE* and choose list of value type as sql query,

```
SELECT    d, r
FROM (SELECT 'Pie' d, 'pie' r, 1 seq
      FROM DUAL
      UNION
      SELECT 'Donut' d, 'donut' r, 2 seq
      FROM DUAL
      UNION
      SELECT 'Funnel' d, 'funnel' r, 3 seq
      FROM DUAL
      UNION
      SELECT 'Pyramid' d, 'pyramid' r, 4 seq
      FROM DUAL
      UNION
      SELECT 'Bar' d, 'bar' r, 5 seq
      FROM DUAL
      UNION
      SELECT 'Area' d, 'area' r, 6 seq
      FROM DUAL
      UNION
      SELECT 'Line' d, 'line' r, 7 seq
      FROM DUAL
      UNION
      SELECT 'Linewitharea' d, 'lineWithArea' r, 8 seq
      FROM DUAL)
```

ORDER BY seq

- ☐ Now, Create a chart region and assign a Static ID as DYN_CHRT.
- ☐ Set Parent region as *Dynamic Chart I/P*.
- ☐ Click on Region attributes and Select the chart type as Pie and choose Value format as Decimal.
- ☐ Place the code given below in the query source,

```
Select JOB, count(EMPNO) value
  from EMP
 where EMPNO is not null
 group by JOB
 order by 2 desc
```

- ☐ Click Save.

Now we have created one chart.

Follow the below given steps to create one more chart,

- ☐ Create a chart region and assign a Static ID as DYN_CHRT1.
- ☐ Set Parent region as *Dynamic Chart I/P*.
- ☐ Click on Region attributes and Select the chart type as Bar.
- ☐ Place the code given below in the query source,

```
Select JOB, count(EMPNO) value
  from EMP
 where EMPNO is not null
 group by JOB
 order by 2 desc
```

- ☐ Click Save.

Until now we have created two charts. Let us see how to add the Javascript part for changing chart type dynamically.

- ☐ Create a dynamic action as follows,
 - Event : Change
 - Selection Type : Items(s)
 - Item(s) : P8_CHART_TYPE
 - Action : Execute Javascript Code
 - Fire on Initialization : Yes

○ Code:

```

var chrt = $('#P8_CHART_TYPE').val();
var chrt_typ1 = ["area", "bar", "line", "lineWithArea"];
var chrt_typ2 = ["funnel", "pie", "pyramid", "donut"];

if (chrt_typ1.indexOf(chrt) >= 0) {
    $('#DYN_CHRT').hide();
    $('#DYN_CHRT1').show();
    apex.region('DYN_CHRT1').widget().ojChart({
        styleDefaults: {
            'threeDEffect': 'on'
        }
    });
    apex.region('DYN_CHRT1').widget().ojChart({
        type: chrt
    });
} else if (chrt_typ2.indexOf(chrt) >= 0) {
    $('#DYN_CHRT1').hide();
    $('#DYN_CHRT').show();
    apex.region('DYN_CHRT').widget().ojChart({
        styleDefaults: {
            'threeDEffect': 'on'
        }
    });
    if (chrt == "pie") {
        apex.region('DYN_CHRT').widget().ojChart({
            type: chrt
        });
        apex.region("DYN_CHRT").widget().ojChart({
            styleDefaults: {
                pieInnerRadius: '0'
            }
        });
    } else if (chrt == "donut") {
        apex.region('DYN_CHRT').widget().ojChart({
            type: "pie"
        });
        apex.region("DYN_CHRT").widget().ojChart({
            styleDefaults: {
                pieInnerRadius: '0.5'
            }
        });
    }
} else {
    apex.region('DYN_CHRT').widget().ojChart({

```

```
        type: chrt
      });
    }

    } else {
      $("#DYN_CHRT").hide();
      $("#DYN_CHRT1").hide();
    }
  }
}
```

- ☐ Click Save.
- ☐ Run a Page and change the chart type dynamically

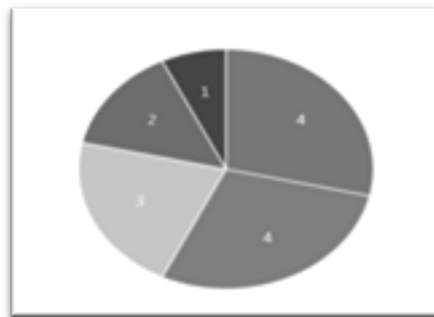
Below are the outputs of data in different Chart types. Based on the input Chart type.

Pie Chart:

Dynamic Chart

Chart Type

Pie



Donut Chart:

Dynamic Chart

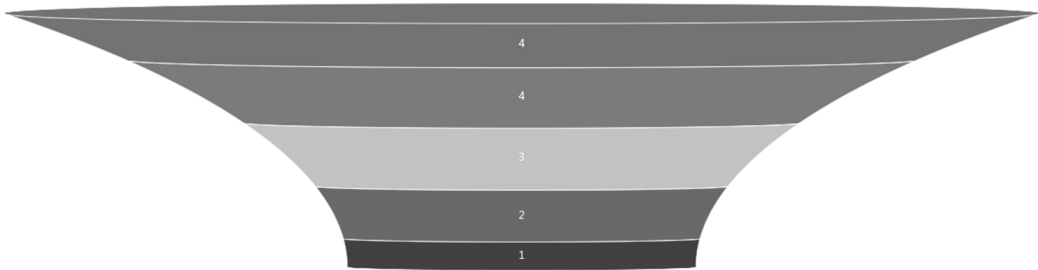
Chart Type
Donut



Funnel Chart:

Dynamic Chart

Chart Type
Funnel



Classic Report

A classic report is the formatted result of an SQL query. Developers can control report layout, pagination, column sorting, error messages, export links, and column break of Classic reports.

To create a simple classic report on page:

- ☐ Create a region and name as **Classic Report** and select type as Classic Report.
- ☐ Select source type as SQL Query and place your code.

Sample code:

```
SELECT dname department, ename Name , job , sal Salary
FROM emp e, dept d
WHERE e.deptno = d.deptno
order by 1, 2
```

The following is an example of a classic report.

Classic Report

Department	Name	Job	Salary
ACCOUNTING	CLARK	MANAGER	2450
ACCOUNTING	MILLER	CLERK	1300
RESEARCH	ADAMS	CLERK	1100
RESEARCH	FORD	ANALYST	3000
RESEARCH	JONES	MANAGER	2975
RESEARCH	SCOTT	ANALYST	3000
RESEARCH	SMITH	CLERK	800
SALES	ALLEN	SALESMAN	1600
SALES	BLAKE	PRESIDENT	5000
SALES	BLAKE	MANAGER	2850
SALES	JAMES	CLERK	950
SALES	MARTIN	SALESMAN	1250
SALES	TURNER	SALESMAN	1500
SALES	WARD	SALESMAN	1250

1 - 14

Customization in Classic report

Classic Report is one of the most versatile components in Application Express because it is template-driven. Using classic report, developers can create their own report templates and visualize data however wanted.

Here we are going to see two samples on how to do customization in Classic report:

- ☐ Merge cell based on Key column
- ☐ Sticky Column Header

Merge cell based on Key column

Here we are going find the column which displays repeated value for the different rows and merge it using minimal JavaScript code.

Let's see how to merge cell based on key column from the following steps.

- ☐ Create a report using the previous steps and assign a static id as **EMP_RPT**. (Here in this report column Department is our key column)
- ☐ Create a dynamic action using
 - Event : After Refresh
 - Selection Type : Region
 - Region : Classic Report (report region)
 - Action : Execute Javascript Code
 - Fire on Initialization : Yes
 - Code :

```
$("#report_EMP_RPT").each(function() {
    var $int_td, cur_val="", int_val="", hdr_cnt="";
    $(this).find("td[headers='DEPARTMENT']").each(function(
    ) {
        cur_val = $(this).text();
        if (int_val == "") {
            int_val = cur_val;
            $int_td = $(this);
            hdr_cnt = 1;
        } else if ( int_val != cur_val ) {
            int_val = cur_val;
            $int_td.attr( "rowspan", hdr_cnt );
            $int_td = $(this);
        }
    });
});
```

```

        hdr_cnt = 1;
    } else if ( int_val == cur_val ) {
        hdr_cnt++;
        $(this).remove();
    }
});
if ( hdr_cnt!="" ){
    $int_td.attr( "rowspan", hdr_cnt );
}
});

```

The following is an example of the custom classic report.

Classic Report			
Department	Name	Job	Salary
ACCOUNTING	CLARK	MANAGER	2450
	MILLER	CLERK	1300
RESEARCH	ADAMS	CLERK	1100
	FORD	ANALYST	3000
	JONES	MANAGER	2975
	SCOTT	ANALYST	3000
	SMITH	CLERK	800
SALES	ALLEN	SALESMAN	1600
	BLAKE	PRESIDENT	5000
	BLAKE	MANAGER	2850
	JAMES	CLERK	950
	MARTIN	SALESMAN	1250
	TURNER	SALESMAN	1500
	WARD	SALESMAN	1250

1 - 14

Sticky Column Header

There is a problem in the classic report like column headings will goes off when we do scrolling. This annoys the user.

Here let's see how to fix this.

- ☐ Create a simple classic report.
- ☐ Place url given below in the Javascript File URLs

```
#IMAGE_PREFIX#libraries/apex/#MIN_DIRECTORY#widget.stickyTableHeader#MIN#.js?v=#APEX_VERSION#
```

- ☐ Then Create dynamic action as follows.
 - Event : After Refresh
 - Selection Type : Region
 - Region : {select your classic report region}
 - Action : Execute javascript code
 - Fire on Initialization : Yes
 - Code :

```
var vRegion$ = $(this.triggeringElement);
vRegion$.setTableHeadersAsFixed();
vRegion$.find('.js-stickyTableHeader').stickyWidget();
```

- ☐ Click Save.

Source: <http://apexbyg.blogspot.com/2017/04/how-to-make-any-table-header-sticky.html>

Column Toggle Report

Overview

Creates a responsive report for mobile applications. By default, column toggle reports are created with all columns set to the same priority. However, the developer can edit the report column attributes and rank columns by importance. Columns with a lesser priority display at narrower widths (for example, columns with priority 6 will be hidden first and columns with priority 1 will be hidden last.) The report includes a Columns button which enables end users to select which columns they want to view.

Use the Column Toggle Report to let user quickly choose columns to display when screen size is limited.

Choose **Column Toggle Report** as Region type.

Examples

☐ Standard

Display a report where end users can select the columns to display at runtime. End users can then change the columns based on the width and orientation of the screen they are using to display the report.

Columns...

Task Name	Start Date	End Date	Status	Assigned To	Cost	Budget
Decommission servers	29-APR-37	27-JUL-37	Pending	Al Bines	\$0.00	\$500.00
Measure effectiveness of improved QA	29-APR-37	29-MAY-37	Pending	Myra Sutcliff	\$0.00	\$1,500.00
Response to Customer Feedback	29-APR-37	21-AUG-37	Pending	Russ Saunders	\$0.00	\$6,000.00
User acceptance testing	14-MAY-37	16-AUG-37	Pending	Russ Saunders	\$0.00	\$2,500.00
End-user Training	16-JUN-37	28-AUG-37	Pending	Myra Sutcliff	\$0.00	\$2,500.00
Rollout sample applications	19-AUG-37	30-AUG-37	Pending	Pam King	\$0.00	\$500.00
Installation	01-MAY-37	31-MAY-37	Pending	Mark Nile	\$0.00	\$1,500.00
Notify users	02-JUN-37	05-JUN-37	Pending	Mark Nile	\$0.00	\$200.00
Plan production release schedule	19-MAR-37	19-MAR-37	Closed	Pam King	\$100.00	\$100.00
Complete questionnaire	15-MAR-37	29-MAR-36	On-Hold	Irene Jones	\$1,200.00	\$800.00
Select servers for Development, Test, Production	29-JAN-37	03-FEB-37	Closed	James Cassidy	\$200.00	\$600.00

Columns...

☒ Task Name
☒ Start Date
☒ End Date
☒ Status
☐ Assigned To
☒ Cost
☒ Budget

Task Name	Start Date	End Date	Status	Cost	Budget
Decommission servers	29-APR-37	27-JUL-37	Pending	\$0.00	
Measure effectiveness of improved QA	29-APR-37	29-MAY-37	Pending	\$0.00	
Response to Customer Feedback	29-APR-37	21-AUG-37	Pending	\$0.00	
User acceptance testing	14-MAY-37	16-AUG-37	Pending	\$0.00	
End-user Training	16-JUN-37	28-AUG-37	Pending	\$0.00	
Rollout sample applications	19-AUG-37	30-AUG-37	Pending	\$0.00	\$500.00
Installation	01-MAY-37	31-MAY-37	Pending	\$0.00	\$1,500.00
Notify users	02-JUN-37	05-JUN-37	Pending	\$0.00	\$200.00
Plan production release schedule	19-MAR-37	19-MAR-37	Closed	\$100.00	\$100.00
Complete questionnaire	15-MAR-37	29-MAR-36	On-Hold	\$1,200.00	\$800.00

☐ No Stripes

Display a report where end users can select the columns to display at runtime. End users can then change the columns based on the width and orientation of the screen they are using to display the report.

							Columns...
Task Name	Start Date	End Date	Status	Assigned To	Cost	Budget	
Decommission servers	29-APR-37	27-JUL-37	Pending	Al Bines	\$0.00	\$500.00	
Measure effectiveness of improved QA	29-APR-37	29-MAY-37	Pending	Myra Sutcliffe	\$0.00	\$1,500.00	
Response to Customer Feedback	29-APR-37	21-AUG-37	Pending	Russ Saunders	\$0.00	\$6,000.00	
User acceptance testing	14-MAY-37	16-AUG-37	Pending	Russ Saunders	\$0.00	\$2,500.00	
End-user Training	16-JUN-37	28-AUG-37	Pending	Myra Sutcliffe	\$0.00	\$2,500.00	
Rollout sample applications	19-AUG-37	30-AUG-37	Pending	Pam King	\$0.00	\$500.00	
Installation	01-MAY-37	31-MAY-37	Pending	Mark Nile	\$0.00	\$1,500.00	
Notify users	02-JUN-37	05-JUN-37	Pending	Mark Nile	\$0.00	\$200.00	
Plan production release schedule	19-MAR-37	19-MAR-37	Closed	Pam King	\$100.00	\$100.00	

							Columns...
Task Name	Start Date	End Date	Status	Assigned To	Cost	Budget	
Decommission servers	29-APR-37	27-JUL-37	Pending		\$0.00		
Measure effectiveness of improved QA	29-APR-37	29-MAY-37	Pending		\$0.00		
Response to Customer Feedback	29-APR-37	21-AUG-37	Pending		\$0.00		
User acceptance testing	14-MAY-37	16-AUG-37	Pending		\$0.00		
End-user Training	16-JUN-37	28-AUG-37	Pending		\$0.00		
Rollout sample applications	19-AUG-37	30-AUG-37	Pending		\$0.00	\$500.00	
Installation	01-MAY-37	31-MAY-37	Pending		\$0.00	\$1,500.00	
Notify users	02-JUN-37	05-JUN-37	Pending		\$0.00	\$200.00	
Plan production release schedule	19-MAR-37	19-MAR-37	Closed		\$100.00	\$100.00	
Complete questionnaire	15-MAR-37	29-MAR-36	On-Hold		\$1,200.00	\$800.00	

☒ Task Name
☒ Start Date
☒ End Date
☒ Status
☐ Assigned To
☒ Cost
☒ Budget

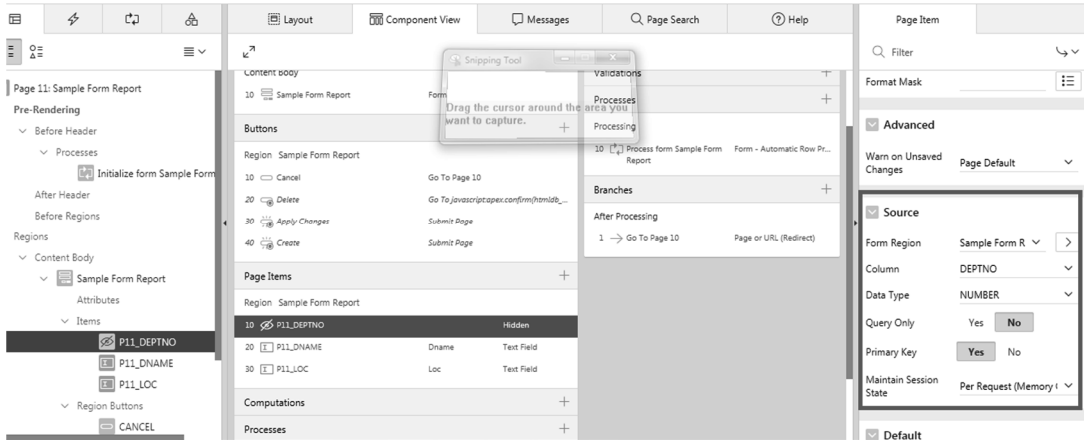
Forms

Form Regions enable user to load and submit the data that are linked to the table columns from a single table. We shall discuss below the form creation methods, the recent updates in form region in Oracle APEX 19.1.

In prior versions of APEX, it will be limited to a single DML load process and a single DML submit process, and all the items on the page that were linked to a table column were required to be from only a single table.

The new components/attributes for Form Regions are:

- ❑ In Form Region type, the source tables are defined in three different methods like Local Database, Rest Enabled SQL Service and Web Source.
- ❑ Item Source attribute must be defined with particular form Region, Query Only and Primary Key options we can set as “Yes/No” based on the requirement.

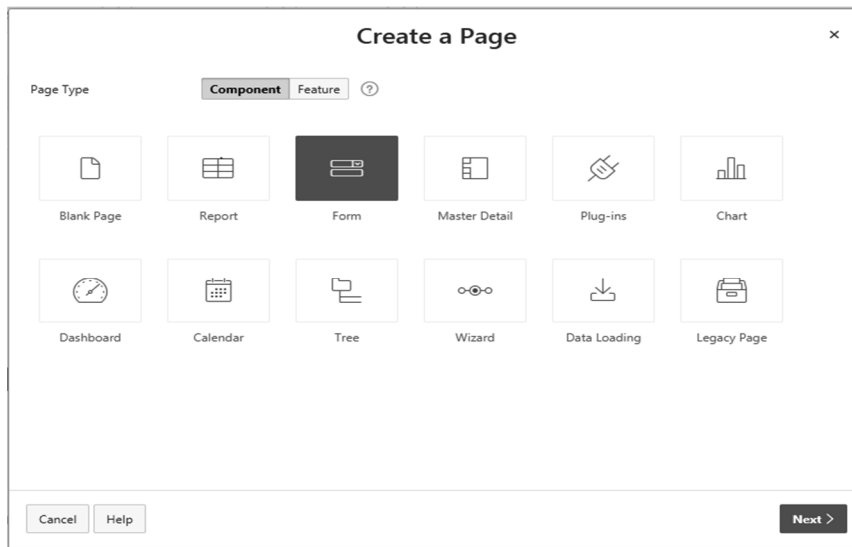


- ❑ Form Initialization and Automatic Row Processing (DML) that is associated with Form region helps us to change the form target region (default is the source table) to a different table or view, or to replace the DML processing with your custom PL/SQL

Source: <https://jeffkemponoracle.com/2019/04/form-regions-superclass-subclass/>

Form Creation

In the *Create Page* wizard, click **Form** and the **Next** button.



Click the “Form” option and click next button. Then click “Report with form” option and “Next” Button.

Create a Report as Interactive Report type and select the form Page Mode as “Normal”.

If you want modal Dialog, then select the Page Mode as “Modal Dialog”.

Data Source

You can provide data source in three different types.

- ☐ Local Database
- ☐ Rest Enabled SQL Service
- ☐ Web Source

Now we will focus on Local Database. Under Local Database we have “Source type” option. There are two source types

- ☐ Table
- ☐ SQL Query

If the Choice is “Table”, then you can choose the table from the list view directly.

If it’s an “SQL Query” then you can give associated query to create a report.

The screenshot shows the 'Data Source' configuration window in Oracle APEX. At the top, there are three tabs: 'Local Database', 'REST Enabled SQL Service', and 'Web Source'. The 'Local Database' tab is selected. Below the tabs, there are two radio buttons for 'Source Type': 'Table' (selected) and 'SQL Query'. Under 'Table', there are two text boxes: 'Table / View Owner' with the value 'BALAJI1991' and 'Table / View Name' with the value 'DEPT (table)'. Below these, there is a section titled 'Select Columns to be shown in the report'. This section contains a list of columns: 'DEPTNO (Number)', 'DNAME (Varchar2)', and 'LOC (Varchar2)'. The 'DEPTNO (Number)' column is highlighted. At the bottom of the window, there are three buttons: a back arrow, 'Cancel', and 'Next >'.

Columns and Primary Key

In the below screen, we can select the columns that need to be displayed in the Form.

Primary Key type

Primary key is mandatory for the Form and we can also manage them either with Default RowID or Select the Primary Key column from the table.

The screenshot shows the 'Create Form - Columns and Primary Key' dialog. On the left, there's a box for 'Select Columns to be displayed in the form'. On the right, a list of columns is shown: DEPTNO (Number), DNAME (Varchar2), and LOC (Varchar2). Below this, the 'Primary Key Type' is set to 'Managed by Database (ROWID)'. The 'Primary Key Column' dropdown is open, showing the same three columns. The 'Select Primary Key Column(s)' option is highlighted.

Set Primary Key Column as “Dept No”.

Click “**Create**” Button to create a Report and the corresponding Form Region.

Interactive Report

Run that page to show the report. It shows only a subset of the **DEPT** table.

Click the Pencil Icon to redirect to the Form Region.

Based on the DeptNo, particular Department details will be displayed in Form Region

Q	Go	Actions	Create
	Deptno	Dname	Loc
✎	10	ACCOUNTING1	NEW YORK
✎	20	RESEARCH	DALLAS
✎	30	SALES	CHICAGO
✎	40	OPERATIONS	BOSTON

Form Region

Form – Initialization and Automatic Row Processing (DML) is automatically created under process section on Before Header by default it’s under source table.

ORACLE APEX

LayoutComponent ViewMessagesPage SearchHelp

10 CancelGo To Page 10

20 DeleteGo To javascript:apex.confirm('htmlidb_...

30 Apply ChangesSubmit Page

40 CreateSubmit Page

Page Items

Region	Sample Form Report
10	P11_DEPTNOHidden
20	P11_DNAMEText Field
30	P11_LOCText Field

Computations

Processes

Before Header

10 Initialize form Sample Form ReportForm - Initialization

Dynamic Actions

Branches

After Processing

1 Go To Page 10Page or URL (Redirect)

Page

Filter

Identification

NameSample Form Report

Page Alias

TitleSample Form Report

Page Group- Select -

Appearance

User InterfaceDesktop

Page ModeNormal

Page TemplateTheme Default

Template OptionsUse Template Defaults

CSS Classes

Media Type

Create a new Department Details

Click the “Create” button and we can create a new department and Location.

Sample Form Report

Dname

test

Loc

test

Cancel

Create

Row created.

GoActions

	Deptno	Dname	Loc
	10	ACCOUNTING1	NEW YORK
	20	RESEARCH	DALLAS
	30	SALES	CHICAGO
	40	OPERATIONS	BOSTON
	50	test	test

1 - 5

We can also edit the department details and click the “Apply changes” to update the information.



The screenshot shows a form with two input fields. The first field is labeled 'Dname' and contains the text 'RESEARCH'. The second field is labeled 'Loc' and contains the text 'DALLAS'. Below these fields are three buttons: 'Cancel', 'Delete', and 'Apply Changes'.

Help Text

Application Builder includes built-in attributes to create Help for your application. Creating Help for your application involves the following steps:

- ☐ Create a dedicated Help page and Help region.
- ☐ Define page Help text.
- ☐ Define item Help text.
- ☐ Create a navigation bar icon to link to your Help page.

Help created in Application Builder displays on a dedicated Help page. To access Help, users click a link that takes them to a dedicated Help page. This Help page displays page and item Help topics specific to the page they are viewing.

Topics in this section include:

- ☐ Creating a Help Page and Region
- ☐ Defining Help Text
- ☐ Creating a Help Navigation Bar Entry

Creating a Help Page and Region

The first step in creating Help for your application is to create a dedicated page and Help Text region. To create a Help Text region:

- ☐ Create a page for your Help.
- ☐ Navigate to the Page Definition of your Help page.
- ☐ Under Regions, select the **Create** icon.
- ☐ When prompted to select a region type, select **Help Text**.

- ☐ Follow the on-screen instructions.

Defining Help Text

You define Help text for a page or single item by editing attributes. Ideally, you would define these attributes as you create your application. For simplicity, however, the following procedures describe how to define this text after the fact.

Topics in this section include:

- ☐ Defining Help for a Page
- ☐ Defining Help Text for an Item
- ☐ Editing Multiple Item Help Topics Simultaneously

Defining Help for a Page

To define page Help text:

- ☐ Navigate to the Page Definition for the page for which you want to add page Help.
- ☐ Under Page, click the **Edit page attributes** icon to view the existing page attributes.
- ☐ Scroll down to **Help**.
- ☐ Enter your Help text in the field provided.
- ☐ Click **Apply Changes**.

Repeat the previous procedure for each page requiring page Help text.

Defining Help Text for an Item

To define item Help text for an item:

- ☐ Navigate to the Page Definition for the page for which you want to add item Help.
- ☐ Under Items, click the name of the item you want to edit.
- ☐ Scroll down to **Help Text**.
- ☐ Enter your Help text in the field provided.
- ☐ Click **Apply Change**.

Repeat the previous procedure for each item requiring Help text.

Editing Multiple Item Help Topics Simultaneously

If you are including item Help in your application, you can edit multiple item Help topics at once using the Bulk Edit Item Help report.

Accessing the Bulk Edit Item Help Report

To view the Bulk Edit Item Help report:

- ☐ On the Workspace home page, click the **Application Builder** icon.
- ☐ Select an application.
- ☐ On the Tasks list, click **Application Reports**.
- ☐ Click **Page Components**.
- ☐ Under Items, click **Item Help Text**. A report displays at the bottom of the page.
- ☐ In Bulk Item Help Report, you can:
 - Update existing Help topics. Edit the Help text that appears and click **Apply Changes**.
 - Link to the Page Definition containing the item by clicking the page number.
 - Link to the Page Item by clicking the item name.

Seeding Item Help Topics

If your application does not yet contain item Help, you can perform a mass update of default Help text. To seed item Help topics:

- ☐ Access the Bulk Edit Item Help report as described in the previous topic.
- ☐ Click **Seed Item Help Text**.
- ☐ In Default Help Text, enter the default text to appear in all Help topics.
- ☐ Click **Apply Changes**.

Searching for Existing Item Help Topics

You can search for existing Help text, or for an item label. To search for existing item Help topic:

- ☐ In Help Contains, enter keywords.
- ☐ Click **Go**.

Searching for an Item Label

To search for an item label:

- ☐ In Help Contains, enter keywords.
- ☐ Click **Go**.

Creating a Help Navigation Bar Entry

Once you have created your Help, the next step is to create a navigation bar entry so users can link to it. To create a navigation bar entry:

- ☐ Navigate to the Page Definition.
- ☐ Under Navigation Bar, click the **Create** icon.
- ☐ For Attributes:
 - Sequence – Specify the sequence for this component. The sequence determines the order of evaluation.
 - Alt Tag Text – Enter ALT text for navigation icons that are images. If you do not specify an image name, then this text displays.
 - Icon Image Name – Enter the name of the image that displays.
 - Image Height – Enter the height of the image.
 - Image Width – Enter the width of the image.
 - Text – Enter additional text to display with the image. You can include text or use icons with no text. This attribute is optional and can be translated.
 - Click **Next**.
- ☐ Next, specify the target location.
- ☐ For Target:
 - Target is a – Select **Page in this application**.
 - Page – Specify the page number of the help page you created in "Creating a Help Page and Region"
 - Request – Enter the following: &APP_PAGE_ID. By specifying substitution string &APP_PAGE_ID as the Request, you are instructing the Application Express engine to display Help text for the current page when the user clicks this icon.
 - Click **Next**.
- ☐ Click **Create**.

Interactive Grid

Interactive grid is a most awaited functionality of oracle apex which combines forms and interactive reports which enable user to not only view the data but also have the privilege of editing the same, and the edited data will be saved to the database. This acts more like an excel sheet where the user can view and save the data further to this, can the modify the view as per needs like charts, or group the data based on some criteria, or even perform arithmetic operation for the data and showcase to the end users. The following image is the typical interactive grid where user can add, edit and save data.

Q	Search: All Text Columns	Go	Actions	Edit	Save	Add Row	Reset
☐	≡	Abbreviation		State Name			
☒	≡	AK		ALASKA			
☐	≡	AL		ALABAMA			
☐	≡	AR		ARKANSAS			
☐	≡	AZ		ARIZONA			
☐	≡	CA		CALIFORNIA			
☐	≡	CO		COLORADO			
☐	≡	CT		CONNECTICUT			
☐	≡	DC		DISTRICT OF COLUMBIA			
☐	≡	DE		DELAWARE			
☐	≡	FL		FLORIDA			
☐	≡	GA		GEORGIA			

There are different options like adding duplicate row as above, delete and refresh row, where the refresh row plays an important role. This enables user to refresh the latest values from the table, as we know if a user is editing a row in Oracle, in turn it locks the particular row, and releases the lock once commit for the change is done. In Doyensys we had a requirement to cascade the select list in the columns of the interactive grid which was not an inbuilt feature.

Step 1: Create an editable Interactive Grid with two dummy columns. Set static id of the region as 'orderdtls'

E.g.:

```
SELECT
    order_id,
    customer_id,
    order_total,
```

ORACLE APEX

```
        order_timestamp,  
        user_name,  
        tags,  
        'N' || order_id checkbox, --dummy column for checkbox  
        'N' status                --dummy column updating status  
FROM  
    demo_orders
```

Step 2: Change the settings of the Checkbox column as follows

- ☐ Type – Display Only
- ☐ Heading – `<span class="igcb fa fa-square-o" id="selectall" onclick="cb_selectall();" style="background:#FFFFFF;color:#000000" value="N"></span>`
- ☐ Column Filter – Enabled - No
- ☐ Sort/Control Break/Aggregate – No
- ☐ Javascript Code:

```
function (config)  
{  
    config.defaultGridColumnOptions =  
        {  
            noHeaderActivate : true  
        };  
    return config;  
}  
  
/*to prevent header options for the checkbox column*/
```

- ☐ Escape Special Character - No

Step 3: Change the settings of the Status column as follows

- ☐ Type – Text Field

Step 4: Add the following code in the Function and Global declaration section of the page

`/*on load function to set unchecked box css for the checkbox column */`

```
apex.jquery(function() {  
    apex.item.create("CHECKBOX", {  
        displayValueFor: function(value) {  
            var ord_id = value.substr(1);  
            var status = value.substr(0,1);
```

```

        var model =
apex.region('orderdtls').widget().interactiveGrid("getViews","grid").model;

//orderdtls - Static ID of the Interactive Grid Region
var colstatus,ord_no,cb_val;

model.forEach(function( object,r )
{
    ord_no      = model.getValue(object,"ORDER_ID");
    colstatus = model.getValue(object,"STATUS");

    if (ord_no == ord_id)
    {
        if ( colstatus == 'Y' )
        {
            cb_val = 'Y';
        }
        else if ( colstatus == 'N' )
        {
            cb_val = 'N';
        }
    }
}));

return '<span aria-hidden="true"
onclick="update_coll_est('+ord_id+');" id="cb_'+ord_id+'" class="igcb fa
' +
        (cb_val === "Y" ? "fa-check-square-o" : "fa-square-o") +
        ' "></span>';
}));
});

//function to select check box
function update_coll_est(req_no)
{
    var model =
apex.region('orderdtls').widget().interactiveGrid("getViews","grid").model;

    var status,ord_id;
    var cb_id = '#'+cb_'+req_no;

    model.forEach(function( object,r )
    {
        var id = model.getRecordId(object);
        ord_id = model.getValue(object,"ORDER_ID");

```

ORACLE APEX

```
        if ( req_no == ord_id)
        {
            status = model.getValue(object,"STATUS");
            if( status.substr(0,1) == 'Y')
            {
                model.setValue(object,"STATUS","N");
                $(cb_id).removeClass('fa-check-square-
o').addClass('fa-square-o');
            }
            else if( status.substr(0,1) == 'N')
            {
                model.setValue(object,"STATUS","Y");
                $(cb_id).removeClass('fa-square-o').addClass('fa-
check-square-o');
            }
        }
    }
}

//function to select all check box
function cb_selectall()
{
    var model =
apex.region('orderdtls').widget().interactiveGrid("getViews","grid").mode
l;
    var status,ord_id,cb_id;

    var value = $('#selectall').attr('value');
    if (value == 'N')
    {
        model.forEach(function( object,r )
        {
            model.setValue(object,"STATUS","Y");
        });
        $('.igcb').removeClass('fa-square-o').addClass('fa-check-
square-o');
        $('#selectall').attr('value','Y');
    }
    else if (value == 'Y')
    { model.forEach(function( object,r )
    {
        model.setValue(object,"STATUS","N");
    });
        $('.igcb').removeClass('fa-check-square-o').addClass('fa-
square-o');
        $('#selectall').attr('value','N');
    }
}
```

```
//function to de select check box when IG report settings changes
```

```
(function($) {
    $(function() {
        $('#orderdtls').on("interactivegridreportsettingschange",
function(event, ui)
    {
        var status = $('#selectall').attr('value');
        if (status == 'Y')
            {cb_selectall(); }
    });
    });
})(apex.jQuery);
```

Screenshot

On load Grid with Unchecked Check Box

Q▼	Search: All Text Columns	Go	Actions▼	Edit		
Order Total	User Name	Order Timestamp	Tags	☐	Status	Remarks
1890	DEMO	03-NOV-17 12.09.37.000000 PM	-	☐	N	-
2380	DEMO	17-NOV-17 12.09.37.000000 PM	LARGE ORDER	☐	N	-
1640	DEMO	28-NOV-17 12.09.37.000000 PM	-	☐	N	-
1090	DEMO	30-NOV-17 12.09.37.000000 PM	-	☐	N	-
950	DEMO	10-DEC-17 12.09.37.000000 PM	-	☐	N	-
1 2 > 1 - 5 of 10						

Multiple selection Check Box and capturing the values in the Status column

Q▼	Search: All Text Columns	Go	Actions▼	Edit		
Order Total	User Name	Order Timestamp	Tags	☐	Status	Remarks
1890	DEMO	03-NOV-17 12.09.37.000000 PM	-	☐	N	-
2380	DEMO	17-NOV-17 12.09.37.000000 PM	LARGE ORDER	☒	Y	-
1640	DEMO	28-NOV-17 12.09.37.000000 PM	-	☒	Y	-
1090	DEMO	30-NOV-17 12.09.37.000000 PM	-	☐	N	-
950	DEMO	10-DEC-17 12.09.37.000000 PM	-	☐	N	-
1 rows selected 1 2 > 1						

Select All Check Box in the Pagination Set 1

Q	Search: All Text Columns	Go	Actions	Edit			
Order Total	User Name	Order Timestamp	Tags	☒	Status	Remarks	
1890	DEMO	03-NOV-17 12.09.37.000000 PM	-	☒	Y	-	
2380	DEMO	17-NOV-17 12.09.37.000000 PM	LARGE ORDER	☒	Y	-	
1640	DEMO	28-NOV-17 12.09.37.000000 PM	-	☒	Y	-	
1090	DEMO	30-NOV-17 12.09.37.000000 PM	-	☒	Y	-	
950	DEMO	10-DEC-17 12.09.37.000000 PM	-	☒	Y	-	
							1 < 2 > 1 - 5 of 10

Select All Check Box in the Pagination Set 2

Q	Search: All Text Columns	Go	Actions	Edit			
Order Total	User Name	Order Timestamp	Tags	☒	Status	Remarks	
1515	DEMO	15-DEC-17 12.09.37.000000 PM	-	☒	Y	-	
905	DEMO	20-DEC-17 12.09.37.000000 PM	-	☒	Y	-	
1060	DEMO	28-DEC-17 12.09.37.000000 PM	-	☒	Y	-	
730	DEMO	03-JAN-18 12.09.37.000000 PM	-	☒	Y	-	
870	DEMO	06-JAN-18 12.09.37.000000 PM	-	☒	Y	-	
							1 < 2 > 6 - 10 of 10

Thus, multiple selection of records can be done and Select All option is available for selecting all the records in all the pagination sets.

Note: If the status column should be hidden, remove the column from report action menu. Do not make the column hidden in the page setting.

Interactive Report

By default the Interactive Reporting Region allows end users to customize reports. This interactive reporting region allows users to customize the layout of the data by choosing the columns they are interested in, applying filters, highlighting, and sorting. They can also define breaks, aggregations, different charts, and their own computations.

Upon the default functionality, the Apex developer can also customize the Interactive Report. Let's see two Samples.

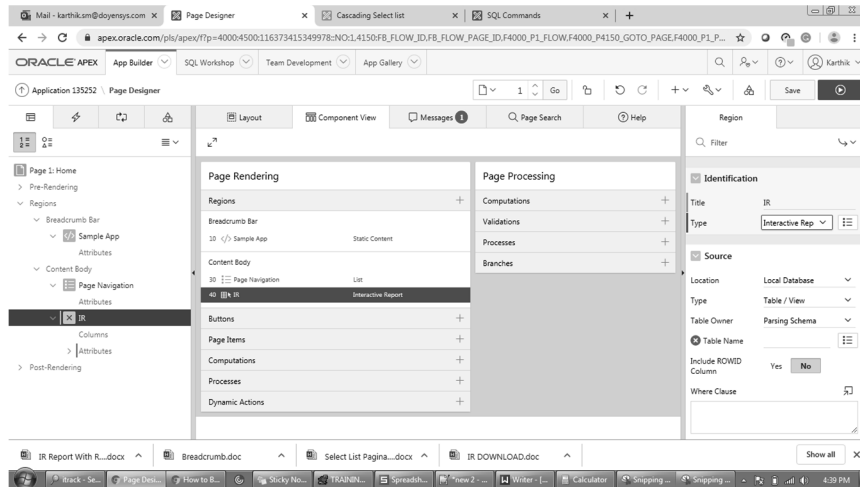
- ☐ Cascading Select list in IR
- ☐ Enable/Disable Download button based on User Role

Cascading Select list in IR

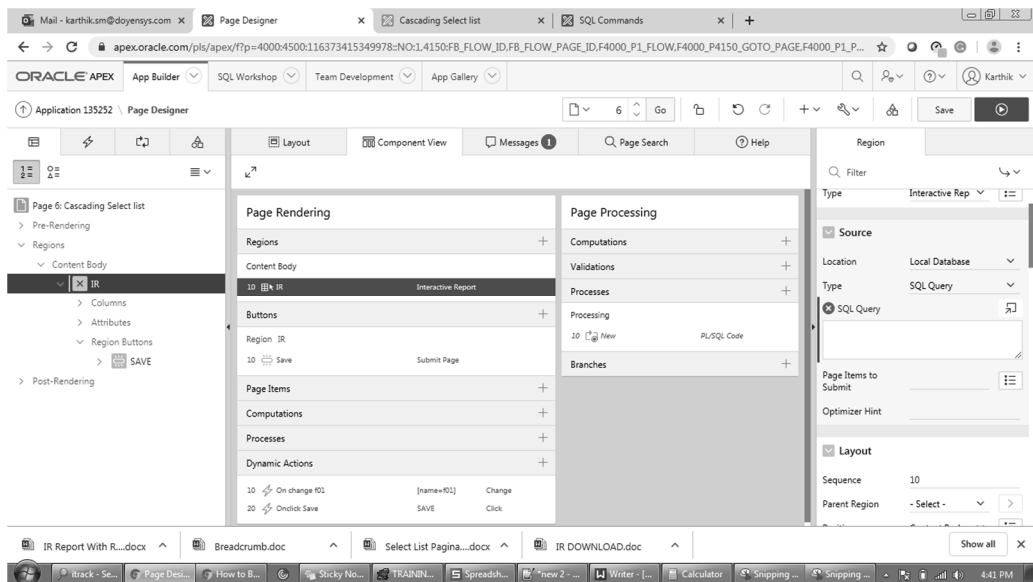
Let's imagine you have an IR report where you want to make it as editable Report with cascading select list. Let's see how to do this in detail.

Follow the steps to create an Editable Interactive Report with cascading select list.

Step 1: In a Page, Create a Region and select report type as Interactive report



Step 2: Change the source type from Table/View to SQL Query

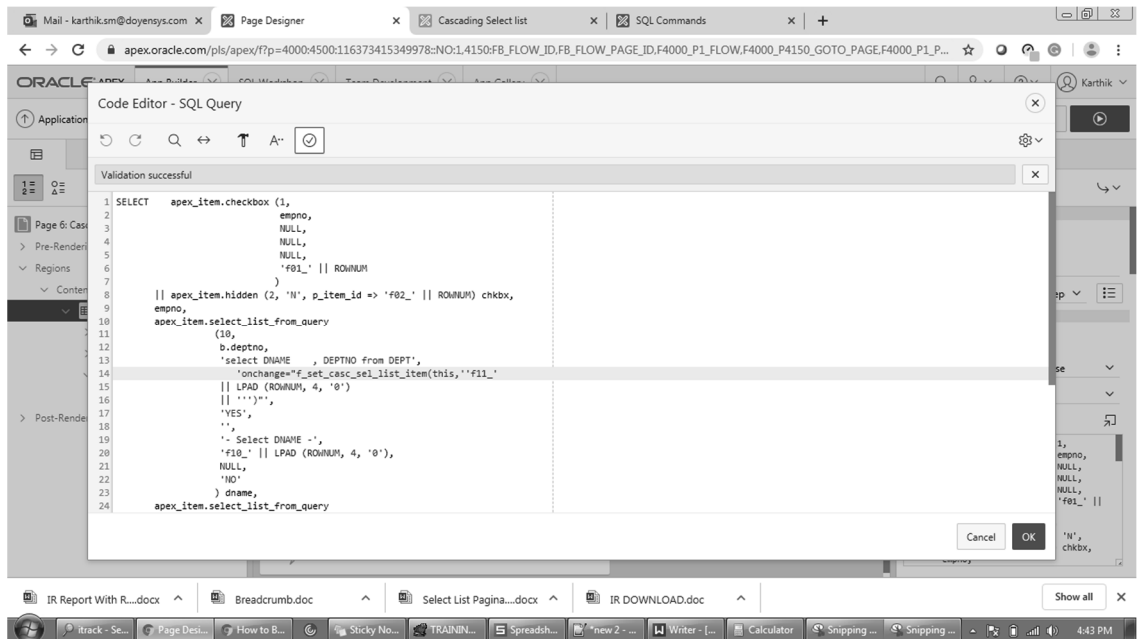


Step 3: Place the below code in the SQL query section,

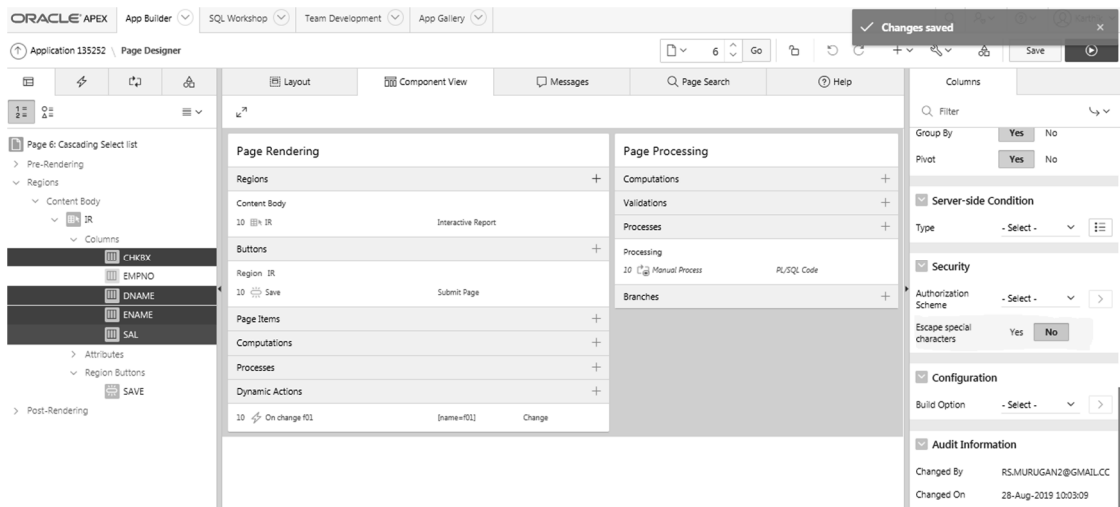
```

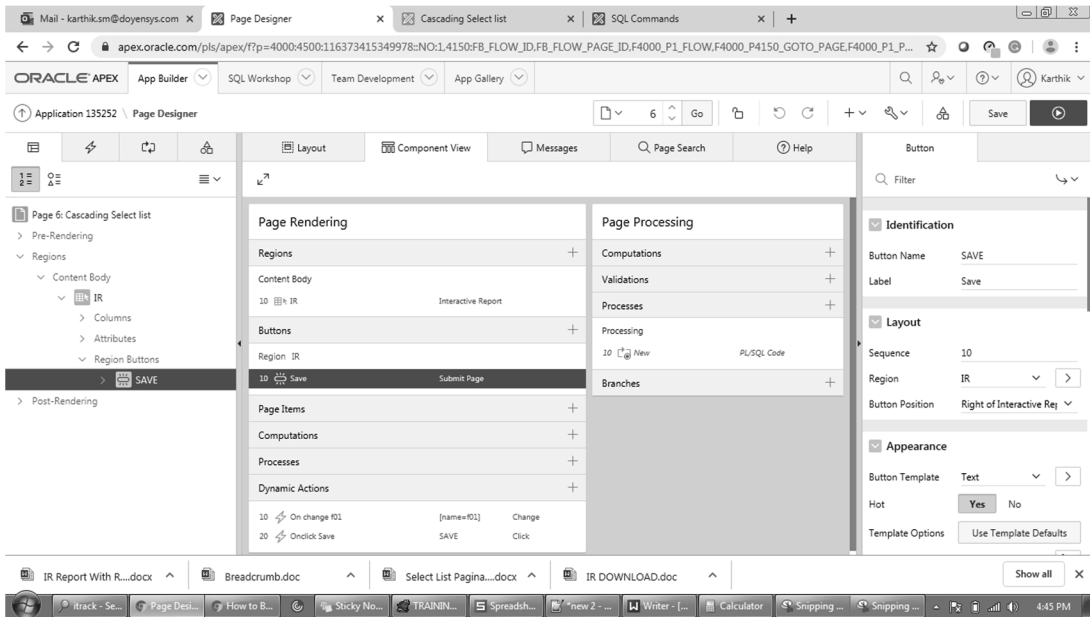
SELECT      apex_item.checkbox (1, empno, NULL, NULL, NULL, 'f01_' ||
ROWNUM)
|| apex_item.hidden (2, 'N', p_item_id => 'f02_' || ROWNUM) chkbx,
empno,
apex_item.select_list_from_query
    (10,
      b.deptno,
      'select DNAME      , DEPTNO from DEPT',
      'onchange="f_set_casc_sel_list_item(this, ''f11_'
|| LPAD (ROWNUM, 4, '0')
|| ''')"',
      'YES',
      '',
      '- Select DNAME -',
      'f10_' || LPAD (ROWNUM, 4, '0'),
      NULL,
      'NO'
    ) dname,
apex_item.select_list_from_query
    (11,
      a.ENAME,
      'SELECT DISTINCT ENAME d, '
|| 'ENAME r FROM EMP where DEPTNO = '
|| a.deptno,
      '',
      'YES',
      '',
      '- Select User -',
      'f11_' || LPAD (ROWNUM, 4, '0'),
      NULL,
      'NO'
    ) ename,
      apex_item.text (12, a.sal, p_item_id => 'f12_' || ROWNUM) sal
FROM emp a, dept b
WHERE a.deptno = b.deptno

```

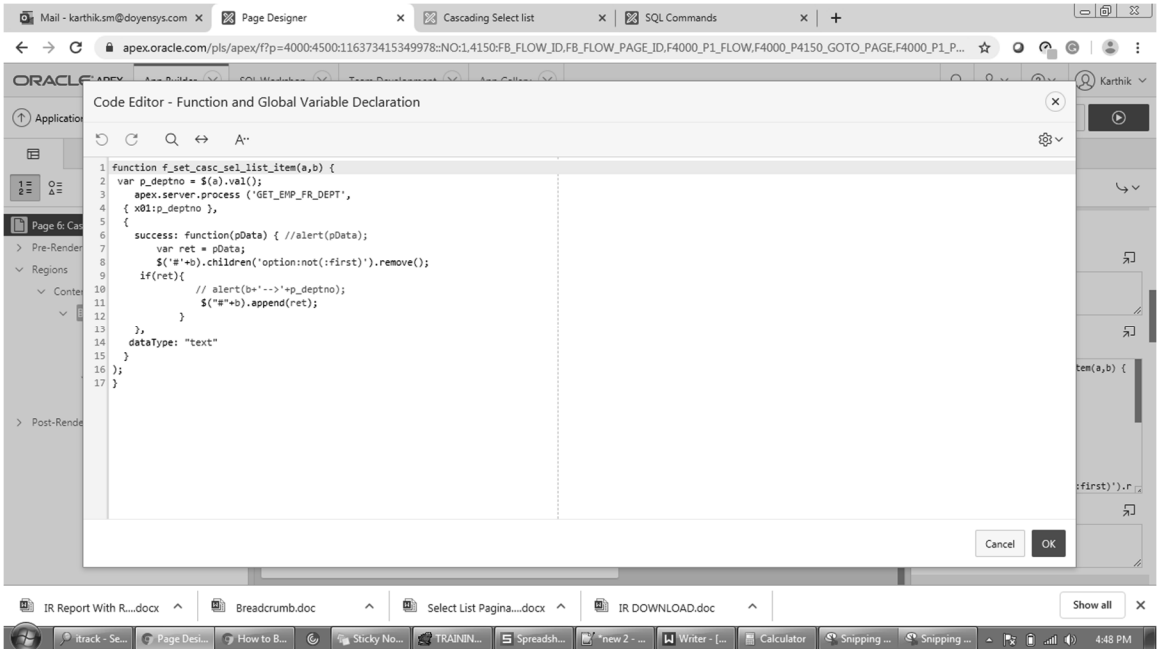


Step 4: Set Escape special characters as **No** for the following Columns CHKBX, DNAME, ENAME, SAL



Step 5: Create a button named as SAVE,**Step 6: Place the below javascript code in the Function and Global Variable Declaration section of Page header**

```
function f_set_casc_sel_list_item(a, b) {
    var p_deptno = $(a).val();
    apex.server.process('GET_EMP_FR_DEPT', {
        x01: p_deptno
    }, {
        success: function(pData) { //alert(pData);
            var ret = pData;
            $('#'+ b).children('option:not(:first)').remove();
            if (ret) {
                $('#'+ b).append(ret);
            }
        },
        dataType: "text"
    });
}
```



Step 7: Create an *Ajax call back* application level process and enter the process name as *GET_EMP_FR_DEPT* and place the code given below,

Shared Components ==> Application Process

BEGIN

```
OWA_UTIL.mime_header ('text/xml', FALSE);
HTP.p ('Cache-Control: no-cache');
HTP.p ('Pragma: no-cache');
OWA_UTIL.http_header_close;
```

```
FOR rec IN (SELECT ename d, empno r
            FROM emp
            WHERE deptno = apex_application.g_x01)
```

LOOP

```
    HTP.pn ('<option value="" || rec.r || ">" || rec.d ||
'</option>');
END LOOP;
```

END;

ORACLE APEX

Application Process

Application: 135252 Sample App

* Sequence: 1

* Process Point: Ajax Callback: Run this application process when requested by a page process.

* Name: GET_EMP_FR_DEPT

Type: PL/SQL Anonymous Block

Source

* PL/SQL Code

```

1 BEGIN
2   OWA_UTIL.mime_header ('text/xml', FALSE);
3   HTP.p ('cache-control: no-cache');
4   HTP.p ('Pragma: no-cache');
5   OWA_UTIL.http_header_close;
6
7   FOR rec IN (SELECT ename d, empno r
8               FROM emp
9               WHERE deptno = apex_application.g_x01)
10  LOOP
11    HTP.pn ('<option value="' || rec.r || '" || '</options>');
12  END LOOP;
13
14 END;
```

Application Processes can be used to perform actions required for every page. Application and page items can be referenced using :ITEM_NAME (bind variable) syntax. The Process Text is treated as a PL/SQL anonymous block. Application Express will wrap a "begin" and "end" around this code. If you need to declare local variables you will need to do so using a "declare", "begin" and "end" block.

Return to page

Step 8: Create a Dynamic action on the target page

- ☐ Event :- *Change*
- ☐ Selection Type :- *jQuery Selector*
- ☐ jQuery Selector :- *[name=f01]*
- ☐ Action :- *Execute Javascript Code*
- ☐ Code :-

```

var column_id = this.triggeringElement.id;
var rw_id = column_id.substr(4);
if ($('#'+column_id).is(":checked")) {
    $('#f02_'+rw_id).val('Y');
} else {
    $('#f02_'+rw_id).val('N');
}
```

APEX PAGE COMPONENTS

The image displays two screenshots of the Oracle APEX Page Designer interface, illustrating the configuration of a dynamic action for an 'On change' event.

Top Screenshot: The 'Dynamic Action' configuration panel is shown. The 'Identification' section has 'Name' set to 'On change f01'. The 'Execution Options' section has 'Sequence' set to '10'. The 'When' section has 'Event' set to 'Change', 'Selection Type' set to 'jQuery Selector', and 'jQuery Selector' set to '[name=f01]'. The 'Client-side Condition' section has 'Type' set to '- Select -'. The 'Advanced' section has 'Event Scope' set to 'Static'.

Bottom Screenshot: The 'Dynamic Action' configuration panel is shown with the 'Action' section expanded. The 'Action' is set to 'Execute JavaScript Code'. The 'Settings' section has 'Code' set to the following JavaScript code:

```
var column_id = this.triggeringElement.id;
var row_id = column_id.substr(0, 4);
if ($('#'+column_id).is(':checked')) {
    $('#f02-'+row_id).val('Yes');
} else {
    $('#f02-'+row_id).val('No');
}
```

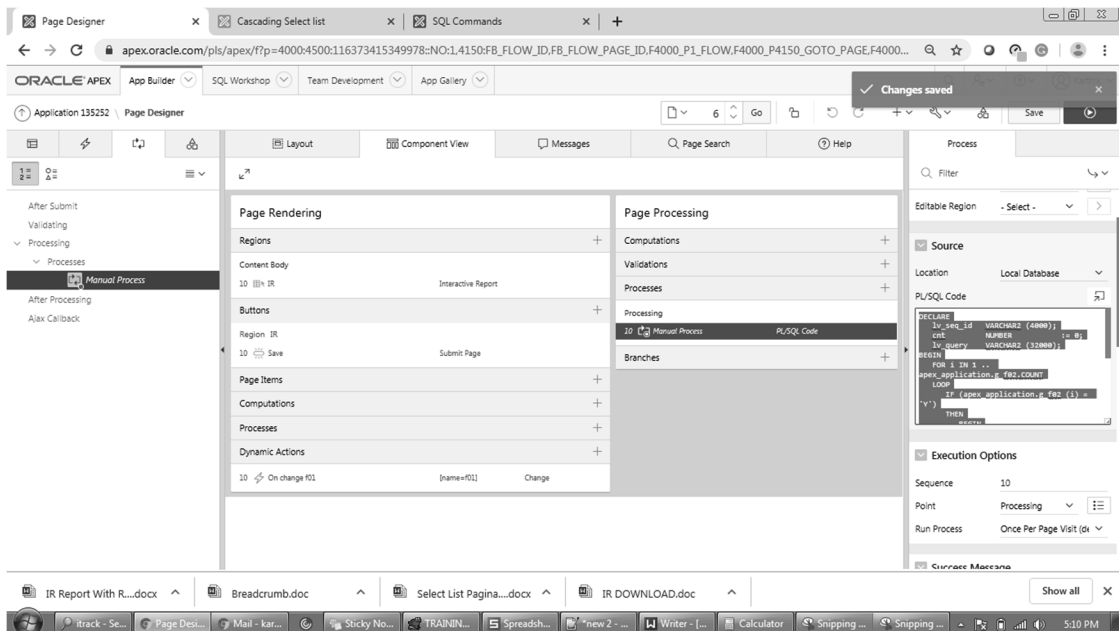
The 'Affected Elements' section has 'Selection Type' set to '- Select -'. The 'Execution Options' section has 'Sequence' set to '10' and 'Event' set to 'On change f01'.

ORACLE APEX

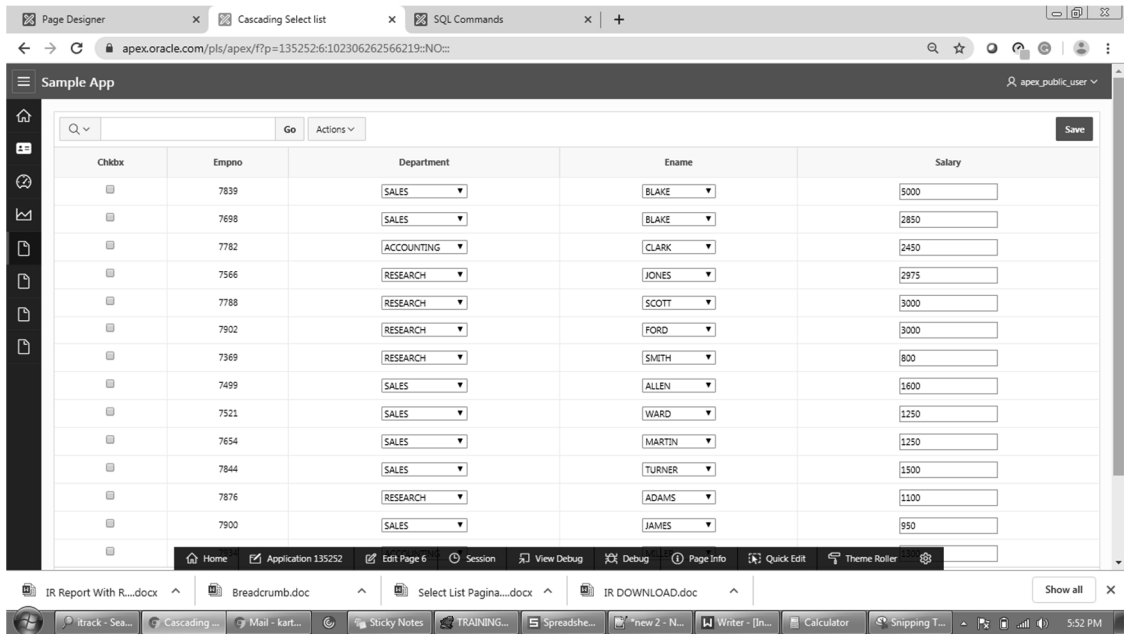
Step 9: Create a page process on the target page.

Code:

```
DECLARE
lv_seq_id   VARCHAR2 (4000);
cnt         NUMBER           := 0;
lv_query    VARCHAR2 (32000);
BEGIN
FOR i IN 1 .. apex_application.g_f02.COUNTLOOP
  IF (apex_application.g_f02 (i) = 'Y') THEN
    BEGIN
      UPDATE emp
      SET ename = apex_application.g_f11 (i),
          deptno = apex_application.g_f10 (i),
          sal = apex_application.g_f12 (i)
      WHERE empno = apex_application.g_f01 (i);
    END;
  END IF;
END LOOP;
END;
```



Step 10: Finally screen design will look like the below image and now you can see the Editable IR report with cascading select list and do the changes and store the changes in the database.



Enable/Disable IR Download button based on User Role

By default IR region allows to either enable Download button to all users or disable Download button to all users. Suppose there is a requirement to enable/disable download button based on user role, then we need to go for customization.

Let's see how to achieve from the following steps.

Step 1: Create an Authorization scheme to validate the user role returning Boolean type.

Shared Components → Security → Authorization Schemes → Create

Name : GET_USER_ROLE

Type : PL/SQL Function Returning Boolean.

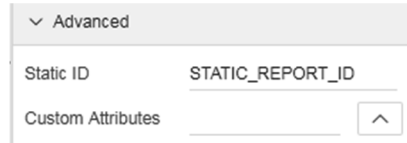
Code:

```
CREATE OR REPLACE FUNCTION HAS_ROLES (PROLE IN VARCHAR2) RETURN BOOLEAN
IS
    ROLE_DETAILS    ROLES%ROWTYPE;
BEGIN
    SELECT *
        INTO ROLE_DETAILS
        FROM ROLES
        WHERE ROLE = PROLE;
    IF ROLE_DETAILS.ROLE = 'ADMIN' THEN
        RETURN TRUE;
    ELSE
```

ORACLE APEX

```
RETURN FALSE;  
END IF;  
END;
```

Step 2: Assign a Static ID to interactive report.

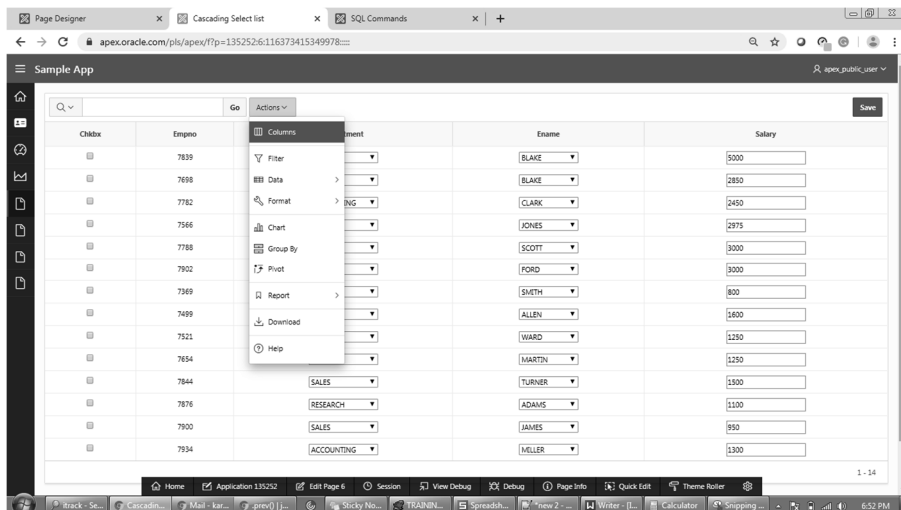


Step 3: Create a Dynamic action.

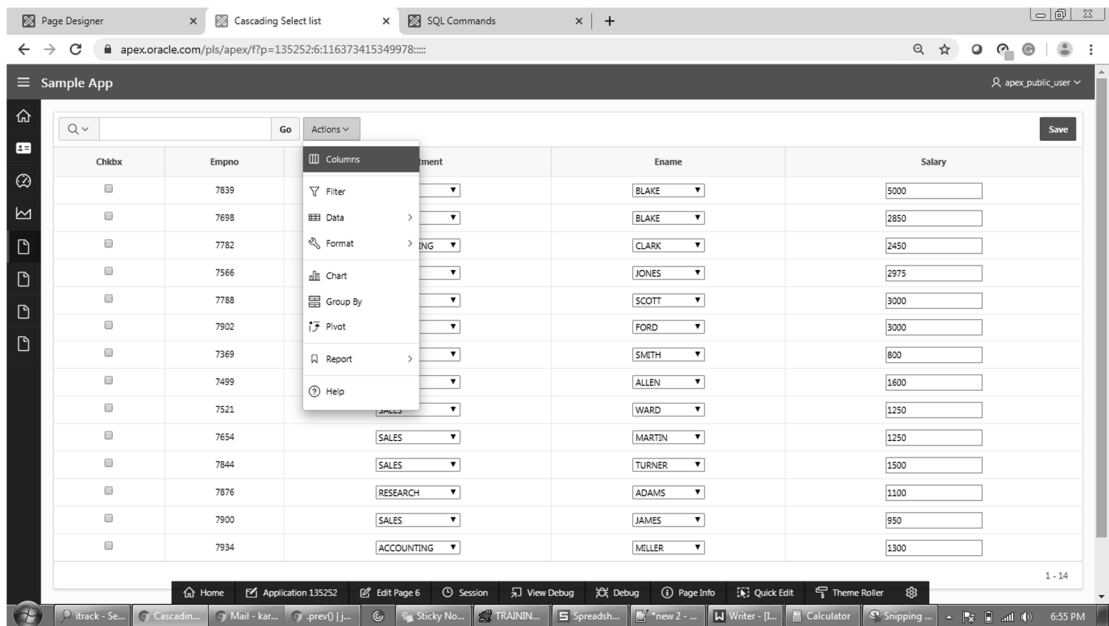
- * Event :- *Click*
- * Selection Type :- *jQuery Selector*
- * Query Selector :- *#STATIC_REPORT_ID_actions_button*
- * Security → Authorization Scheme :- *GET_USER_ROLE*
- * Action :- *Execute Javascript Code*
- * Code :-

```
setTimeout(function() {  
    $( ".icon-irr-download"  
    ).parent().parent().parent().parent().prev().remove();  
    $( ".icon-irr-download"  
    ).parent().parent().parent().parent().remove();  
}, 100);
```

Step 4: Screen looks like below image when Admin user logs into the application.



Steps 5: Screen looks like below image when Support user logs into the application.



List

A list template provides a collection of links which can be used for navigation and other action-oriented tasks. You can optionally show badges, icons, sub list items, and more. A list will be available under Navigation region on the Shared Components, and it can be created globally and associated with any of the pages within the application. The appearance of the list display in the page is based on the list templates. Each list can be controlled by adding a display condition to it. Let's see how to create a list,

Two types of lists can be created in Oracle Application Express:

- ☐ Static Lists
- ☐ Dynamic Lists

Static Lists

A Static List is based on predefined display and return values. When you wish to create a static list, a list entry label and a target (either a page or URL) attributes need to be defined. A static list can be created using the following option, creating from scratch, by copying

existing entries, or by adding the list entries. Each list will have display condition, which enables to control its display.

The following describes how to create a Static List:

- ☐ Navigate to the application shared components.
- ☐ Create the list using the Create List Wizard.
- ☐ Mention whether to create the list from scratch or by copying an existing list.
- ☐ If the option from scratch is chosen, then enter the details for the Name, Select type as Static and click next.
- ☐ Define list entries and click next.
- ☐ Specify the details for the following and click Create list.
 - Create List regions
 - Region Position
 - Region Template
 - List template
- ☐ Create a region on a page and change the region type to List.
- ☐ Navigate to source and click on the drop down of the list. You can view the collections of lists that are created on the shared component.
- ☐ Choose any of the ones in the drop down.
- ☐ Navigate to Appearance section available on the list attribute.
- ☐ If you see in the List template, by default links list will be selected. But we can also change the template by choosing any of the below,
 - Badge list
 - Cards
 - Links List
 - Media List
 - Menu Bar
 - Menu Pop up
 - Navigation Bar
 - Side Navigation Menu
 - Tabs
 - Top Navigation Menu

- Top Navigation Tabs
- Wizard Progress

Dynamic List

A Dynamic list is based on an SQL query or a PL/SQL function executed at runtime. Using Dynamic list, a list can be created dynamically which will also support for mobile frameworks.

The list definition displays a specific type of page item, such as progress bars, sidebar, bullet navigation list, or navigation menu. You can control how a list displays through templates.

The following describes the process to create Dynamic List:

- ☐ Navigate to the application shared components.
- ☐ Create the list by using the Create List Wizard.
- ☐ Mention whether to create the list from scratch or by copying an existing list.
- ☐ If the option "from scratch" is chosen, then enter the details for the Name, Select type as Dynamic and click next.
- ☐ Enter an SQL query or a PL/SQL function returning an SQL query.
- ☐ Click Create button to create a dynamic list.
- ☐ Navigate to the target page and create a region and change the region type to List.
- ☐ Navigate to source and click on the drop down of the list. You can view the collections of lists that are created on the shared component.
- ☐ Choose any of the ones in the drop down.
- ☐ Navigate to Appearance section available on the list attribute.
- ☐ If you see in the List template, by default links list will be selected. But we can also change the template by choosing any of the below,
 - Badge list
 - Cards
 - Links List
 - Media List
 - Menu Bar
 - Menu Pop up

- Navigation Bar
- Side Navigation Menu
- Tabs
- Top Navigation Menu
- Top Navigation Tabs
- Wizard Progress

List View

List view is optimized for data display and smart phone navigation. List view on a page contains the formatted result of an SQL query. You provide a custom SQL SELECT statement or a return SQL Query PL / SQL function.

Create a simple list view report on a page by the following steps,

- ☐ Create a page in the application.
- ☐ Navigate to the created page and create a region using tree view / component view.
- ☐ Select region type as List view.
- ☐ Navigate to Source section and select the source type as Table/ View, SQL Query or PL/SQL Function Body returning SQL query. I have chosen SQL Query.
- ☐ Place the following code, *SELECT * FROM DEPT*
- ☐ Navigate to setting attribute which is available in the region attributes.
- ☐ Select the column value for Text Column.(Here DNAME was selected.)
- ☐ Click Save and Run a Page.

The report looks like the below image. It will display the value based on the selection done in the Text column.

DEPT
ACCOUNTING
RESEARCH
SALES
OPERATIONS

Oracle Apex allows us to make more informative list view by adding simple HTML code. Let's see the example of creating an informative list and based on the value selected in the list, the classic report should get refreshed and display the result. Follow the steps given below,

- ☐ Create a Region using tree view/component view.
- ☐ Change the region type to List View.
- ☐ Select the source type as Table/View and choose DEPT table.
- ☐ Navigate to setting attribute available in the list attribute and choose "Advanced Formatting".
- ☐ Add the below piece of code in the Text Formatting,

```
<div id="&DEPTNO." class="cList-dept">
    <h4>&DNAME.</h4>
    <span>&LOC.</span>
</div>
```

Go to page header and navigate to the inline section and place the code given below,

```
.cSelected{
    background-color:#0076df;
    color : white
;}
.cList-dept{
    cursor:pointer;
}
```

- ☐ Create the hidden item in the name of PX_DEPT.
- ☐ Create a Region and change the type as Classic Report.
- ☐ Change the source type as SQL Query.
- ☐ Place the below piece of code in the SQL Query section,

```
SELECT e.empno,e.ename,e.job,e.sal,e.mgr,e.hiredate,d.dname
FROM emp e, dept d
where e.deptno= d.deptno
and d.deptno = :p23_dept
ORDER by ename
```

- ☐ Navigate to Layout section and select "Start New Row" as **No**.
- ☐ Create a Dynamic action as follows,
 - Event : **Click**

- Selection Type : jQuery Selector
- jQuery Selector : .cList-dept
- Create three True Actions as follows,
 - True Action 1
 - Action : *Execute Javascript code*
 - Code :


```
var dept_id = $('#P23_DEPT').val();
if (dept_id != "") {
    $('#'+ dept_id).removeClass("cSelected");
}
apex.item("P23_DEPT").setValue(this.triggeringElement.id);
$('#'+
this.triggeringElement.id).addClass("cSelected");
```
 - True Action 2
 - Action : *Execute Javascript code*
 - PL/SQLCode : *Begin Null; End;*
 - Items to Submit : P23_DEPT(Hidden field)
 - True Action 3
 - Action : *Refresh*
 - Selection Type : *Region*
 - Region : EMP(Classic Report)
- Click Save and Run a page.

The Final screen will look like an image as shown below. Here in this example the Department **Research** is selected in the list view and the classic EMP displays the employee data for the selected department.

List View

ACCOUNTING

NEW YORK

RESEARCH

DALLAS

SALES

CHICAGO

OPERATIONS

BOSTON

EMP

Empno	Ename	Job	Mgr	Hiredate	Sal	Dname
7876	ADAMS	CLERK	7788	23-MAY-87	1100	RESEARCH
7902	FORD	ANALYST	7566	03-DEC-81	3000	RESEARCH
7566	JONES	MANAGER	7839	02-APR-81	2975	RESEARCH
7788	SCOTT	ANALYST	7566	19-APR-87	3000	RESEARCH
7369	SMITH	CLERK	7902	17-DEC-80	800	RESEARCH

1 - 5

Source : <https://krutten.blogspot.com/2018/06/46-list-view-in-apex-181.html>

Map Chart

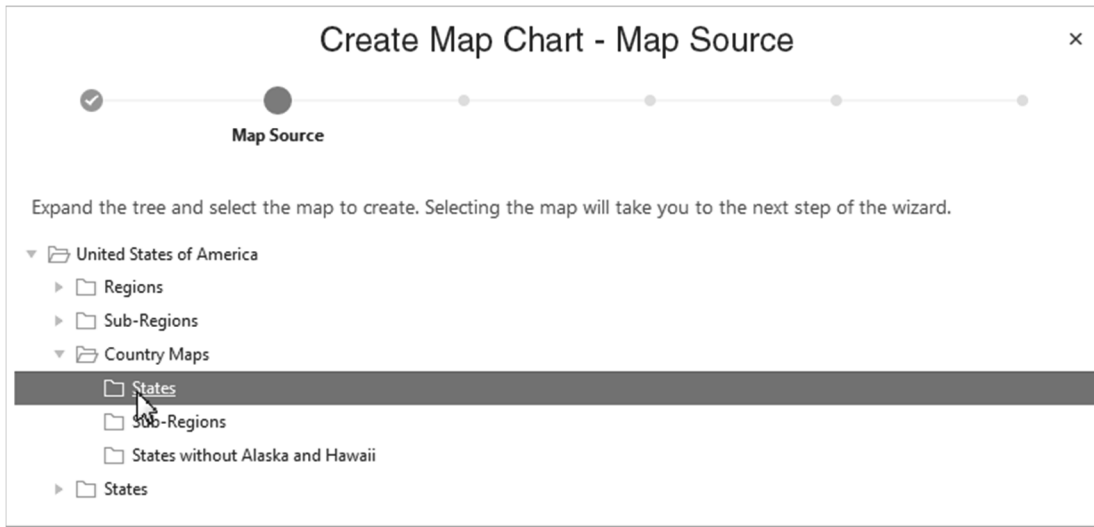
A **chart** is a graphical representation of data, in which the data is represented by symbols, such as bars in a bar chart, lines in a line chart, or slices in a Pie chart.

A chart can also represent tabular numeric data, functions, or some kinds of qualitative structure, and provides different information.

- ☐ On the Workspace home page, click the App Builder icon.
- ☐ Select an application.
- ☐ Click Create Page.
- ☐ Click Chart.
- ☐ Select Map Chart.
- ☐ For Map Type, select United States of America and click next.
- ☐ For Map Source, expand Country Maps and select States.

Image Reference: Oracle Docs

- For Page Attributes specify the appropriate attributes and click Next.
- ☐ For Navigation Preference, specify the type of navigation to include on this page and click Next. The navigation options (for example, navigation menu or tabs) depends upon the current application theme.
- ☐ For Map Attributes, select the appropriate attributes and click Next.
- ☐ The Query page appears.



- Scroll down and expand and review the Map Query Example and Map Reference Information regions.
- Scroll down and expand and review the Map Reference Information region at the bottom of the page.

Image Reference: Oracle Docs

Enter SQL Query or PL/SQL function returning a SQL Query:

```
FROM (
  SELECT 'Ireland' country, 4203200 people
  FROM dual
)
```

Build Query

☒ Perform query validation ☐ Save query without validation

Page Items to Submit:

Maximum Rows:

When No Data Found Message:

> Map Query Example

▼ Map Reference Information

Region Id	Region Name	Centroid X	Centroid Y	State Fips	Sub Region	Gmi Admin
AK	Alaska	-151.777777777778	65.676268861454	2	Pacific	

< Cancel **Create**

Sample Query

```
SELECT null link, region label, value Populations
FROM (
  SELECT 'Florida' region, 18328340 value FROM dual
  UNION ALL
  SELECT 'Alaska' region, 686293 value FROM dual)
```

Map Chart to a New Page

- ☐ On the Workspace home page, click the App Builder icon.
- ☐ Select an application.
- ☐ Click Create Page.
- ☐ The Create Page Wizard appears.
- ☐ For Create a Page:
- ☐ User Interface - Select a user interface for the page.
- ☐ This attribute only displays for applications using older themes and for which Desktop and Mobile User Interfaces have been defined.
- ☐ Select a page type - Select Chart.
- ☐ On Create Chart, select Map Chart.
- ☐ On Create Page, select a map type (for example, Europe), and click Next.
- ☐ On Create Map Chart, expand the tree and select the map to create.
- ☐ For Page Attributes:
- ☐ Page Number - Select a page in which the chart object is to appear.
- ☐ Page Name - If creating a new page, enter a page name.
- ☐ Page Mode - Select a page mode.
- ☐ Breadcrumb - Select whether you want to use a breadcrumb navigation control on your page, and which breadcrumb navigation control you want to use
- ☐ Click Next.
- ☐ On Navigation Menu, specify a Navigation Preference to include on this page and click Next. The navigation options (for example, navigation menu or tabs) depend upon the current application theme.
- ☐ For Map Attributes, select the appropriate attributes and click Next.
- ☐ Entering an SQL query in the field provided.

- ☐ Clicking the Build Query button. When the Build Chart Query Wizard appears, follow the on-screen instructions and click create.

Map Chart in Page Designer

- ☐ On the Workspace home page, click the App Builder icon.
- ☐ Select an application.
- ☐ Select a page.
- ☐ Title - Enter a title for the region.

Layout

- ☐ Sequence - Enter the display sequence for this item.
- ☐ Parent Region - Select the parent region to which this region belongs.
- ☐ Position - Select the template position used to display this region.

Appearance

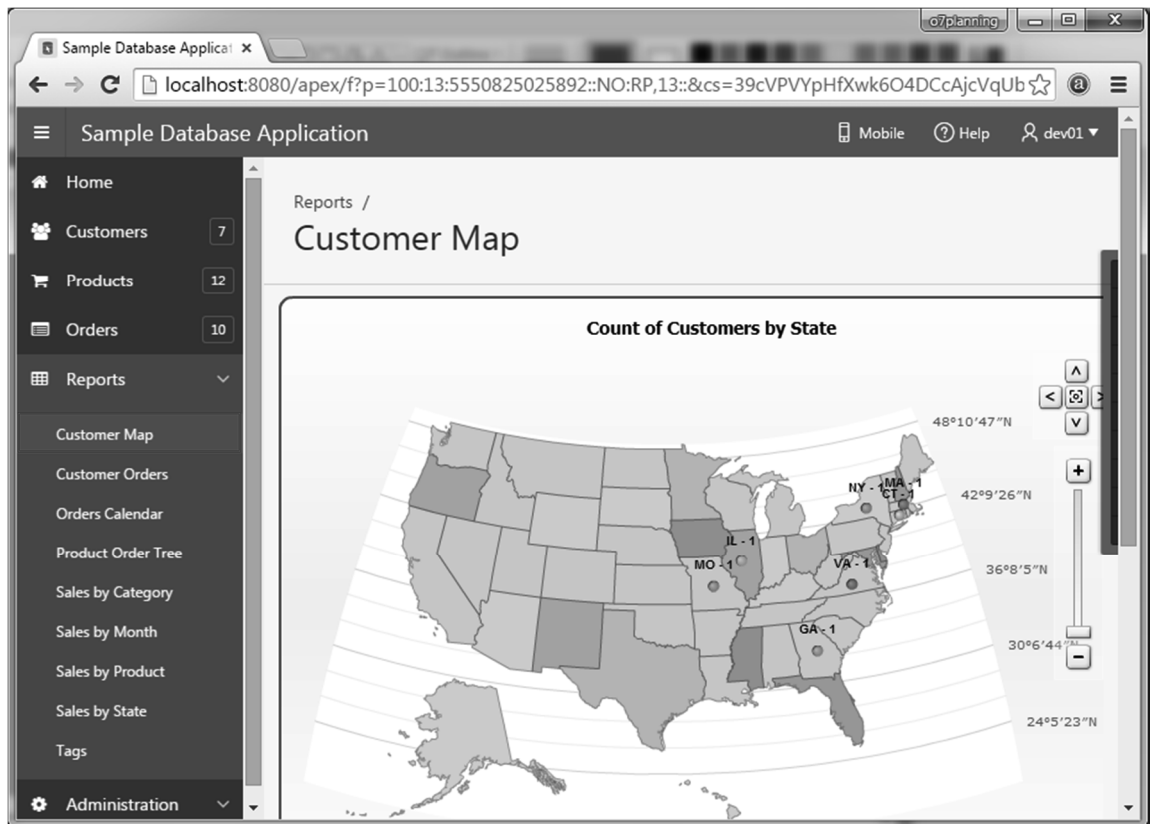
- ☐ Template - Select a region template.
- ☐ Item Display Position - Specify where page items display in relation to the main region content.
- ☐ Layout - Specify the Sequence, Parent Region, and Position.
- ☐ Appearance - Specify the template. In the Rendering tab, under the map chart title, select the Attributes node.
- ☐ Map, Level 1 - Select the map level from the available list (for example Europe).
- ☐ Map, Level 2 - Select the map level from the available list (for example, Denmark).
- ☐ Source, SQL Query - Enter the SQL Query for the chart series and click save.

Custom XML with Map Charts

- ☐ View the page in Page Designer.
- ☐ On the Workspace home page, click the App Builder icon.
- ☐ Select an application.
- ☐ Select a page.
- ☐ Page Designer appears.
- ☐ In the Rendering tab, under the region title, select the Attributes node.
- ☐ The map chart attributes display in the Property Editor.

- ☐ Locate and expand the Custom XML.
- ☐ For Custom XML, Custom, select Yes.
- ☐ Edit the XML.
- ☐ Click Save.

Result



Plsql Dynamic Content

PL/SQL Dynamic Content region based on PL/SQL enables you to render any HTML or text using the PL/SQL Web utility Packages.

Step 1: Create a Blank Page

Create a Page

Page Type

Component

Feature

?

Blank Page

Report

Form

Master Detail

Plug-ins

Chart

Dashboard

Calendar

Tree

Wizard

Data Loading

Legacy Page

Cancel

Help

Next >

Create a Blank Page

Page Attributes

* Page Number

2

?

* Name

Dynamic Plsql Content

?

* Page Mode

Normal

Modal Dialog

Non-Modal Dialog

?

Page Group

- Select Page Group -

?

Breadcrumb

Breadcrumb

?

Parent Entry

No parent entry

?

Entry Name

Dynamic Plsql Content

?

> Optional Static Content Regions

<

Cancel

Next >

✓

Navigation Menu

Navigation Preference

☐ Do not associate this page with a navigation menu entry ?

☒ **Create a new navigation menu entry**

☐ Identify an existing navigation menu entry for this page

* New Navigation Menu Entry

Dynamic Plsql Content

?

Parent Navigation Menu Entry

- No parent selected - Home ?

< Cancel

Next >

Create a Blank Page

×

✓

✓

Confirm

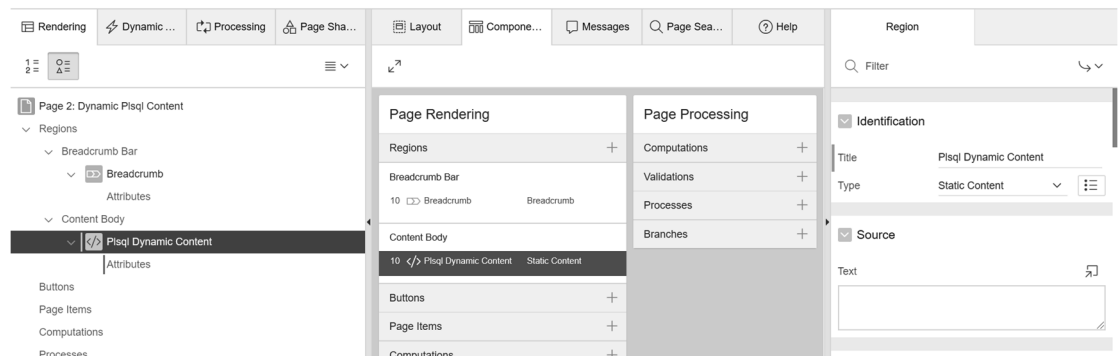
You have requested to create a page with the following attributes. Please confirm your selections.

Application	105
Page	2
Page Name	Dynamic Plsql Content
Page Title	Dynamic Plsql Content

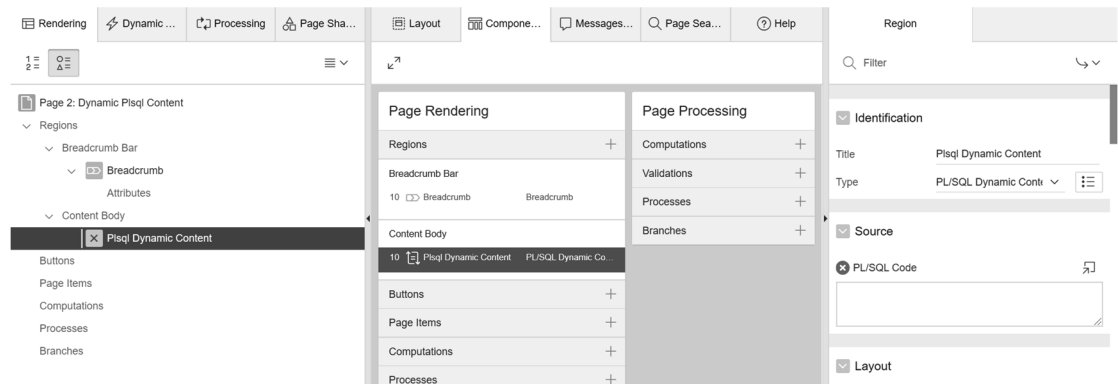
< Cancel

Finish

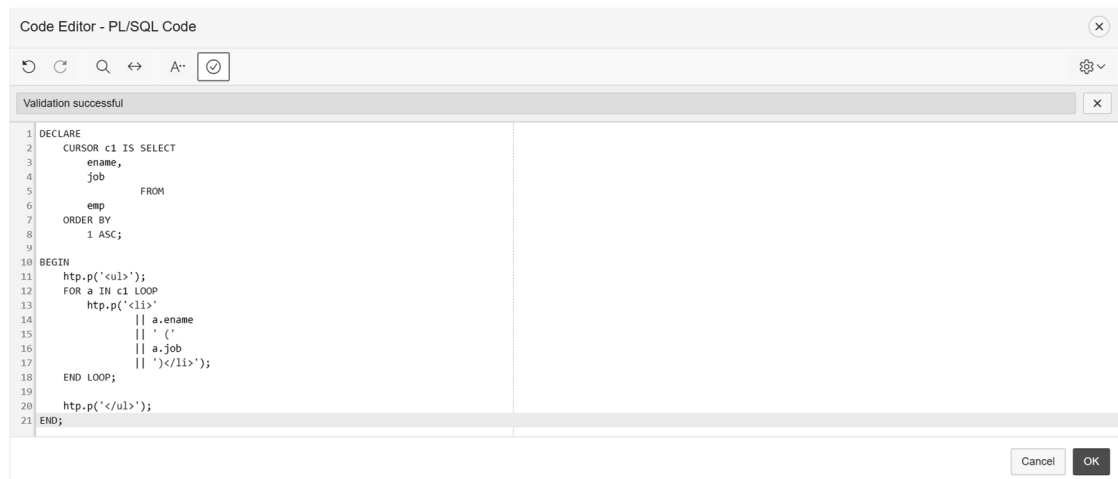
Step 2: create a static content Region



Step 3: Assign the Region type as Plsql Dynamic Content



Step 4: Enter the Plsql Code in the Code Editor Window

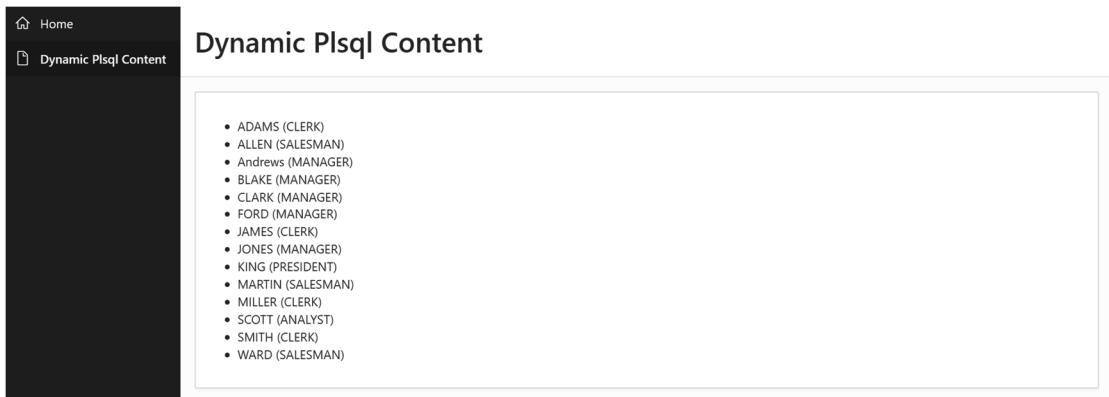


Sample Query

```

DECLARE
    CURSOR c1 IS
        SELECT ename,
               job
        FROM emp
        ORDER BY 1 ASC;
BEGIN
    http.P('<ul>');
    FOR a IN c1 LOOP
        http.P('<li>'
              || a.ename
              || ' ('
              || a.job
              || ')</li>');
    END LOOP;
    http.P('</ul>');
END;
```

Result



The screenshot shows an APEX page with a dark sidebar on the left containing two links: 'Home' and 'Dynamic Plsql Content'. The main content area has a title 'Dynamic Plsql Content' and a list of 14 items, each consisting of an employee name followed by their job in parentheses, all enclosed in bullet points.

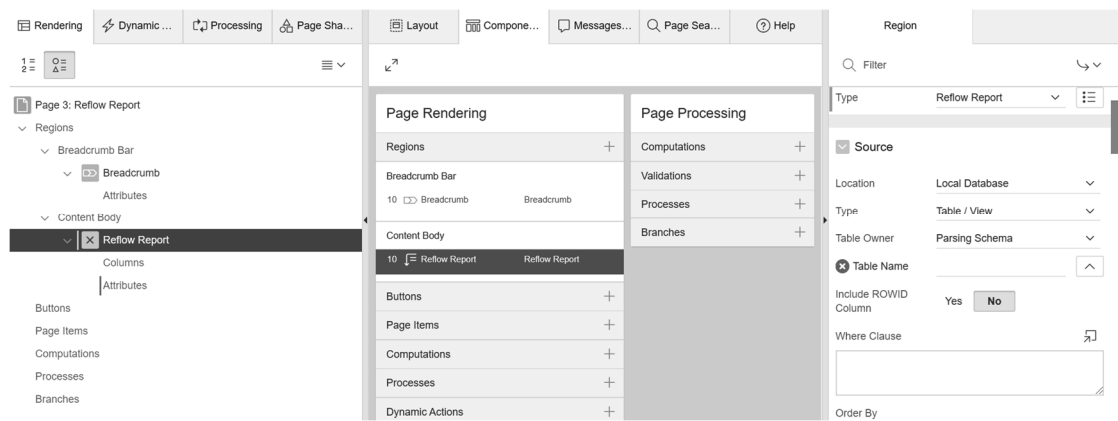
- ADAMS (CLERK)
- ALLEN (SALESMAN)
- Andrews (MANAGER)
- BLAKE (MANAGER)
- CLARK (MANAGER)
- FORD (MANAGER)
- JAMES (CLERK)
- JONES (MANAGER)
- KING (PRESIDENT)
- MARTIN (SALESMAN)
- MILLER (CLERK)
- SCOTT (ANALYST)
- SMITH (CLERK)
- WARD (SALESMAN)

Reflow Report

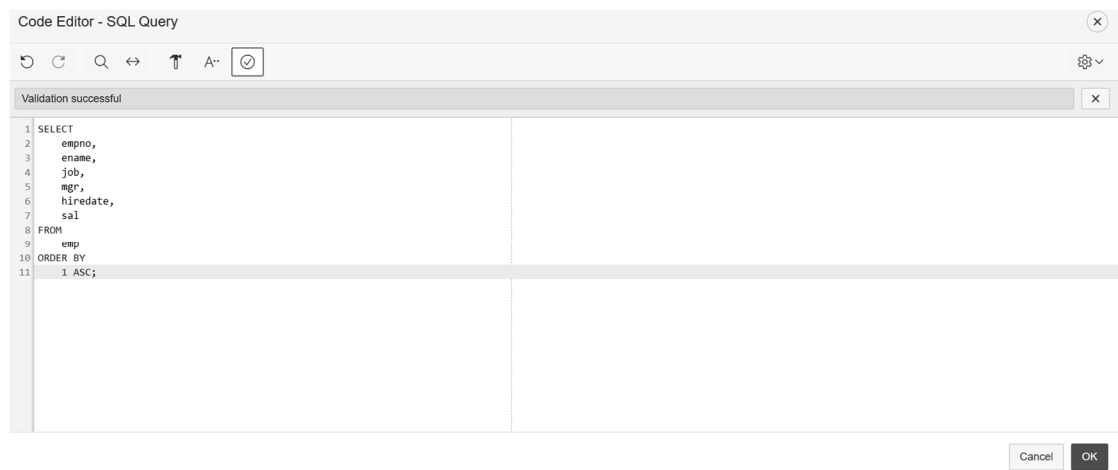
The Reflow Report is a responsive table-based report where on small screens and also the table columns become stacked and each row becomes a block of data

It is an alternative display to the Column Toggle Report and will also be useful when you want to display all columns of a tabular report in a way that works well for small screens.

Step 1: create a Region and set the type as Reflow Report.



Step 2: Select the table or Enter the sql query in the Code editor



Sample Query

```
SELECT empno, ename, job, mgr,hiredate, sal
FROM emp
ORDER BY 1 ASC;
```

Result

With Stripes

Reflow Report

Empno	Ename	Job	Mgr	Hiredate	Sal
7369	SMITH	CLERK	7902	17-DEC-80	800
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600
7521	WARD	SALESMAN	7698	22-FEB-81	1250
7566	JONES	MANAGER	7839	02-APR-81	2975
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250
7698	BLAKE	MANAGER	7839	01-MAY-81	2850
7782	CLARK	MANAGER	7839	09-JUN-81	2450
7788	SCOTT	ANALYST	7566	19-APR-87	3000
7839	KING	PRESIDENT		17-NOV-81	5000
7844	Andrews	MANAGER	7698	08-SEP-81	1500
7876	ADAMS	CLERK	7788	23-MAY-87	1100
7900	JAMES	CLERK	7698	03-DEC-81	950
7902	FORD	MANAGER	7566	03-DEC-81	3000
7934	MILLER	CLERK	7782	23-JAN-82	1300

Without Stripes

Reflow Report

Empno	Ename	Job	Mgr	Hiredate	Sal
7369	SMITH	CLERK	7902	17-DEC-80	800
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600
7521	WARD	SALESMAN	7698	22-FEB-81	1250
7566	JONES	MANAGER	7839	02-APR-81	2975
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250
7698	BLAKE	MANAGER	7839	01-MAY-81	2850
7782	CLARK	MANAGER	7839	09-JUN-81	2450
7788	SCOTT	ANALYST	7566	19-APR-87	3000
7839	KING	PRESIDENT		17-NOV-81	5000
7844	Andrews	MANAGER	7698	08-SEP-81	1500
7876	ADAMS	CLERK	7788	23-MAY-87	1100
7900	JAMES	CLERK	7698	03-DEC-81	950
7902	FORD	MANAGER	7566	03-DEC-81	3000
7934	MILLER	CLERK	7782	23-JAN-82	1300

Region Display Selector

The Region Display Selector is a Region component that provides a page level navigation control for other regions on the with the Region Display Selector property set to Yes. It can be configured to work in two modes:

View Single Region

Selecting a tab will make the corresponding region visible and hide the other selections.

Scroll Window

It always displays all the regions on the page. Selecting a tab will scroll your window to the corresponding region.

Step 1: Create a Region Display Selector region and place it in the Breadcrumb region position on your page.

Step 2: Modify the other regions on the page and set the Region Display Selector property to Yes for the regions you want to appear in the Region Display Selector.

Result

Tab 1Tab 2Tab 3

Tab 1

Content 1

You can easily nest tabs by creating another Tabs Container region within a sub region of an existing Tabs Container region.

Tab 1Tab 2Tab 3Tab 4 - Nested

Tab 1

Content 1

Tab 1Tab 2Tab 3Tab 4 - Nested

Nested Region ANested Region B

Nested Region A

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Dolore accusamus quidem saepe in voluptate ad ea quis maxime consectetur voluptatum eos, laboriosam voluptates cupiditate atque obcaecati numquam fuga, explicabo illo.

104

Tab 1

Tab 2

Tab 3

Tab 4 - Nested

Nested Region A

Nested Region B

Nested Region B

Lorem ipsum dolor sit amet, consectetur adipisicing elit. Similique ducimus, rerum illo ea provident a. Veniam voluptates sed quidem incididunt, dolores vel eveniet. Vel officiis facere, totam fuga voluptatum obcaecati!

Static Content

Static Content is like a plain canvas. It is up to the developer to decide what he/she has to do with the region. In majority of the cases it is used to display some html or static text like notes, or instructions to the end users can be displayed here.

In Doyensys we have come across the requirement where one of our customers asked for dynamic html table with mergeable text area. We have achieved this with static content region

Edit	Risk Display Order	Risk Description	Client Facing Risk	Error Display Order	Error Type	Impact Score	Tolerance level
✕ Edit	1	Risk1002	☒	1	Error1	LOW	9 Errors
✕ Edit	2	Risk 23	☐	1	Risk 23	LOW	5 Errors
				2	ERR1%	MEDIUM	1 Error

We have done the following changes to fulfil the above requirement. We have written HTML in the static content, we have coded the following in the static content

```
<style>
.t-Report-colHead
{
  background-color:#128475;
  color: #FFFFFF;}
</style>
<table id="myhtml" class="t-Report-report" style="width:100%">
  <tr>
    <th class="t-Report-colHead" style="border-left: 1px solid #00433c" id="EDIT">Edit</th>
    <th class="t-Report-colHead hiddenth" id="RISK_ID" style="width: 65px;">Risk<br>ID</th>
    <th class="t-Report-colHead" id="RISK_ORD" style="width: 65px">Risk<br>Display<br>Order</th>
```

```

        <th class="t-Report-colHead" id="RISK_ORD_HID" style="display:
none">Risk Order Hidden</th>
        <th class="t-Report-colHead" id="RISK_DESC">Risk Description</th>
        <th class="t-Report-colHead" id="CF_RISK" style="width:
100px">Client<br>Facing Risk</th>
        <th class="t-Report-colHead" id="ERR_ORD" style="width:
65px">Error<br>Display<br>Order</th>
        <th class="t-Report-colHead" id="ERR_ORD_HID" style="display:
none">Error Order Hidden</th>
        <th class="t-Report-colHead" id="ERROR_DESC">Error Type</th>
        <th class="t-Report-colHead" id="IMPACT" style="width:
100px">Impact <br> Score</th>
        <th class="t-Report-colHead" style="border-right: 1px solid
#00433c" id="TOL" style="width: 115px">Tolerance <br> level</th>
        <th style="border-left: 1px solid #00433c;width: 25px"
id="ERR_DEL"></th>
    </tr>
</table>
<br>
<div>
    <span class="edit" id="insnewrisk" onclick="addrisk()" style="margin-
left: 45px;">Insert New Risk</span>
</div>

```

the following JavaScript function in the global variable declaration

```

setInterval(function() {
    $("#AUTOSAVE").click();
}, 100000);
var max_row = 1;
var curr_td_element;

function createhtml(rownum, risk_arr) {
    var risk_array = risk_arr.split("~`~");
    var risk = risk_array[0];
    var cf_risk = risk_array[1];
    var error = risk_array[2];
    var cf_error = risk_array[3];
    var impact = risk_array[4];
    var tol = risk_array[5];
    var risk_ord = risk_array[6];
    var err_ord = risk_array[7];
    var risk_id = risk_array[8];
    var editfunc = 'edit(this)';

```

```

var imp_options = "<option value=''>---Select---</option><option
value='LOW'>LOW</option><option value='MEDIUM'>MEDIUM</option><option
value='HIGH'>HIGH</option>";
var tol_options = "<option value=''>---Select---</option><option
value='0'>None</option><option value='1'>1 Error</option><option
value='2'>2 Errors</option><option value='3'>3 Errors</option><option
value='4'>4 Errors</option><option value='5'>5 Errors</option><option
value='6'>6 Errors</option><option value='7'>7 Errors</option><option
value='8'>8 Errors</option><option value='9'>9 Errors</option><option
value='10'>10 Errors</option>";
var imp_options = "<option value=''>---Select---</option><option
value='LOW'>LOW</option><option value='MEDIUM'>MEDIUM</option><option
value='HIGH'>HIGH</option>";
var cf_risk_prop;
if (impact == 'n') impact = '';
if (tol == '99') tol = '';
if (cf_risk == 'Y') {
    var cf_risk_prop = 'checked';
} else {
    var cf_risk_prop = 'unchecked';
}
var row = "<tr id=row_" + rownum + ">" +
    "<td class='myhtmltd edit' id='f01_" + rownum + "'
headers='EDIT'><span onclick='del(this,1)'><img
src='#APP_IMAGES#delete_icon.jpg'></span><br><span id='edit_" + rownum +
"' onclick='" + "" + editfunc + "" + ">Edit</span></td>" +
    "<td class='myhtmltd1' id='f13_" + rownum + "'
headers='RISK_ID'><input type='text' value='" + risk_id + "'></td>" +
    "<td class='myhtmltd' id='f09_" + rownum + "'
headers='RISK_ORD'><select class='apex-item-select riskord' id=risk_ord_"
+ rownum + "'></select></td>" +
    "<td class='myhtmltd' id='f10_" + rownum + "'
headers='RISK_ORD_HID' style='display: none'><input type='text'
id=risk_ord_hid" + rownum + "' value='" + risk_ord + "'></td>" +
    "<td class='myhtmltd' id='f02_" + rownum + "'
headers='RISK_DESC'><textarea class='myhtmlitem text apex-item-textarea
disable' maxlength='200' type='text'>" + risk + "</textarea></td>" +
    "<td class='myhtmltd' id='f03_" + rownum + "' headers='CF_RISK'
align='center'><input class='myhtmlitem' type='checkbox' " + cf_risk_prop
+ "'></td>" +
    "<td class='myhtmltd' id='f11_" + rownum + "'
headers='ERR_ORD'><select class='apex-item-select errorord' id=err_ord_"
+ rownum + "'></select></td>" +

```

ORACLE APEX

```

        "<td class='myhtmltd' id='f12_" + rownum + "'
headers='ERR_ORD_HID' style='display: none'><input type='text'
id=err_ord_hid" + rownum + " value=" + err_ord + "></td>" +
        "<td class='myhtmltd' id='f04_" + rownum + "'
headers='ERROR_DESC'><textarea class='myhtmlitem text apex-item-textarea
disable' maxlength='200' type='text'>" + error + "</textarea></td>" +
        "<td class='myhtmltd' id='f06_" + rownum + "'
headers='IMPACT'><select onchange='impchg(this)' class='myhtmlitem apex-
item-select disable' id=imp_" + rownum + ">" + imp_options +
"</select></td>" +
        "<td class='myhtmltd' id='f07_" + rownum + "'
headers='TOL'><select onchange='tolchg(this)' class='myhtmlitem apex-
item-select disable' id=tol_" + rownum + ">" + tol_options +
"</select></td>" +
        "<td id='f08_" + rownum + "' headers='ERR_DEL'
onclick='del(this,2)'><img src='#APP_IMAGES#delete_icon.jpg'></td>" +
        "</tr>";

$('#myhtml').append(row);
$("#imp_" + rownum).val(impact);
$("#tol_" + rownum).val(tol);
max_row = rownum;
}

function add_error(addrow) {
    var first_rw_no = $(curr_td_element).parents('tr').index();
    var cond = validation('adderror');
    if (cond) {
        var rowspan = $(curr_td_element).prop("rowspan");
        var row_num = $(addrow).parents('tr').index();
        var row_id = $(addrow).parents('tr').prop('id');

        $('#ADD_ERROR').remove();

        var imp_options = "<option value=''>--Select--</option><option
value='LOW'>LOW</option><option value='MEDIUM'>MEDIUM</option><option
value='HIGH'>HIGH</option>";
        var tol_options = "<option value=''>--Select--</option><option
value='0'>None</option><option value='1'>1 Error</option><option
value='2'>2 Errors</option><option value='3'>3 Errors</option><option
value='4'>4 Errors</option><option value='5'>5 Errors</option><option
value='6'>6 Errors</option><option value='7'>7 Errors</option><option
value='8'>8 Errors</option><option value='9'>9 Errors</option><option
value='10'>10 Errors</option>";
    }
}

```



```

    var add_error = '<div><br><input type="button" class="adderror"
id="ADD_ERROR" onclick="add_error(this)" value="Add Error Type"></div>';

    max_row += 1;
    var new_error = "<tr id=row_" + max_row + ">" +
        "<td class='myhtmltd current' id='f11_" + max_row + "'
headers='ERR_ORD'><select class='apex-item-select errorord' id=err_ord_" +
max_row + "></select></td>" +
        "<td class='myhtmltd current' id='f12_" + max_row + "'
headers='ERR_ORD_HID' style='display: none'><input type='text'
id=err_ord_hid" + max_row + "></td>" +
        "<td class='myhtmltd current' id='f04_" + max_row + "'
headers='ERROR_DESC'><textarea class='myhtmlitem text apex-item-textarea'
maxlength='200' type='text' class='text_field apex-item-
text'></textarea>" + add_error + "</td>" +
        "<td class='myhtmltd current' id='f06_" + max_row + "'
headers='IMPACT'><select onchange='impchg(this)' class='myhtmlitem apex-
item-select' id=imp_" + max_row + ">" + imp_options + "</select></td>" +
        "<td class='myhtmltd current' id='f07_" + max_row + "'
headers='TOL'><select onchange='tolchg(this)' class='myhtmlitem apex-
item-select' id=tol_" + max_row + ">" + tol_options + "</select></td>" +
        "<td id='f08_" + max_row + "' headers='ERR_DEL'
onclick='del(this,2)'><img src='#APP_IMAGES#delete_icon.jpg'></td>" +
        "</tr>";
    $('##' + row_id).after(new_error);

    $('#myhtml > tbody > tr:nth-child(' + parseInt(first_rw_no + 1) +
')').find('td').each(function() {
        var header = $(this).attr('headers');
        if ((header == "EDIT") || (header == "RISK_DESC") || (header
== "RISK_ORD") || (header == "CF_RISK")) {
            $(this).prop('rowspan', parseInt(rowspan + 1));
        }
    });
    add_err_ord(max_row, rowspan, first_rw_no);
}
}

function addrisk() {
    var cond = validation('addrisk');
    if (cond) {
        $('#myhtmlitem').prop('disabled', 'true');
        $('#myhtmlitem').addClass('disable');
        $('#ADD_ERROR').remove();
        var last_row_index = $('#myhtml tr:last').index();

```

```

        max_row += 1;
        var imp_options = "<option value=''>--Select--</option><option
value='LOW'>LOW</option><option value='MEDIUM'>MEDIUM</option><option
value='HIGH'>HIGH</option>";
        var tol_options = "<option value=''>--Select--</option><option
value='0'>None</option><option value='1'>1 Error</option><option
value='2'>2 Errors</option><option value='3'>3 Errors</option><option
value='4'>4 Errors</option><option value='5'>5 Errors</option><option
value='6'>6 Errors</option><option value='7'>7 Errors</option><option
value='8'>8 Errors</option><option value='9'>9 Errors</option><option
value='10'>10 Errors</option>";
        var editfunc = 'edit(this)';
        var add_error = '<div><br><input type="button" class="adderror"
id="ADD_ERROR" onclick="add_error(this)" value="Add Error Type"></div>';
        var err_options = "<option value='1'>1</option>";

        var row = "<tr id=row_" + max_row + ">" +
            "<td class='myhtmltd edit' id='f01_" + max_row + "'
headers='EDIT'><span onclick='del(this,1)'"><img
src='#APP_IMAGES#delete_icon.jpg'"></span><br><span id='edit_" + max_row +
"' onclick=" + "" + editfunc + "" + ">Edit</span></td>" +
            "<td class='myhtmltd1' id='f13_" + max_row + "'
headers='RISK_ID' style='display: none'"><input type='text'"></td>" +
            "<td class='myhtmltd' style='border-left:none' id='f09_" +
max_row + "' headers='RISK_ORD'"><select class='apex-item-select riskord'
id=risk_ord_" + max_row + "></select></td>" +
            "<td class='myhtmltd' id='f10_" + max_row + "'
headers='RISK_ORD_HID' style='display: none'"><input type='text'
id=risk_ord_hid" + max_row + "></td>" +
            "<td class='myhtmltd' id='f02_" + max_row + "'
headers='RISK_DESC'"><textarea class='myhtmlitem text apex-item-textarea'
maxlength='200' type='text' class='text_field apex-item-
text'"></textarea></td>" +
            "<td class='myhtmltd' id='f03_" + max_row + "'
headers='CF_RISK' align='center'"><input class='myhtmlitem'
type='checkbox' unchecked/></td>" +
            "<td class='myhtmltd' id='f11_" + max_row + "'
headers='ERR_ORD'"><select class='apex-item-select errorord' id=err_ord_" +
max_row + ">" + err_options + "</select></td>" +
            "<td class='myhtmltd' id='f12_" + max_row + "'
headers='ERR_ORD_HID' style='display: none'"><input type='text'
id=err_ord_hid" + max_row + "></td>" +
            "<td class='myhtmltd' id='f04_" + max_row + "'
headers='ERROR_DESC'"><textarea class='myhtmlitem text apex-item-textarea'

```

```

maxlength='200' type='text' class='text_field apex-item-
text'></textarea></td>" +
        "<td class='myhtmltd' id='f06_" + max_row + "'
headers='IMPACT'><select onchange='impchg(this)' class='myhtmlitem apex-
item-select' id=imp_" + max_row + ">" + imp_options + "</select></td>" +
        "<td class='myhtmltd' id='f07_" + max_row + "'
headers='TOL'><select onchange='tolchg(this)' class='myhtmlitem apex-
item-select' id=tol_" + max_row + ">" + tol_options + "</select></td>" +
        "<td id='f08_" + max_row + "' headers='ERR_DEL'
onclick='del(this,2)'><img src='#APP_IMAGES#delete_icon.jpg'></td>" +
        "</tr>";

        if ($('#myhtml > tbody > tr').length > 1) {
            var newid = ($('#myhtml tr:last').prop('id'));
            $("##" + newid).after(row);
        } else if ($('#myhtml > tbody > tr').length == 1) {
            $("#myhtml").append(row);
        }

        edit($('#edit_' + max_row), 'new');
    }
    add_del_risk_ord('add', max_row);
}

function del(element, cond) {
    if (cond == 1) {
        $.confirm({
            boxWidth: '30%',
            useBootstrap: false,
            title: 'Confirm!',
            type: 'orange',
            content: 'Confirm Risk Description Deletion',
            buttons: {
                Yes: function() {
                    var first_rw =
$(element).parents('td').parents('tr').index();
                    var row_ord = $("#risk_ord_" + first_rw).val();
                    //first row changes for new added risk
                    if (row_ord == undefined) {
                        row_ord = $("#risk_ord_" + parseInt(first_rw +
1)).val();
                    }
                    var rowspan =
$(element).parents('td').prop('rowspan');
                    if (rowspan == undefined) {

```

```

        rowspan = 1;
    }
    for (i = 1; i <= rowspan; i++) {
        var rownum = parseInt(first_rw + i);
        $('#myhtml > tbody > tr:nth-child(' + rownum + ')
> td ').each(function() {
            $(this).remove();
        });
    }
    add_del_risk_ord('del', 0, row_ord);
    //to remove empty tr (above del function removes only
td)

    $('#myhtml').find('tr').each(function() {
        var len = $(this).html().length;
        if (len == 0) {
            $(this).remove();
        }
    });
},
No: function() {}
}
});
} else {
    $.confirm({
        boxWidth: '30%',
        useBootstrap: false,
        title: 'Confirm!',
        type: 'orange',
        content: 'Confirm Error Type Deletion',
        buttons: {
            Yes: function() {
                var first_rw = $(element).parents('tr').index();
                var rowid = $(element).parents('tr').attr('id');
                var status = false;
                var rowspan;
                var row_ord = $("#risk_ord_" + first_rw).val();
                $('#myhtml > tbody > tr:nth-child(' +
parseInt(first_rw + 1) + ') > td').each(function() {
                    var header = $(this).attr("headers");
                    if ((header == 'RISK_DESC') || (header ==
'RISK_ORD')) {
                        status = true;
                        rowspan = $(this).prop("rowspan");
                        if (rowspan == undefined) {
                            rowspan = 1;

```

```

    }
  }
});
if (status) {
  if (rowspan == 1) {
    //for only one error type, risk must be
removed
    $(element).parents('tr').remove();
    add_del_risk_ord('del', 0, row_ord);
  } else {
    //for first error type of a risk, risk must
be removed and merged with other error type
    var element1;
    var element2;
    var element3;
    var element4;
    var element5;

    $('#myhtml > tbody > tr:nth-child(' +
parseInt(first_rw + 1) + ') > td').each(function() {
      var header = $(this).attr("headers");
      if (header == 'EDIT') {
        var span =
parseInt($(this).prop('rowspan') - 1);
        $(this).prop('rowspan', span);
        element1 = $(this).prop('outerHTML');
        $(this).remove();
      } else if (header == 'RISK_ORD') {
        var span =
parseInt($(this).prop('rowspan') - 1);
        $(this).prop('rowspan', span);
        element2 = $(this).prop('outerHTML');
        $(this).remove();
      } else if (header == 'RISK_DESC') {
        var span =
parseInt($(this).prop('rowspan') - 1);
        $(this).prop('rowspan', span);
        element3 = $(this).prop('outerHTML');
        $(this).remove();
      } else if (header == 'CF_RISK') {
        var span =
parseInt($(this).prop('rowspan') - 1);
        $(this).prop('rowspan', span);
        element4 = $(this).prop('outerHTML');
        $(this).remove();

```

```

        } else if (header == 'RISK_ID') {
            var span =
parseInt($(this).prop('rowspan') - 1);
            $(this).prop('rowspan', span);
            element5 = $(this).prop('outerHTML');
            $(this).remove();
        } else {
            $(this).remove();
        }
    });
    $('#myhtml > tbody > tr:nth-child(' +
parseInt(first_rw + 1 + 1) + ') > td').each(function() {
        if ($(this).attr("headers") == 'ERR_ORD')
        {
            $(this).before(element1 + element5 +
element2 + element3 + element4);
        }
    });
    $('#' + rowid).remove();
    del_err_ord(parseInt(rowspan - 1), first_rw);
    }
    } else {
        //for other error type of a risk, error removed
        and risk span must be reduced by 1
        var rowid1 = $(element).parents('tr').prop('id');
        var adderror;

        if ($('#" + rowid1 + " td:nth-child(1) >
#ADD_ERROR").length == 1) {
            adderror = true;
        }

        var id1, id2, id3, id4, id5, rowspan1, rowspan2,
rowspan3, rowspan4, rowspan5, loopcond = true;

        //to get the before Risk/Edit td
        $('#myhtml > tbody > tr > td').each(function() {
            if (loopcond) {
                if (rowid1 !=
$(this).parents('tr').prop('id')) {
                    if ($(this).attr("headers") ==
'EDIT') {
                        id1 = $(this).prop('id');
                        rowspan1 =
$(this).prop('rowspan');

```

```

    'RISK_DESC') {
        id2 = $(this).prop('id');
        rowspan2 =
    } else if ($(this).attr("headers") ==

$(this).prop('rowspan');

    'CF_RISK') {
        id3 = $(this).prop('id');
        rowspan3 =
    } else if ($(this).attr("headers") ==

$(this).prop('rowspan');

    'RISK_ORD') {
        id4 = $(this).prop('id');
        rowspan4 =
    } else if ($(this).attr("headers") ==

$(this).prop('rowspan');

    'RISK_ID') {
        id5 = $(this).prop('id');
        rowspan5 =
    } else if ($(this).attr("headers") ==

$(this).prop('rowspan');

    }
    } else {
        loopcond = false;
    }
}

});
$('#' + rowid).remove();
$('#' + id1).prop('rowspan', rowspan1 - 1);
$('#' + id2).prop('rowspan', rowspan2 - 1);
$('#' + id3).prop('rowspan', rowspan3 - 1);
$('#' + id4).prop('rowspan', rowspan4 - 1);
$('#' + id5).prop('rowspan', rowspan5 - 1);

//to add error type button to previous row, if
adderrortype row is removed
if (adderror) {
    $('#' + id1).find('span').each(function() {
        if ($(this).prop('id').indexOf('edit') !=
-1) {

            $(this).click();

        }
    });
} //alert(parseInt(rowspan1-
1)+' ,'+$('#'+id1).parents('tr').index());

```

```

                                del_err_ord(parseInt(rowspan1 - 1), $('##' +
id1).parents('tr').index());
                                }
                                },
                                No: function() {}
                                }
                                });
                                }
                                }

function edit(element, stat) {
    var cond;

    if (stat == 'new') {
        cond = true;
    } else {
        cond = validation('edit');
    }

    if (cond) {
        //disabling all items and insert error row excpt err and risk ord
        $('.myhtmlitem').prop('disabled', 'true');
        $('.myhtmlitem').addClass('disable');

        $('#myhtmlitem,.riskord,.errorord').parents('td').removeClass('current');
        $('.edit').removeClass('current');
        $('#ADD_ERROR').remove();
        curr_td_element = $(element).parents('td');
        var first_rw_no = $(curr_td_element).parents('tr').index();
        var rowspan = $(curr_td_element).prop("rowspan");
        if (rowspan == undefined) {
            rowspan = 1;
        }
        var curr_row = first_rw_no + rowspan;
        //enabling the edit row items
        for (i = 0; i < rowspan; i++) {
            var rownum = parseInt(first_rw_no + i + 1);

            $('#myhtml > tbody > tr:nth-child(' + rownum + ') >
td').find('textarea,input,select,span').each(function() {
                $(this).removeAttr('disabled');
                $(this).removeClass('disable');
                $(this).parents('td').addClass('current');

                //disabling tolerance level if impact score is HIGH

```



```

        if ($(this).parents('td').attr('headers') == "TOL") {
            if ($(this).val() == '0') {
                $(this).prop('disabled', 'true');
            }
        }
        //ends here
    });
}
//add error type link
var j = 1;

    $('#myhtml > tbody > tr:nth-child(' + curr_row +
    ')').find('td').each(function() {
        if ($(this).attr('headers') == "ERROR_DESC") {
            var err_typ_id = $('#myhtml > tbody > tr:nth-child(' +
            curr_row + ') > td:nth-child(' + j + ')').attr('id');
            $('#' + err_typ_id).append('<div><br><input type="button"
            class="adderror" id="ADD_ERROR" onclick="add_error(this)" value="Add
            Error Type"></div>');
        }
        j++;
    });
}

function merge() {
    $(document).ready(function() {
        $('#myhtml').each(function() {
            var risk_id = null;
            var edit_rsk, cf_risk, risk_seq, risk_desc, i = 1;

            // First, scan first row of headers for the column.
            $(this).find('th').each(function() {
                var col_id = $(this).attr('id');
                if (col_id == 'RISK_ID') {
                    risk_id = i;
                } else if (col_id == 'EDIT') {
                    edit_rsk = i;
                } else if (col_id == 'CF_RISK') {
                    cf_risk = i;
                } else if (col_id == 'RISK_ORD') {
                    risk_seq = i;
                } else if (col_id == 'RISK_DESC') {
                    risk_desc = i;
                }
            });
        });
    });
}

```

```

    }
    i++;
  });

  // first_instance holds the first instance of identical td
  var first_instance = null;
  var edit_col = null;
  var cfrisk_col = null;
  var risk_seq_col = null;
  var risk_desc_col = null;

  // iterate through rows
  $(this).find('tr').each(function() {
    // find the td of the correct column (determined by the
risk_desc set above)

    var risk_id_td = $(this).find('td:nth-child(' + risk_id +
  '));
    var col1 = $(this).find('td:nth-child(' + edit_rsk +
  '));
    var col2 = $(this).find('td:nth-child(' + cf_risk + '));
    var col3 = $(this).find('td:nth-child(' + risk_seq +
  '));
    var col4 = $(this).find('td:nth-child(' + risk_desc +
  '));

    if (risk_id_td.html() != undefined) {
      if (first_instance == null) {
        // must be the first row
        first_instance = risk_id_td;
        edit_col = col1;
        cfrisk_col = col2;
        risk_seq_col = col3;
        risk_desc_col = col4;
      } else if (risk_id_td.html() ==
first_instance.html()) {
        // the current td is identical to the previous
        // remove the current td
        risk_id_td.remove();
        col1.remove();
        col2.remove();
        col3.remove();
        col4.remove();

        risk_id_td.text("");
        var v_first_instance = 0;

```

```

        if (isNaN(first_instance.attr('rowspan'))) {
            v_first_instance = 1;
        } else {
            v_first_instance =
first_instance.attr('rowspan');
        }

        // increment the rowspan attribute of the first
instance
            first_instance.attr('rowspan',
parseInt(v_first_instance) + 1);
            edit_col.attr('rowspan',
parseInt(v_first_instance) + 1);
            cfrisk_col.attr('rowspan',
parseInt(v_first_instance) + 1);
            risk_seq_col.attr('rowspan',
parseInt(v_first_instance) + 1);
            risk_desc_col.attr('rowspan',
parseInt(v_first_instance) + 1);
            first_instance.attr("style", "border-bottom: 1px
solid #00433c");
        } else {
            // this cell is different from the last
            first_instance = risk_id_td;
            edit_col = col1;
            cfrisk_col = col2;
            risk_seq_col = col3;
            risk_desc_col = col4;
        }
    }
});
});
});
}
//function to get value from report for enabling disabling print button
function getval() {
    var val;
    $('#myhtml').find('td').each(function() {
        if ($(this).prop('headers') == 'RISK_DESC') {
            val += $(this).find('textarea').val();
        }
        if ($(this).prop('headers') == 'CF_RISK') {
            if ($(this).find('input').is(':checked') == true) {
                val += 'Y';
            }
        }
    });
}

```

ORACLE APEX

```
        } else {
            val += 'N';
        }
    }
    if ($(this).prop('headers') == 'ERROR_DESC') {
        val += $(this).find('textarea').val();
    }
    if ($(this).prop('headers') == 'IMPACT') {
        val += $(this).find('select').val();
    }
    if ($(this).prop('headers') == 'TOL') {
        val += $(this).find('select').val();
    }
    });
    return val;
}
var initval = getval();

$('#myhtml').on('change', 'input,select,textarea', function() {
    var newval = getval();
    if (newval == initval) {
        $('#PRINT').removeAttr("disabled");
    } else {
        $('#PRINT').prop("disabled", "true");
    }
});
//ends here
//function for setting err_ord while addin error
function add_err_ord(new_id, curspan, first_rw) {
    var options = "";
    curspan += 1;
    for (i = 1; i <= curspan; i++) {
        options += "<option value='" + i + "'>" + i + "</option>";
    }
    //retain selected value and increase option values
    for (i = 1; i <= curspan; i++) {
        var rownum = parseInt(first_rw + i);

        $('#myhtml > tbody > tr:nth-child(' + rownum + ') >
td').find('select').each(function() {
            if ($(this).hasClass('errorord')) {
                var curval = $(this).val();
                $(this).html(options);
                $(this).val(curval);
                $('#err_ord_' + new_id).val(curspan);
            }
        });
    }
}
```

This is the best learning we have ever had. In the beginning when we had the layout as requirement, we had decided it is not possible in Oracle Apex to design the layout. So we had given suggestions for the work, but later we have done tons of research and development to achieve this. So nothing is impossible in Oracle Apex.

Tree

Tree is the Oracle apex item which creates the hierarchical view from apex version 5.0. Oracle introduced new features which help in javascript based, cross browser component with optional keyboard navigation. This is the same as jstree .You can create tree from a query based on a hierarchical relationship identified by id and parent id in the column using start with .. connect by clause

A tree can be created based on a query and returns data that can be represented in a hierarchical manner. When you create an inbuilt tree component using the Create Page Wizard, it generates this hierarchical query for you based on the options you select.

- ☐ To create a tree on a new page:
- ☐ Click the Application Builder icon located in Workspace home page.
- ☐ Select an application.
- ☐ Click Create Page.
- ☐ For Create a Page:
 - Select Desktop as User Interface.
 - Select a page type - Select Tree.
 - Click Next.
- ☐ For Page Attributes, specify the following:
 - Enter Page Number.
 - Enter Page Name.
 - Select a page mode.
 - Select a region template for the tree region.
 - Enter a name for the region to contain the tree.
 - Click Next.

- ☐ For Navigation Preference, specify the type of navigation to include on this page and click Next. The navigation options (for example, navigation menu or tabs) depends upon the current application theme.
- ☐ For Table/View Owner and Name:
 - Table/View Owner - Select the owner of the table from which the tree will be based.
 - Table / View Name - Select the table or view which contains the columns to be included in the master page.
 - Click Next.
- ☐ For Query, identify the column you want to use as the ID, the Parent ID, and text that should appear on the nodes:
 - ID - Select the column to use as the ID.
 - Parent ID - Select the column to use as the parent ID.
 - Node Text - Select the text to appear on the tree nodes.
 - Start With - Select the column to be used to specify the root of the hierarchical tree query.
 - Start Tree - Choose how to start your query. Options include:
 - Based on Existing Item - Select an existing application or page item.
 - Based on an SQL Query - Enter an SQL query that returns a single row or single column.
 - Based on a Static Value - Enter a static value.
 - Value is NULL.
 - Click Next.
- ☐ For Where and Order by, specify the following:
 - Where Clause - Enter an optional WHERE clause. For example:
 - `ename='JONE'`
 - Order Siblings By - Select the order siblings by column, such as ENAME. The default value is based on the Node Text column selected.
 - Click Next.
- ☐ For Tree Attributes, specify the following:
 - Include Buttons - Select the buttons to include.

- Selected Node Page Item - Select the page or application item to hold the selected node value. This item can be used to save the tree state, by holding the value of the last selected node. The value of the selected node can be saved to the selected item using the node link attribute or a page process. When the tree is reloaded, the tree opens to the last selected tree node.
- Tooltip - Displays a tooltip when the mouse hovers over a leaf node. Options include:
 - Static Assignment - Specifies a static value to be used as the tooltip text. Substitution strings can be used in the static text, to incorporate information from the SQL query in the tooltip text. The following substitution strings can be used:
 - ☐ #VALUE# - refers to the value of ID column.
 - ☐ #TITLE# - refers to the value of Node Text column
 - ☐ #LINK# - refers to the value of Link option
 - Database Column - Select the column to use as the tooltip text.
- Link Option:
 - Existing Application Item - Makes the leaf node text a link. If you select this option, you must specify a page to link to and an existing application item to link leaf node text. This option only appears if you select the Existing Application Item link option.
 - Nothing
- Click Next.
- ☐ Confirm your selections and click Create.

So far we have seen how to create the default tree, but we as a developer need some extra touch which we can bring from the javascript .combine check box to tree and make the tree editable.

Step 1: create tree region with the following script

```
SELECT      CASE
            WHEN CONNECT_BY_ISLEAF = 1
              THEN 0
            WHEN LEVEL = 1
              THEN 1
```

ORACLE APEX

```
        ELSE -1
    END AS status,
    LEVEL, entry_text AS title, NULL AS icon,
    list_entry_id AS VALUE, entry_text AS tooltip, NULL AS LINK
FROM apex_application_list_entries
WHERE application_id = :app_id and list_entry_id not in (SELECT
list_entry_parent_id
FROM (SELECT list_entry_parent_id, COUNT (list_entry_id)
FROM apex_application_list_entries
WHERE list_entry_parent_id IN (
SELECT list_entry_parent_id
FROM (SELECT list_entry_parent_id, COUNT
(report_id)
FROM doy_tb_report_configuration d,
apex_application_list_entries a
WHERE d.report_id = a.list_entry_id
AND report_configuration !=
:p81_report_config
GROUP BY list_entry_parent_id))
GROUP BY list_entry_parent_id
INTERSECT
SELECT list_entry_parent_id, COUNT (report_id)
FROM doy_tb_report_configuration d,
apex_application_list_entries a
WHERE d.report_id = a.list_entry_id
AND report_configuration != :p81_report_config
GROUP BY list_entry_parent_id)
UNION
SELECT report_id
FROM doy_tb_report_configuration
WHERE report_configuration != :p81_report_config)
START WITH PARENT_ENTRY_TEXT = 'Reports'
CONNECT BY PRIOR list_entry_id = list_entry_parent_id
ORDER SIBLINGS BY list_id;
```

Step 2: Place this javascript in the page header

```
regTree = apex.jQuery("#Treestatic-id").find("div.tree");
regTree.tree({
    ui: {
        theme_name: "checkbox"
    },
    callback: {
        onchange: function(NODE, TREE_OBJ) {
            if (TREE_OBJ.settings.ui.theme_name == "checkbox") {
```



```

        var $this = $(NODE).is("li") ? $(NODE) :
$(NODE).parent();
        if ($this.children("a.unchecked").size() == 0) {
            TREE_OBJ.container.find("a").addClass("unchecked");
        }
        $this.children("a").removeClass("clicked");
        if ($this.children("a").hasClass("checked")) {

$this.find("li").andSelf().children("a").removeClass("checked").removeClass("undetermined").addClass("unchecked");
            var state = 0;
        } else {

$this.find("li").andSelf().children("a").removeClass("unchecked").removeClass("undetermined").addClass("checked");
            var state = 1;
        }
        $this.parents("li").each(function() {
            if (state == 1) {
                if ($(this).find("a.unchecked,
a.undetermined").size() - 1 > 0) {

$(this).parents("li").andSelf().children("a").removeClass("unchecked").removeClass("checked").addClass("undetermined");
                    return false;
                } else

$(this).children("a").removeClass("unchecked").removeClass("undetermined").addClass("checked");
            } else {
                if ($(this).find("a.checked,
a.undetermined").size() - 1 > 0) {

$(this).parents("li").andSelf().children("a").removeClass("unchecked").removeClass("checked").addClass("undetermined");
                    return false;
                } else

$(this).children("a").removeClass("checked").removeClass("undetermined").addClass("unchecked");
            }
        });
    },
    onopen: function(NODE, TREE_OBJ) {
        $(NODE).removeClass("open").addClass("closed");
    },
    onclose: function(NODE, TREE_OBJ) {
        $(NODE).removeClass("closed").addClass("open");
    }

```

ORACLE APEX

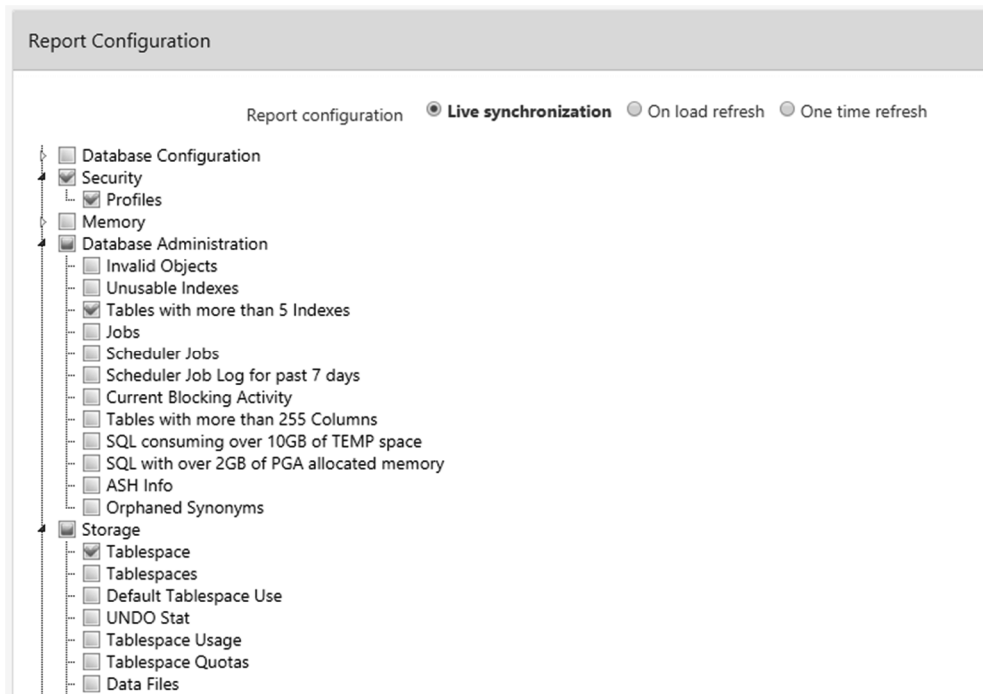
```
    } } } };
```

```
document.getElementById("tree373743011093796525").onclick = function() {
    clickFunction()
};

function clickFunction() {
    var lPassengers = [];
    $(".leaf a.checked").parent()
        .each(function() {
            lPassengers.push($(this).attr("id"))
        });
    $s("P81_LIST_ID", lPassengers.join(":"));
}

$(document).ready(function() {
    $.each($v("P81_LIST_ID").split ":"),
        function(intIndex, objValue) {
            $(".li#" + objValue + ".leaf a:first-child").click();
        });
});
```

After this you can see the customized screen



https://docs.oracle.com/html/E39147_04/tree_query.htm

ITEMS

Checkbox

Creating checkbox is relatively easy. By default the Oracle Apex allows to create a simple checkbox in the HTML region. The HTML region types are used for forms in Oracle Apex. Let's see how to create a simple checkbox in Form/HTML region.

Step 1: Create a static content region type.

Step 2: Create an Item either using Tree view/ Component view.

Step 3: Change the type from Text to Checkbox.

Step 4: Go to List of Values by scrolling down from right pane.

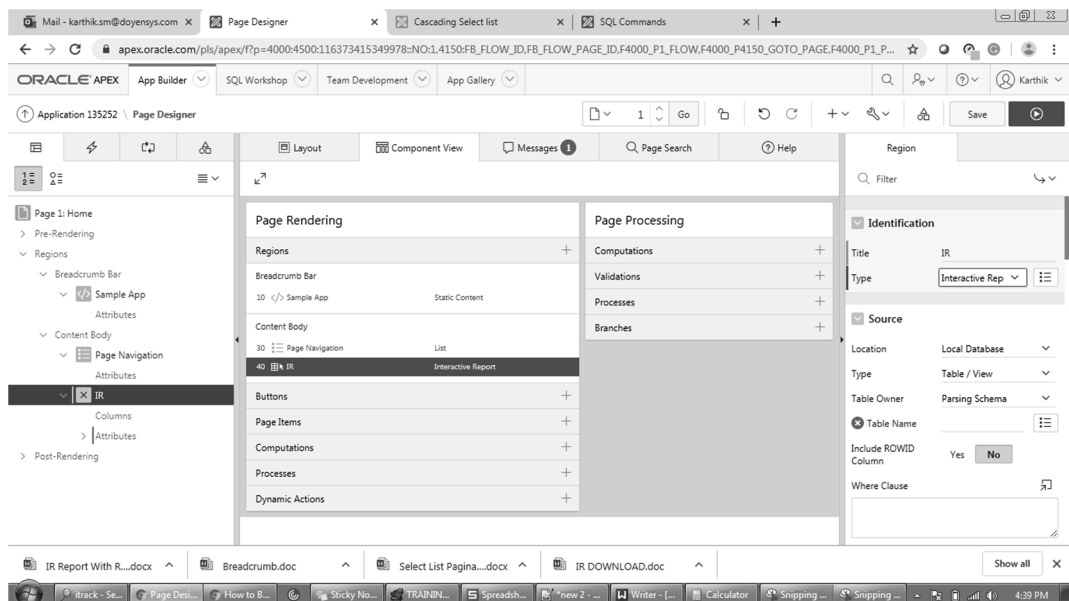
Step 5: Based on the requirement select the LOV type.

Step 6: Click Save and Run a page.

Finally, the checkbox will be seen in the page.

Now let's see how to do the customization.

Adding Checkbox column in Report

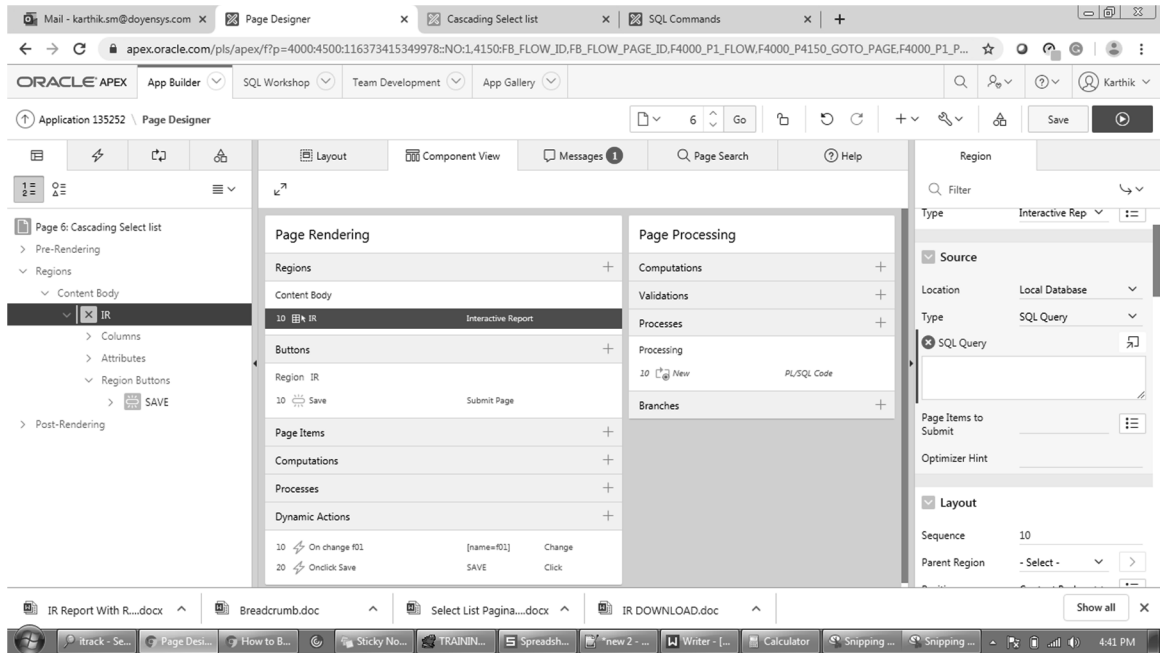


ORACLE APEX

Let's imagine you have an IR report where you want to add a checkbox column. Let's see how to do this in detail.

Step 1: In a Page, Create a Region and select report type as Interactive report

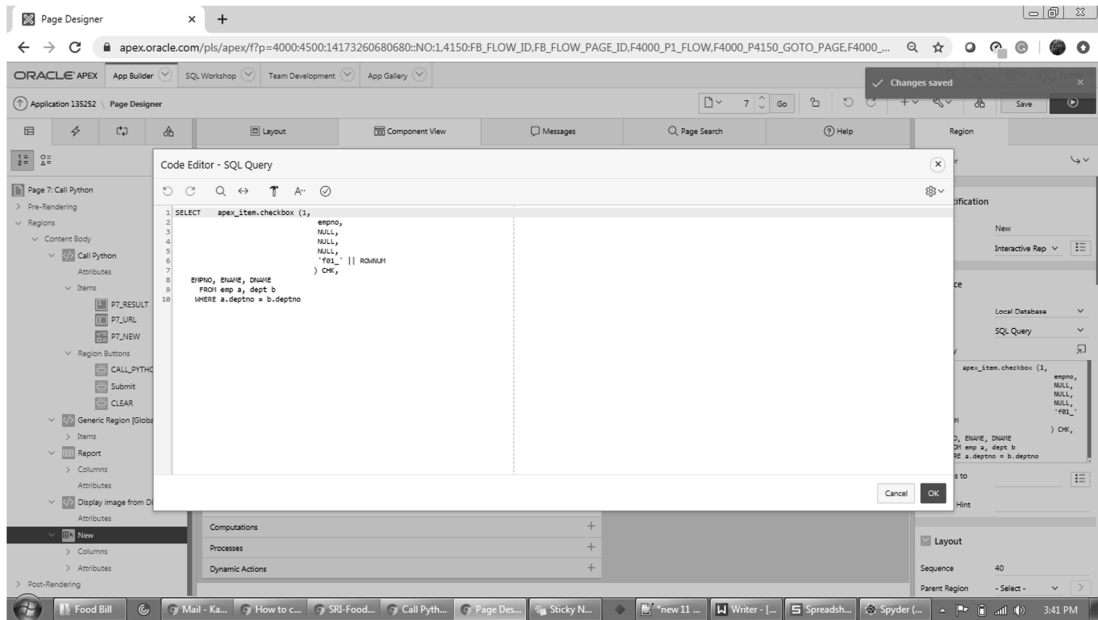
Step 2: Change the source type from Table/View to SQL Query



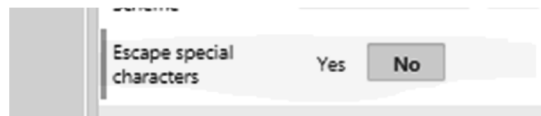
Step 3: Place the below code in the SQL query section,

```
SELECT      apex_item.checkbox (1,
                                empno,
                                NULL,
                                NULL,
                                NULL,
                                'f01_' || ROWNUM
                                ) CHK,

EMPNO, ENAME, DNAME
FROM emp a, dept b
WHERE a.deptno = b.deptno
```



Step 4: Set Column Type as *Plain Text* and *Escape special characters* as **No** for the following Column *CHK*



Step 5: Click on Save and Run a Page.

The screen look as same as the image given below.

Chk	Empno	Ename	Dname
☐	7839	BLAKE	SALES
☐	7698	BLAKE	SALES
☐	7782	CLARK	ACCOUNTING
☐	7566	JONES	RESEARCH
☐	7788	SCOTT	RESEARCH
☐	7902	FORD	RESEARCH
☐	7369	SMITH	RESEARCH
☐	7499	ALLEN	SALES
☐	7521	WARD	SALES
☐	7654	MARTIN	SALES
☐	7844	TURNER	SALES
☐	7876	ADAMS	RESEARCH
☐	7900	JAMES	SALES
☐	7934	MILLER	ACCOUNTING

Display Image

The display image item is nothing but it will enable the display of images stored in an image URL stored in another page item or in a BLOB column. The user is also able to alter where the image is stored, edit the item and configure it based on attribute.

It consists of some setting attributes below as

Based On

This option helps to determines where the item is retrieved from.

Alternative Text Column

This option helps if the user specified it, then the returned text is used as an alternative text for the image.

Filename Column

If the user specified this option, the returned file name is used by the browser in the Save As dialog when the user tries to saves the image.

BLOB Last Updated Column

If the user selects this option, the column is used for browser caching. It identifies if the image changed since the last browser request. If not entered, no caching is done.

Note: This attribute is case sensitive and the column must be of type DATE.

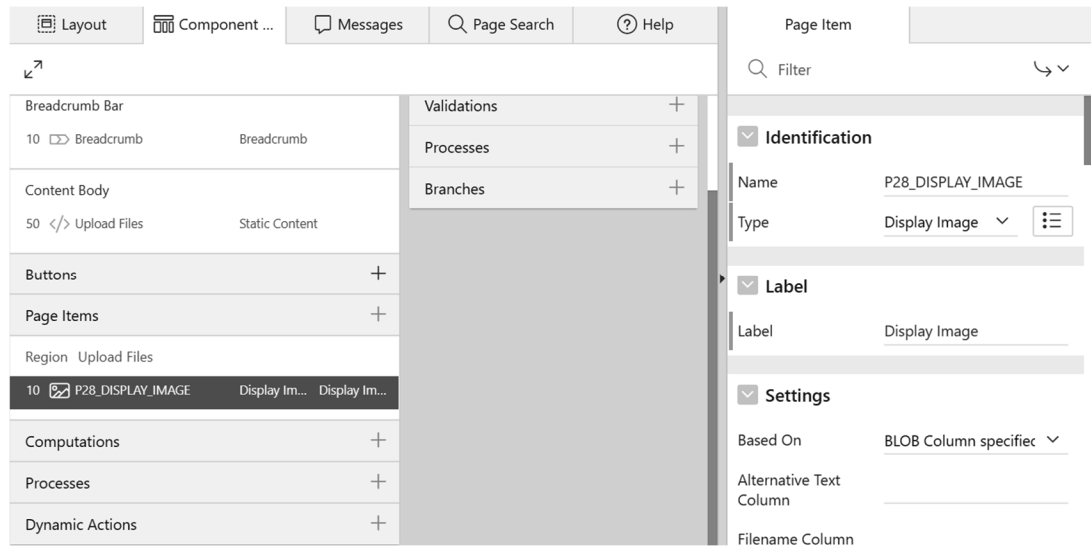
Step 1: Create an Item and select the type as Display Image.

Image URL stored in Page Item Value

This option helps to get the value of image stored in the session state of the page item.

BLOB Column specified in Item Source

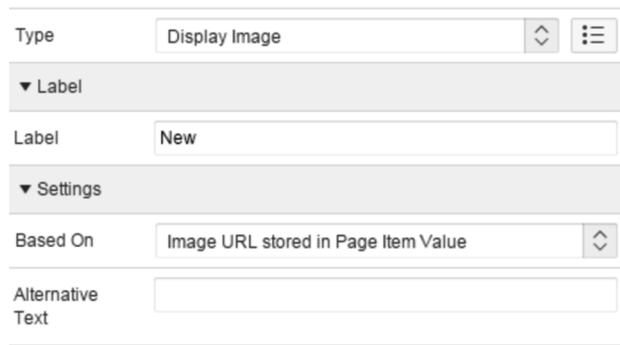
This options will read image from the blob column as specified in item source.



In Settings, we are able to set three types of options

BLOB Column returned by SQL statement

It will display the image based on the BLOB column which is returned by the SQL statement.



We can use the BLOB columns when working with images coming from the database, but here you can just use the "Image URL" option.

All you need to do then is to specify the URL in the item's source.

▼ Source

Type

Static Value

⌵

⋮

Static Value

🔗

#WORKSPACE_IMAGES#d.png

⌵

Used

Only when current value in session state is null

⌵

Sample Syntax

```

```

Result



File Browse

The File Browse item will display a text field with the option of browse Button. This enables the user to locate and upload a file from a local file system.

It consists of some setting features as below.

Value Required

This option helps the user as if Yes is selected and the page item is visible, Oracle Application Express automatically performs a NOT NULL validation when the page is submitted.

Storage Type

It helps to Identify where the uploaded file is stored.

MIME Type Column

This option helps to specify the name of the column where the MIME type of the uploaded file should be stored.

Filename Column

This option helps to specify the name of the column where the file name of the uploaded file should be stored.

Character Set Column

This option helps to specify the name of the column where the character set of the uploaded file should be stored.

BLOB Last Updated Column

If the user specifies the name of the column to update with the current time stamp when a file gets uploaded.

Display Download Link

This option helps the user if Select Yes to display the download link next to the Browse button.

Download Link Text

It will display text used for the download link to download the files.

Content Disposition

This option helps to determines if the downloaded content is displayed in line within the Browser window or if an Open dialog displays.

Step 1: Create a page item and set type as “File Browse”.

Step 2: In Settings Option, Select the Storage Type as
TABLE_APEX_APPLICATION_FILES.

ORACLE APEX

Page Rendering		Page Processing		Identification	
Regions +		Computations +		Name P28_FILE_BROWSE	
Breadcrumb Bar		Validations +		Type File Browse... ▾	
10 <> Breadcrumb Breadcrumb		Processes +			
Content Body		Branches +		Label File Browse	
50 </> Upload Files Static Content				Settings	
Buttons +				Storage Type Table APEX_APPLICATION	
Page Items +				Purge File at End of Session ▾	
Region Upload Files				Allow Multiple Files Yes No	
10 P28_FILE_BROWSE File Browse File Brows...					

Step 3: Create two buttons as Upload and Cancel.

↖ ↗

Breadcrumb Bar	
10 <> Breadcrumb	Breadcrumb
Content Body	
50 </> Upload Files	Static Content
Buttons +	
Region Upload Files	
10 Upload	Submit Page
10 Cancel	Go To Page 28
Page Items +	

Step 4: Create the function by using the below Query

```
create or replace FUNCTION pdd_clobfromblob (
    p_blob BLOB
) RETURN CLOB IS

    l_clob          CLOB;
    l_dest_offsset  INTEGER := 1;
    l_src_offsset   INTEGER := 1;
```

```

    l_lang_context    INTEGER := dbms_lob.default_lang_ctx;
    l_warning         INTEGER;
BEGIN
    IF p_blob IS NULL THEN
        RETURN NULL;
    END IF;
    dbms_lob.createtemporary(lob_loc => l_clob, cache => false);
    dbms_lob.converttoclob(dest_lob => l_clob, src_blob => p_blob, amount
=> dbms_lob.lobmaxsize, dest_offset => l_dest_offsset, src_offset =>
l_src_offsset
, blob_csid => dbms_lob.default_csid, lang_context =>
l_lang_context, warning => l_warning);

    RETURN l_clob;
END PDD_clobfromblob;
/

```

Step 5: Create the type & Object by using the below Query

```

CREATE OR REPLACE TYPE "PDD_BURSTING_OBJECT_TYPE" IS OBJECT (
    batch_number          VARCHAR2(100),      test_name
    VARCHAR2(100),
    test_sequence         VARCHAR2(100),      test_value
    VARCHAR2(100),
    actual_result_date     VARCHAR2(100),      equipment_info
    VARCHAR2(100),
    tester                VARCHAR2(100),      inspector
    VARCHAR2(100),
    product               VARCHAR2(100),      bin_number
    VARCHAR2(100),
    inspection_type        VARCHAR2(100),      flag
    VARCHAR2(100)
);
CREATE OR REPLACE TYPE "PDD_BURSTING_TABLE_TYPE" AS
TABLE OF PDD_bursting_object_type

```

Step 6: Create tables by using the below Query

```

CREATE TABLE "PDD_BURTSING_FILES_TBL"
( "FILE_ID" NUMBER, "FILE_NAME" VARCHAR2(500),
  "MIME_TYPE" VARCHAR2(50), "BLOB_CONTENT" BLOB,
  "CREATED_BY" VARCHAR2(50), "CREATED_DATE" DATE,
  "STATUS" VARCHAR2(30) );
CREATE TABLE "PDD_BURSTING_STG_TBL"
( "BURSTING_ID" NUMBER, "FILE_ID" NUMBER,
  "FILE_NAME" VARCHAR2(100), "BATCH_NUMBER" VARCHAR2(100),
  "TEST_NAME" VARCHAR2(100), "TEST_SEQUENCE" VARCHAR2(100),

```

ORACLE APEX

```
"TEST_VALUE" VARCHAR2(100),      "ACTUAL_RESULT_DATE" VARCHAR2(100),
"EQUIPMENT_INFO" VARCHAR2(100),    "TESTER" VARCHAR2(100),
"INSPECTOR" VARCHAR2(100),        "PRODUCT" VARCHAR2(100),
"BIN_NUMBER" VARCHAR2(100),        "INSPECTION_TYPE" VARCHAR2(100),
"FLAG" VARCHAR2(100),             "IS_RETEST" VARCHAR2(1),
"SAMPLE_NO" VARCHAR2(100),         "IS_ACTIVE" VARCHAR2(1),
"STATUS" VARCHAR2(50),             "ERROR_MESSAGE" VARCHAR2(500),
"CREATED_BY" VARCHAR2(50),         "CREATED_DATE" DATE,
"UPDATED_BY" VARCHAR2(50),         "UPDATED_DATE" DATE,
"SAMPLE_TYPE" VARCHAR2(1),         "FAILURE_CRITERIA" NUMBER,
"DEPARTMENT" VARCHAR2(100)
);/
CREATE TABLE "PDD_BURTSING_FILES_HIS_TBL"
(  "HIS_ID" NUMBER,  "FILE_ID" NUMBER,
"FILE_NAME" VARCHAR2(50),  "MIME_TYPE" VARCHAR2(50),
  "BLOB_CONTENT" BLOB,    "CREATED_BY" VARCHAR2(50),
  "CREATED_DATE" DATE,    "STATUS" VARCHAR2(100)
);/
```

Step 7: Create function by using the below Query

```
CREATE OR REPLACE FUNCTION PDD_bursting_extract_csv (
  p_filename VARCHAR2,
  p_type VARCHAR2
) RETURN PDD_bursting_table_type
PIPELINED
IS
  l_clob CLOB;
  l_n PLS_INTEGER;
  l_line VARCHAR2(4000);
BEGIN
  BEGIN
    IF p_type = 'UPLOAD' THEN
      BEGIN
        SELECT PDD_clobfromblob(blob_content)
        INTO l_clob
        FROM PDD_burtsing_files_tbl
        WHERE upper(file_name) = upper(p_filename)
        AND 'UPLOAD' = p_type;
      END;
    END IF;

    IF substr(l_clob,-1,1) != chr(10) THEN
      l_clob := l_clob || chr(10);
    END IF;
    l_n := instr(l_clob,chr(10) );
```

```

l_line := substr(l_clob,1,l_n)  || ',';
WHILE l_n > 0 LOOP
    PIPE ROW (
PDD_bursting_object_type(substr(l_line,1,instr(l_line,',') -
1),substr(l_line,instr(l_line,',') + 1,instr(l_line,',',1,2) -
instr(l_line,',') - 1),substr(l_line,instr(l_line,',',1,2) +
1,instr(l_line,',',1,3) - instr(l_line,',',1,2) -
1),substr(l_line,instr(l_line,',',1,3) + 1,instr(l_line,',',1,4) -
instr(l_line,',',1,3) - 1),substr(l_line,instr(l_line,',',1,4) +
1,instr(l_line,',',1,5) - instr(l_line
,',',1,4) - 1),substr(l_line,instr(l_line,',',1,5) +
1,instr(l_line,',',1,6) - instr(l_line,',',1,5) -
1),substr(l_line,instr(l_line,',',1,6) + 1,instr(l_line,',',1,7) -
instr(l_line,',',1,6) - 1),substr(l_line,instr(l_line,',',1,7) +
1,instr(l_line,',',1,8) - instr(l_line,',',
,1,7) - 1),substr(l_line,instr(l_line,',',1,8) + 1,instr(l_line,',',1,9)
- instr(l_line,',',1,8) - 1),substr(l_line,instr(l_line,',',1,9)+
1,instr(l_line,',',1,10) - instr(l_line,',',1,9) -
1),substr(l_line,instr(l_line,',',1,10) + 1,instr(l_line,',',1,11) -
instr(l_line,',',
,1,10) - 1),substr(l_line,instr(l_line,',',1,11) +
1,instr(l_line,',',1,12) - instr(l_line,',',1,11) - 1) ) );
        l_clob := substr(l_clob,l_n + 1);
        l_n := instr(l_clob,chr(10) );
        l_line := substr(l_clob,1,l_n)
        || ',';
    END LOOP;
END;
END;
```

Step 8: Create a Process in Apex in Page Processing Section.

Code Editor - PL/SQL Code

```

1 DECLARE
2   V_HISID NUMBER;
3   CURSOR cur_upload IS SELECT
4     id,
5     name,
6     filename,
7     mime_type,
8     blob_content
9     FROM
10      apex_application_temp_files
11     WHERE
12      name =:P28_FILE_BROWSE;
13
14 BEGIN
15   FOR i IN cur_upload LOOP
16     SELECT (NVL(MAX(HIS_ID),0)+1) HIS_ID INTO V_HISID FROM TTK_PDD_BURTSING_FILES_HIS_TBL;
17     INSERT INTO TTK_PDD_BURTSING_FILES_HIS_TBL
18     SELECT V_HISID,A.* FROM ttk_pdd_burtsing_files_tbl A
19     where upper(file_name)=upper(i.filename);
20     delete from ttk_pdd_burtsing_files_tbl
21     where upper(file_name)=upper(i.filename);
```

Cancel OK

Sample Code

Process 1

```

DECLARE
V_HISID NUMBER;
    CURSOR cur_upload IS
        SELECT id, name, filename, mime_type, blob_content
        FROM apex_application_temp_files
        WHERE name =:P28_FILE_BROWSE;
BEGIN
    FOR i IN cur_upload LOOP
        SELECT (NVL(MAX(HIS_ID),0)+1) HIS_ID
        INTO V_HISID FROM PDD_BURTSING_FILES_HIS_TBL;
        INSERT INTO PDD_BURTSING_FILES_HIS_TBL
        SELECT V_HISID,A.* FROM PDD_burtsing_files_tbl A
        where upper(file_name)=upper(i.filename);
        delete from PDD_burtsing_files_tbl
        where upper(file_name)=upper(i.filename);
        INSERT INTO PDD_burtsing_files_tbl (
            file_id,
            file_name,
            mime_type,
            blob_content,
            created_by,
            created_date,
            status
        ) VALUES (
            i.id,
            upper(i.filename),
            i.mime_type,
            i.blob_content,
            :app_user,
            SYSDATE,
            'STAGING'
        );

    END LOOP;
END;
```

Process 2

```

DECLARE
    v_bursting_id    NUMBER;
    v_fileid         NUMBER;
    v_file_name      VARCHAR2(200);
```

```

CURSOR cur_stg_file IS
  SELECT DISTINCT file_id, file_name
    FROM PDD_burtsing_files_tbl
   WHERE status = 'STAGING';

CURSOR cur_loaded_data (
  p_filename VARCHAR2
) IS SELECT a.*
  FROM TABLE ( PDD_bursting_extract_csv(p_filename,'UPLOAD') ) a
  WHERE upper(batch_number) <> 'BATCH_NUMBER';

BEGIN
  BEGIN
    FOR i IN cur_stg_file LOOP
      DELETE FROM PDD_bursting_stg_tbl
      WHERE
        upper(file_name) = upper(i.file_name)
        AND status IN ( 'STAGING','FAILED');

      FOR k IN cur_loaded_data(i.file_name) LOOP
        SELECT  nvl(MAX(bursting_id),0) + 1 bursting_id
          INTO v_bursting_id
          FROM PDD_bursting_stg_tbl;

        v_fileid := i.file_id;
        v_file_name := i.file_name;
        INSERT INTO PDD_bursting_stg_tbl (
          bursting_id,                file_id,
          file_name,                  batch_number,
          test_name,                  test_sequence,
          test_value,                 actual_result_date,
          equipment_info,             tester,
          inspector,                  product,
          bin_number,                 inspection_type,
          flag,                       is_retest,
          sample_no,                  is_active,
          status,                     error_message,
          created_by,                 created_date,
          updated_by,                 updated_date
        ) VALUES (
          v_bursting_id,              v_fileid,
          v_file_name,                k.batch_number,
          k.test_name,                k.test_sequence,
          k.test_value,
          k.actual_result_date,
          k.equipment_info,           k.tester,

```

ORACLE APEX

```
                k.inspector,                k.product,
                k.bin_number,                k.inspection_type,
                k.flag,                      'N',
                NULL,                        'Y',
                'STAGING',                   NULL,
                :app_user,                   SYSDATE,
                NULL,                        NULL
            );

        END LOOP;
    END LOOP;
END;
EXCEPTION
    WHEN OTHERS THEN
        raise_application_error(-20002,'Error in Getting file records.');
```

Step 9: Create an Interactive Report in Oracle Apex by using the below query.

```
select BURSTING_ID,      FILE_ID,      FILE_NAME,      BATCH_NUMBER,
       TEST_NAME,        TEST_SEQUENCE, TEST_VALUE,
ACTUAL_RESULT_DATE,
       EQUIPMENT_INFO,    TESTER,        INSPECTOR,      PRODUCT,
BIN_NUMBER,
       INSPECTION_TYPE,   FLAG,          IS_RETEST,      SAMPLE_NO,
IS_ACTIVE,
       STATUS,            ERROR_MESSAGE,   CREATED_BY,      CREATED_DATE,
       UPDATED_BY,        UPDATED_DATE,    SAMPLE_TYPE,
FAILURE_CRITERIA,
       DEPARTMENT
from PDD_BURSTING_STG_TBL;
```

Result

When the user uploads the .CSV Format file by using the File browse page Item, it will call the Plsql Process and all .CSV File Records will be loaded into the staging table and display in interactive report.

CSV File Browse

File Browse

[Download Sample File](#)

Bursting Id	File Id	File Name	Batch Number	Test Name	Test Sequence	Test Value	Actual Result Date	Equipment Info	Tester	Inspector
1	32979403685532001458	TBL19098F_04_201906252238_JEGAN.CSV	TBL19098F	P	1	0.8613	25/06/2019 22:38:31 PM	6	-	-
2	32979403685532001458	TBL19098F_04_201906252238_JEGAN.CSV	TBL19098F	P	2	1.2472	25/06/2019 22:38:31 PM	1	-	-

CSV File Browse

Bursting Id	File Id	File Name	Batch Number	Test Name	Test Sequence	Test Value	Actual Result Date	Equipment Info	Tester	Inspector
133	32979403685532001458	TBL19098F_04_201906252238_JEGAN.CSV	TBL19098F	V	34	20.71733	25/06/2019 22:38:31 PM	3	-	-
136	32979403685532001458	TBL19098F_04_201906252238_JEGAN.CSV	TBL19098F	V	55	24.0844	25/06/2019 22:38:31 PM	6	-	-
137	32979403685532001458	TBL19098F_04_201906252238_JEGAN.CSV	TBL19098F	V	56	27.7494	25/06/2019 22:38:31 PM	7	-	-
138	32979403685532001458	TBL19098F_04_201906252238_JEGAN.CSV	TBL19098F	V	57	26.1973	25/06/2019 22:38:31 PM	0	-	-
139	32979403685532001458	TBL19098F_04_201906252238_JEGAN.CSV	TBL19098F	V	58	26.0174	25/06/2019 22:38:31 PM	3	-	-

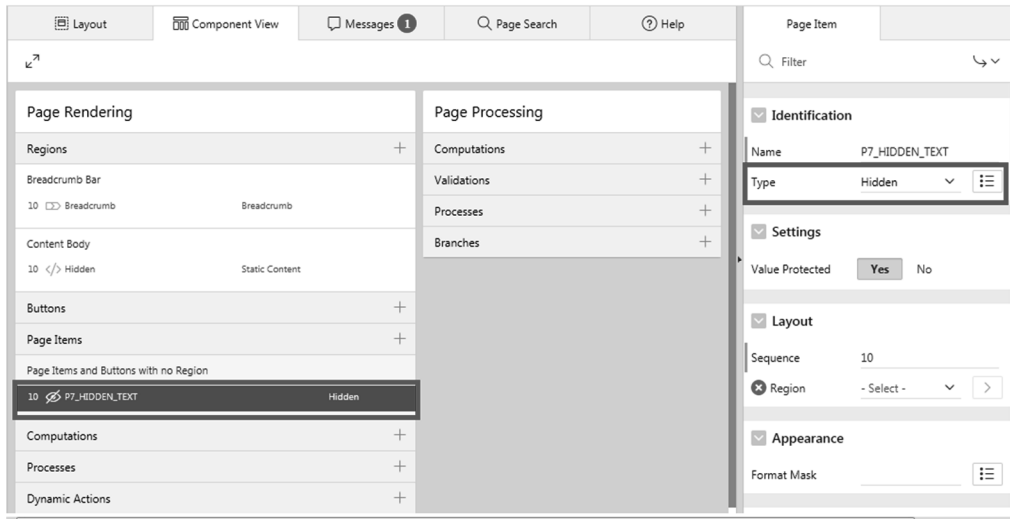
Hidden

Items that are included within the page source but are not rendered. Hidden item values are saved in session state. They are generally used to store values required by page processing or other page items, but should not be displayed to the end user.

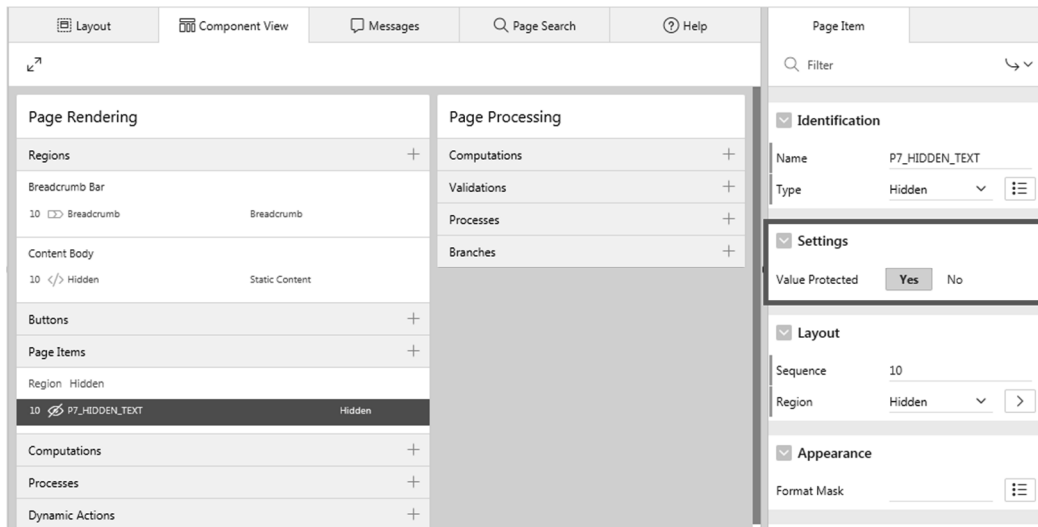
Below we are going to see how the hidden items are created and their Customization. We are able to create the hidden items using two types like Page Designer and Dynamically.

Page Designer

We can create a new page Item and select the type as “Hidden”.



Value Protected



Settings, Value is Protected - Select Yes to prevent hidden values from being manipulated when a page is posted. If we set the value as “Yes”, then checksum will be generated whenever the form is loaded.

When we submit a page with different value for the hidden item, then the checksum is not valid and it may throw an error.

If we set the value as “No”, we can manipulate the data for the hidden Items.

APEX_ITEM.HIDDEN

We can create the form hidden item dynamically using the function APEX_ITEM.HIDDEN.

Syntax

```
APEX_ITEM.HIDDEN (
    p_idx          IN      NUMBER,
    p_value         IN      VARCHAR2 DEFAULT
    p_attributes    IN      VARCHAR2 DEFAULT NULL,
    p_item_id       IN      VARCHAR2 DEFAULT NULL,
    p_item_label    IN      VARCHAR2 DEFAULT NULL
) RETURN VARCHAR2;
```

Parameters:

p_idx	Number to identify the item you want to generate. The number determines which G_FXX global is populated
p_value	Value of the hidden input form item
p_attributes	Extra HTML parameters you want to add
p_item_id	HTML attribute ID for the <input> tag
p_item_label	Invisible label created for the item

Eg:

```
SELECT
    empno,
    APEX_ITEM.HIDDEN(1,empno) hid_emp,ename from emp;
```

Create a new classic report region with the above SQL Code and set the Source type as “SQL Query”.

ORACLE APEX

The screenshot displays the Oracle APEX Classic Report editor. The top navigation bar includes 'Layout', 'Component View', 'Messages', 'Page Search', and 'Help'. The main workspace is divided into 'Page Rendering' and 'Page Processing' sections. The 'Page Rendering' section shows a breadcrumb bar and a content body with a hidden region named 'Hidden_classic'. The 'Page Processing' section shows computations, validations, processes, and branches. The right-hand panel shows the configuration for the 'Hidden_classic' report, including its title, type (Classic Report), location (Local Database), and the SQL query used to fetch data from the EMP table.

Save and Run the Page.

Output:

Hidden_classic

Empno ↑≡	Hid Emp	Ename
7369	<input type="hidden" name="f01" value="7369" />	SMITH
7499	<input type="hidden" name="f01" value="7499" />	ALLEN
7521	<input type="hidden" name="f01" value="7521" />	WARD
7566	<input type="hidden" name="f01" value="7566" />	JONES
7654	<input type="hidden" name="f01" value="7654" />	MARTIN
7698	<input type="hidden" name="f01" value="7698" />	BLAKE
7782	<input type="hidden" name="f01" value="7782" />	CLARK
7788	<input type="hidden" name="f01" value="7788" />	SCOTT
7839	<input type="hidden" name="f01" value="7839" />	KING

Set the “Escape special characters” to “No” and the output looks below.

Hidden_classic		
Empno ↑	Hid Emp	Ename
7369		SMITH
7499		ALLEN
7521		WARD
7566		JONES
7654		MARTIN
7698		BLAKE
7782		CLARK
7788		SCOTT
7839		KING

The “Hid Emp” values seem hidden, we can view the value by inspect (Right click on the Hid Emp field and Click Inspect) the corresponding field.

Empno	Hid Emp	Ename
7369		SMITH
7499		ALLEN
7521		WARD

```

<td class="t-Report-cell" headers="EMPNO">7369</td>
<td class="t-Report-cell" headers="HID_EMP"> == $0
  <input type="hidden" name="f01" value="7369">
</td>
<td class="t-Report-cell" headers="ENAME">SMITH</td>

```

Reference:<https://docs.oracle.com/en/database/oracle/application-express/19.1/htmldb/about-item-types.html#GUID-8CAA1BA5-DCA8-4DF0-A841-3DC40E67CA9C>

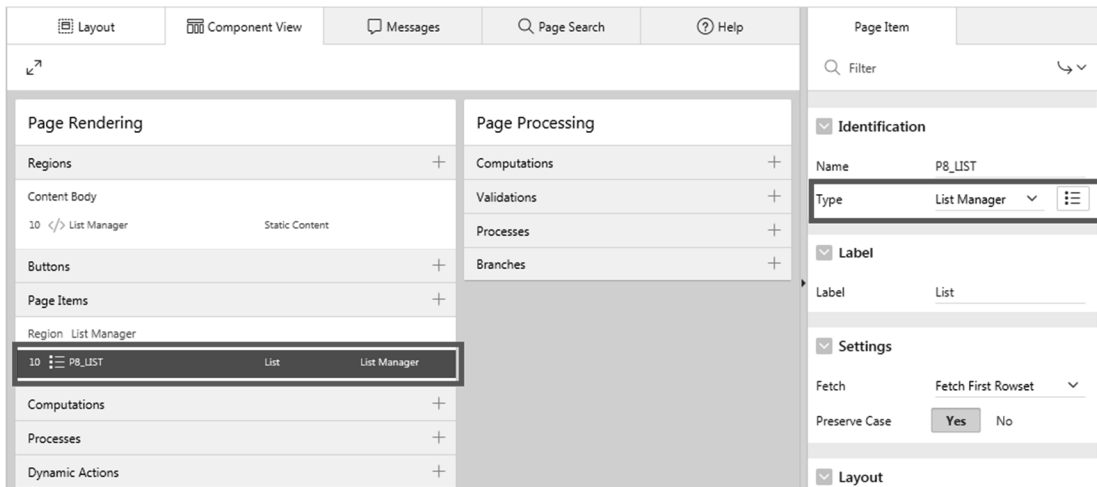
List Manager

List Manager is a type item, that allows users to enter values and it will be added in the list. It displays as a text item with a popup list of values icon, Add and Remove buttons, and a list of selected values.

You can type in the value or pick from the list of available items. You can then utilize the buttons to manage the values selected. The selected values are stored in a single colon-delimited string.

Below we can see how to create a List Manager and their main attributes.

Create a new page item and select the type as “**List Manager**” and set the label as “List”.



Page Designer attributes:

Settings

Fetch - Select how data is displayed when the Popup LOV is opened.

Page Rendering	Page Processing	
Regions +	Computations +	☒ Identification Name P8_LIST Type List Manager
Content Body	Validations +	
10 </> List Manager Static Content	Processes +	
Buttons +	Branches +	
Page Items +		
Region List Manager		☒ Label Label List
10 ≡ P8_LIST List List Manager		☒ Settings Fetch Fetch First Rowset Preset Fetch First Rowset
Computations +		☒ Layout Sequence 10 Region List Manager
Processes +		
Dynamic Actions +		

Validation

Value Required- If select **Yes** and the page item is visible, Oracle Application Express automatically performs a NOT NULL validation when the page is submitted.

Page Rendering	Page Processing	
Regions +	Computations +	Format Mask Value Placeholder ☒ Validation Value Required <input checked="" type="text" value="Yes"/> No
Content Body	Validations +	
10 </> List Manager Static Content	Processes +	
Buttons +	Branches +	
Page Items +		
Region List Manager		☒ List of Values Type SQL Query SQL Query select ename,empno '-' ename from emp Display Extra Values <input checked="" type="text" value="Yes"/> No
10 ≡ P8_LIST List List Manager		
Computations +		
Processes +		
Dynamic Actions +		

List of Values

Type

We can choose input type for the list manager, whether you can set the data for the List Manager by using Shared Component, SQL Query, Static values and PLSQL function body returning SQL query.

For this below example we choose the **type** as “SQL Query”

In “**SQL Query**”, place the below code

ORACLE APEX

```
select ename,empno||'-'||ename from emp;
```

The screenshot displays the Oracle APEX page editor interface. On the left, the 'Page Rendering' pane shows a list of components: Regions, Content Body, Buttons, Page Items, Computations, Processes, and Dynamic Actions. The 'Page Items' section is expanded, showing a list item named 'PB_LIST' of type 'List'. On the right, the 'Page Processing' pane shows 'Computations', 'Validations', 'Processes', and 'Branches'. The 'Validation' pane is also expanded, showing a 'List of Values' configuration. The 'List of Values' configuration includes a 'Type' dropdown set to 'SQL Query', a text area containing the SQL query 'select ename,empno||'-'||ename from emp;', and a 'Display Extra Values' checkbox set to 'Yes'.

Click Save and Run the page and the list manager looks as below.

The screenshot shows the Oracle APEX List Manager interface. At the top, there is a horizontal entry box labeled 'List' with a small icon to its right. Below the entry box are 'Add' and 'Remove' buttons. The main area of the List Manager is a large vertical box containing a list of items. The list is currently empty.

The top horizontal box is the entry box where the user will enter the data or select the value from the list by clicking the small icon near the entry box.

Click on the “**Add**” button to add the entry to the list.

The lower vertical box will contain the entered items. The user can remove items from the list by selecting the item in the list and pressing the “**Remove**” button.

The screenshot shows the 'List Manager' component. At the top, there is a header 'List Manager'. Below it, there is a text input field labeled 'List' containing the value '7698-BLAKE'. To the right of the input field is a menu icon (three horizontal lines with a dot). Further right are two buttons: 'Add' and 'Remove'. Below the input field is a list box containing two entries: '7782-CLARK' and '7788-SCOTT'. The list box has a vertical scrollbar on the right side.

The list box will expand and collapse based on the entries.

On Page Submission, the value return in the form of colon separated

We can also set as Cascading LOV, by selecting the “Cascading LOV Parent Item” and select the Item to submit value.

The screenshot shows the APEX Page Editor interface. On the left, there is a 'Page Rendering' pane with a tree view showing the page structure. The 'List Manager' component is selected. On the right, there is a 'Page Processing' pane with a tree view showing the component's configuration. The 'List of Values' configuration is expanded, showing the following settings:

- Type:** SQL Query
- SQL Query:** `select ename,empno||'-'||ename from emp;`
- Display Extra Values:** Yes
- Cascading LOV Parent Item(s):** P8_LIST
- Items to Submit:** P8_LIST
- Optimize Refresh:** Yes

Number Field

Displays a number field. You can configure number range checks for values entered by configuring the Minimum Value and Maximum Value attributes. This item type automatically checks if the entered value is a number. No extra validations are required.

Page Designer:

Create a New Item and set the type as “Number Field”.

Page Rendering

Regions

10 Breadcrumb

20 Hidden

20 Hidden classic

Buttons

Page Items

Region Hidden

10 P7_NUMBER

Computations

Processes

Page Processing

Computations

Validations

Processes

Branches

Identification

NameP7_NUMBER

TypeNumber Field

Label

LabelNum_check

Settings

Minimum Value

Maximum Value

Number AlignmentRight

Layout

Sequence10

Settings, Minimum Value - Enter the minimum value permitted. The value can be a static value or a dynamic value using substitution syntax.

Page Rendering

Regions

10 Breadcrumb

20 Hidden

20 Hidden classic

Buttons

Page Items

Region Hidden

10 P7_NUMBER

Computations

Processes

Page Processing

Computations

Validations

Processes

Branches

Identification

NameP7_NUMBER

TypeNumber Field

Label

LabelNum_check

Settings

Minimum Value10

Maximum Value

Number AlignmentRight

Layout

Sequence10

Settings, Maximum Value – Enter the maximum value permitted. The value can be a static value or a dynamic value using substitution syntax.

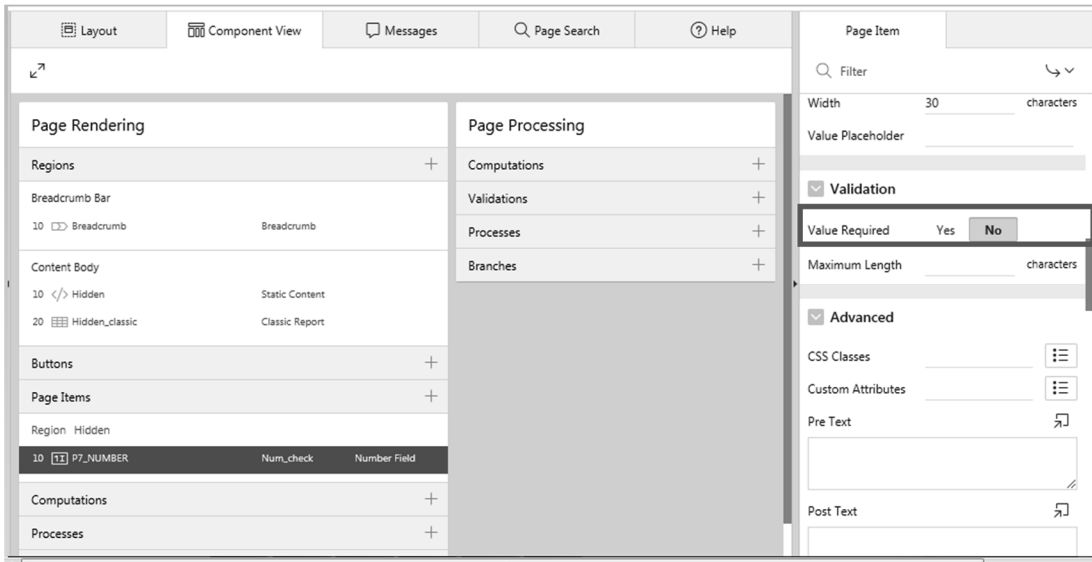
The screenshot displays the APEX Page Editor interface. On the left, the 'Page Rendering' pane shows a tree structure with 'Regions' expanded, containing 'Breadcrumb Bar' and 'Content Body'. The 'Page Items' section is highlighted, showing a list of items including 'P7_NUMBER' (Number Field). The 'Page Processing' pane on the right shows sections for 'Computations', 'Validations', 'Processes', and 'Branches'. The 'Properties' pane on the far right is open for the 'P7_NUMBER' item, showing the following settings:

- Identification:** Name: P7_NUMBER, Type: Number Field.
- Label:** Label: Num_check.
- Settings:** Minimum Value: 10, Maximum Value: 40 (highlighted with a red box), Number Alignment: Right.
- Layout:** Sequence: 10.

Settings, Number Alignment – Select how to align the number that displays. This setting does not apply when the item is rendered as Read Only or where HTML Form Element Attributes are specified.

This screenshot shows the same APEX Page Editor interface as the previous one, but with the 'Number Alignment' dropdown menu open in the 'Settings' section. The dropdown menu lists the following options: 'Right' (selected), '- Select -', 'Left', 'Center', and 'Right' (repeated). The 'Layout' section is also visible, showing 'Sequence' as 10. Other settings like 'Minimum Value' (10), 'Maximum Value' (40), 'Region' (Hidden), 'Start New Row' (Yes), 'Column' (Automatic), and 'Column Span' (Automatic) are also visible.

Validation, Value Required- If select **Yes** and the page item is visible, Oracle Application Express automatically performs a NOT NULL validation when the page is submitted.



Click Save and Run the Page.

Output

Minimum value and Maximum value for the item set are 10 and 40. So the value will be accepted only in the range of 10 to 40

Enter the value for **Num_check** = 1 and its shows the error value should be in the range of 10 to 40



If you enter the value as 12. Then value will be accepted for the next level processing.

Instead of Number, we try to enter any text values, then it will throw an error as “Num_check” must be numeric.

Reference: <https://docs.oracle.com/en/database/oracle/application-express/19.1/htmldb/about-item-types.html#GUID-8CAA1BA5-DCA8-4DF0-A841-3DC40E67CA9C>

Password

The Password field item will render as an HTML Password Form Element.

It defines that what is entered into a password field shows up as dots on the screen. This Field helps to prevent others from reading the password on the screen. It consists of below settings as

Value Required

If you select Yes and the page item is visible, Oracle Application Express automatically performs a NOT NULL validation when the page is submitted.

Submit when Enter Pressed

This option helps the user if Select Yes is chosen, to submit the page when ENTER is pressed. This saves the password in a database table when the page is submitted. Select No to not save the password in a database table.

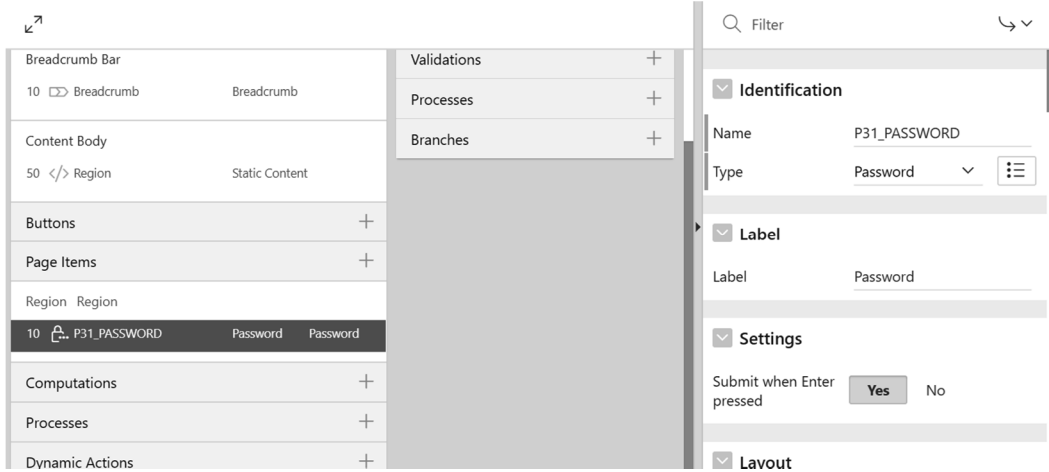
The User has only to select Yes when the password is needed in session state for use by other pages during the session.

Does not save state

This option helps the user if Select Yes is chosen, to suppress text entered into the field and not save the value in session state.

For security reasons user should always set this attribute to Yes.

Step 1: Create a page item and set the type as Password.



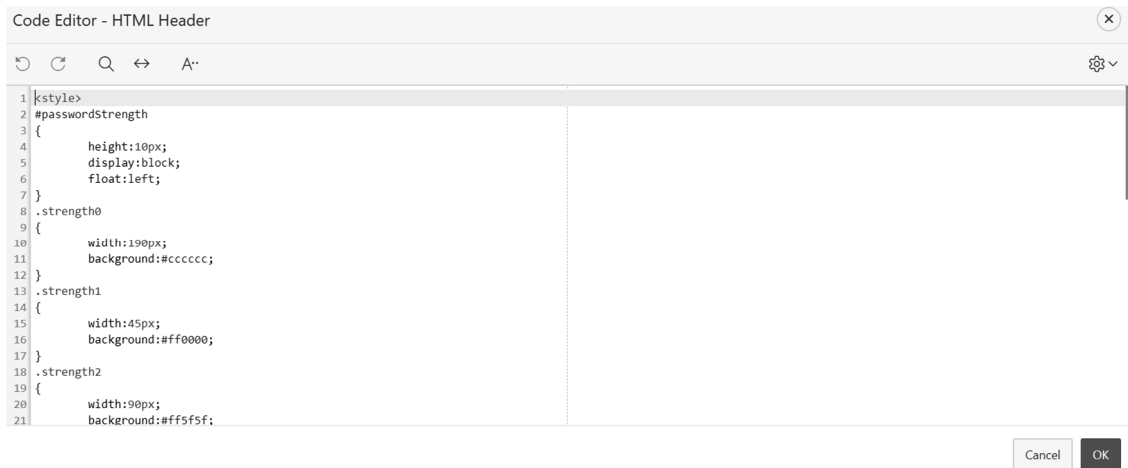
Password Field

A visual representation of the password field. It consists of a rectangular input box with a light gray border. Inside the box, the word 'Password' is displayed in a small font, followed by a small rectangular input area. Below the input box is a larger, empty rectangular area.

Customization – Password Strength Meter

Step 1: Create a Password Item as P31_PASSWORD, then Copy and Paste the below script in Page Inline or HTML Header

```
<style>
#passwordStrength { height:10px; display:block; float:left; }
.strength0{width:190px; background:#cccccc; }
.strength1{width:45px; background:#ff0000; }
.strength2{width:90px; background:#ff5f5f; }
.strength3{width:135px; background:#56e500; }
.strength4{background:#4dcd00; width:160px; }
.strength5{background:#399800; width:190px; }
</style>
```



Step 2: Copy and Paste the below script in HTML Header.

```
<script language="javascript">
function passwordStrengthmeter() {
var pwd = document.getElementById('P31_PASSWORD');
var password = document.getElementById("P31_PASSWORD").value;
var desc = new Array();
    desc[0] = "Very Weak";
    desc[1] = "Weak";
    desc[2] = "Better";
    desc[3] = "Medium";
    desc[4] = "Strong";
    desc[5] = "Strongest";
var score = 0;
if (password.length > 6) score++;
```


☒ **Advanced**

CSS Classes

Custom Attributes

Pre Text

Step 4: Go to the Post Element Text Of P31_PASSWORD Item and paste the following

```
<div id="passwordDescription">Password not entered</div>
<div id="passwordStrength" class="strength0"></div>
```

Post Text

```
<div id="passwordDescription">
&nbsp;&nbsp;&nbsp; Password not entered
</div>
<div id="passwordStrength"
class="strength0">
&nbsp;&nbsp;&nbsp;
</div>
```

Warn on Unsaved Changes

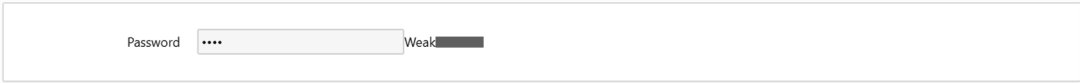
Result

Password Field

Password Password not entered

Weak Password

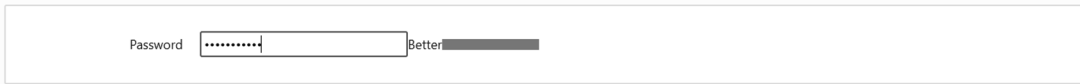
Password Field



A password field with the label "Password" and a text input containing four dots. To the right of the input is a strength indicator labeled "Weak" with a progress bar that is approximately 10% full.

Better Password

Password Field



A password field with the label "Password" and a text input containing eight dots. To the right of the input is a strength indicator labeled "Better" with a progress bar that is approximately 40% full.

Medium Password

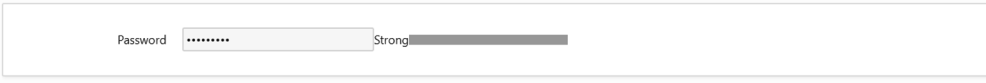
Password Field



A password field with the label "Password" and a text input containing ten dots. To the right of the input is a strength indicator labeled "Medium" with a progress bar that is approximately 60% full.

Strong Password

Password Field



A password field with the label "Password" and a text input containing twelve dots. To the right of the input is a strength indicator labeled "Strong" with a progress bar that is approximately 80% full.

Radio Group

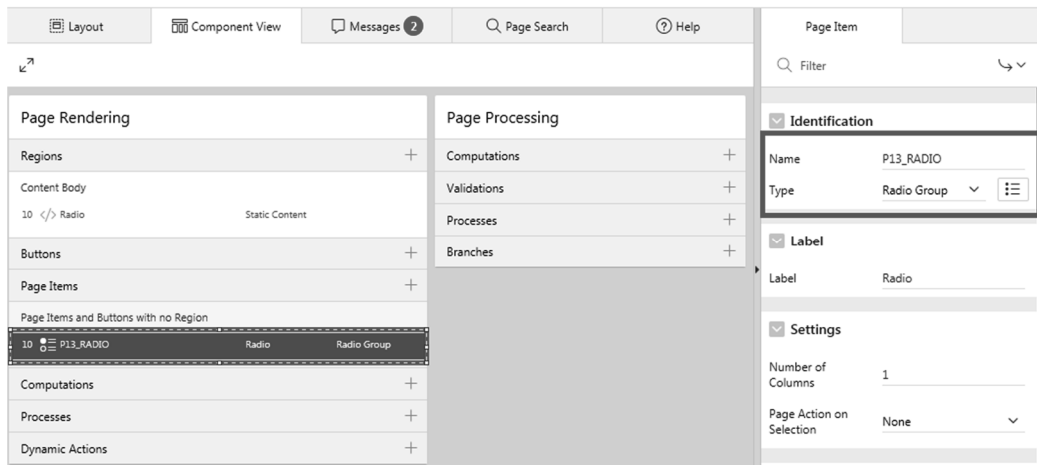
Radio group is a list of radio buttons, that provides users list of choices from where they can choose one value at a time. It is similar to select list option, but if there are less items in the list of values, then we can go for radio group. If there is a huge list of values then we can go for select list.

We can set the value for radio group by using LOV. In the LOV we can specify the display values and value of the page item will be set to when the radio button is selected.

Below we can see how to create a simple radio group with their key attributes, and based on the value selection in the radio button, how to display the corresponding images for it.

Creation of Simple Radio Group and key attributes

Create a new page item and set the type as radio group



Settings

Number of Columns

Enter the number of radio group columns to display. For example, a value of 2 would display two columns. If there were four values then it would display over two rows. In the below example we set as 1.

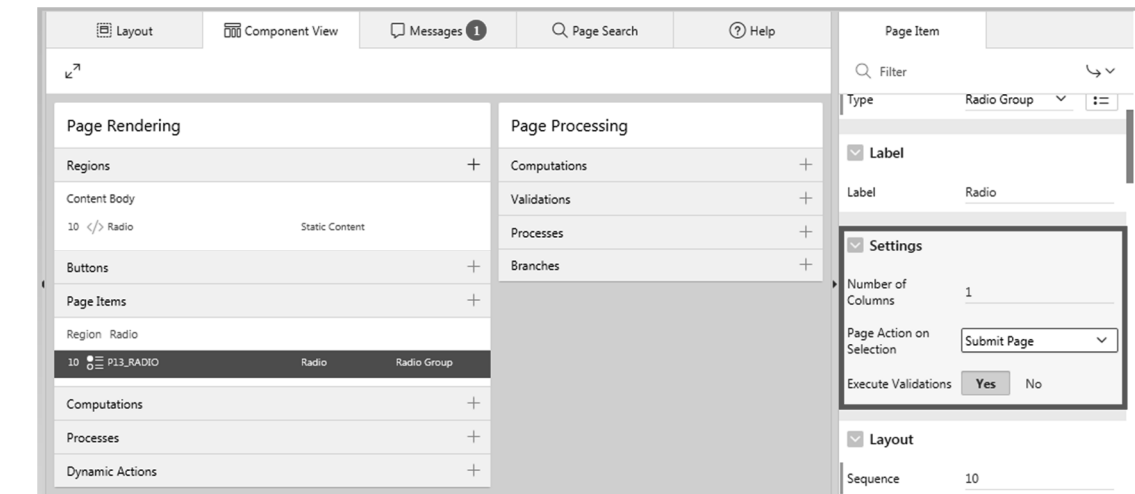
Page Action on Selection

Select what action is taken when a radio group value is selected.

Execute Validations -

Displays if Page Action when Value Changed is set to **Submit Page**. Controls the execution of validations when a page is submitted. Set this attribute to **No** to prevent the execution of validations and associated error messages. Set this attribute to **Yes** to execute all page and

built-in validations and associated error messages when the page is submitted. Use this attribute in conjunction with the validation attribute.



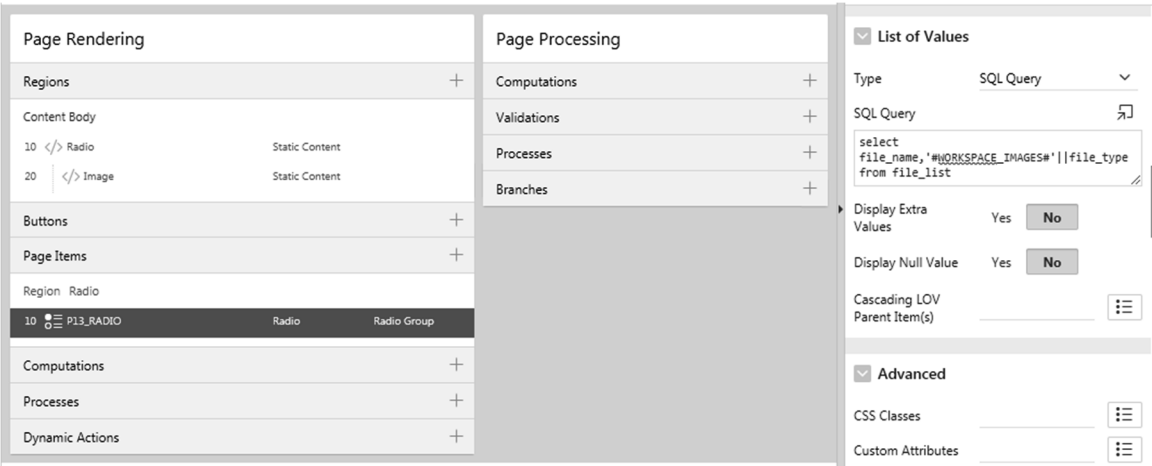
List of Values

Type

Select the source type for the radio Group. Now we have to choose the source type as “SQL Query”. Place the below code in SQL Query box.

```
select file_name,'#WORKSPACE_IMAGES#'||file_type from file_list;
```

The above query will display the file name and return the value as file_type.



Click Save and run the page

Output

Radio

Radio

☐ Hydrangeas

☐ Chrysanthemum

☐ Jellyfish

☐ Desert

Now we can see how to display the image based on the selected value in the radio button.

Create a new region as “Image” and set the Parent region as “Radio”.

Set Start new row as “No”.

Page Rendering

Regions

Content Body

10 </> Radio Static Content

20 </> Image Static Content

Buttons

Page Items

Region Radio

10 P13_RADIO Radio Radio Group

Computations

Processes

Dynamic Actions

Page Processing

Computations

Validations

Processes

Branches

Identification

Title Image

Type Static Content

Source

Text

Layout

Sequence 20

Parent Region Radio

Start New Row Yes No

Column Automatic

Upload the images in the Shared Component -> Static Workspace files.

Application 94584 \ Shared Components \ Static Workspace Files

Name	Mime Type	Size	Reference	File
Chrysanthemum.jpg	image/jpeg	859KB	#WORKSPACE_IMAGES#Chrysanthemum.jpg	Download
Desert.jpg	image/jpeg	826KB	#WORKSPACE_IMAGES#Desert.jpg	Download
Hydrangeas.jpg	image/jpeg	581KB	#WORKSPACE_IMAGES#Hydrangeas.jpg	Download
Jellyfish.jpg	image/jpeg	758KB	#WORKSPACE_IMAGES#Jellyfish.jpg	Download

1 - 4

Place the below code in the Image region.

```
<html>

</html>
```

Click save and run the page

Output

Radio

Radio

☒ **Hydrangeas**
☐ Chrysanthemum
 ☐ Jellyfish
 ☐ Desert

Image



We can also create a radio group function using APEX_ITEM.RADIOGROUP and below is the syntax for it.

```

APEX_ITEM.RADIOGROUP (
    p_idx          IN      NUMBER,
    p_value        IN      VARCHAR2 DEFAULT NULL,
    p_selected_value IN    VARCHAR2 DEFAULT NULL,
    p_display      IN      VARCHAR2 DEFAULT NULL,
    p_attributes   IN      VARCHAR2 DEFAULT NULL,
    p_onblur       IN      VARCHAR2 DEFAULT NULL,
    p_onchange     IN      VARCHAR2 DEFAULT NULL,
    p_onfocus     IN      VARCHAR2 DEFAULT NULL,
    p_item_id      IN      VARCHAR2 DEFAULT NULL,
    p_item_label   IN      VARCHAR2 DEFAULT NULL)
RETURN VARCHAR2;
```

For more information about the APEX_ITEM.RADIOGROUP and Radio Group Functionality you can refer the below links.

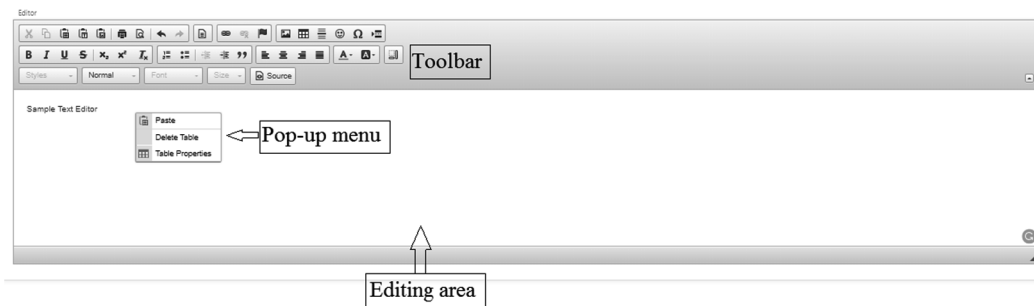
<https://docs.oracle.com/database/apex-5.1/AEAPI/RADIOGROUPFunction.htm#AEAPI204>

http://www.rampant-books.com/t_easy_html_db_radio_buttons.htm

Rich Text Editor

The rich text editor stretches the skill of a high edit box. It provides for the strong formatting of text content, including basic structural treatments such as lists, formatting treatments such as bold and italic text, and drag-and-drop inclusion and image sizing. It consists of these three distinct elements:

- ☐ Toolbar
- ☐ Editing area
- ☐ Pop-up menu



Toolbar

The toolbar, which is the area at the top of the editor, contains many different buttons that extend and activate a variety of editor functions. The toolbar is highly personalisable and can vary from field to field. The toolbar is collapsible. Click the triangle located next to the toolbar to toggle between visible and collapsed.

Editing Area

The editing area is directly below the toolbar. You enter text in this area.

Pop-up Menu

A pop-up menu is a contextual menu that appears when you right-click the mouse inside an editing area. The functions on the menu change based on what element, such as a table or an image, is under the cursor when you click the mouse.

Source:

https://docs.oracle.com/cd/E80738_01/pt854pbh2/eng/pt/tupa/task_WorkingWithRichTextEditorFields-c77ea3.html#topofpage

Creating Rich Text editor in Oracle apex is as simple as creating a simple text field.

Step 1: Create a static content region type.

Step 2: Create an Item either using Tree view or Component view.

Step 3: Change the type from Text to Rich Text editor.

Step 4: Click Save and Run a page to see the Rich text Editor.

To make Responsive editor

The Rich text editor is responsive in Oracle Apex 19.1. But in Oracle Apex 5.1, if you're using Universal Theme, then it requires the simple javascript to make the editor responsive. Let's see the example,

Step 1: Create a Rich text editor item using the above given steps. In my example, page item name is P17_EDITOR.

Step 2: In Right pane, you will be able to see the properties of Rich text editor and navigate to the "Javascript Initialization Code" attribute and place the below code,

```
function (o) {
    o.width = $("#P17_EDITOR").closest(".t-Form-inputContainer").width()
    - 5;
    o.height = 300; // Specify your desired item height, in pixels
    return o;
}
```

The above code defines the width of the region container of the item and subtracts 5 from it, and finally the output will initialise the size when the page is initially displayed. But

note, it won't handle the case when the browser is resized. To handle this, we need to create a dynamic action.

Step 3: Create a dynamic action as follows,

- ☐ Event : Resize
- ☐ True Action : Execute Javascript Code
- ☐ Code :

```
CKEDITOR.instances.P17_EDITOR.resize( $("#P17_EDITOR").closest(".t-Form-inputContainer").width() - 5, 300);
```

Finally the Rich text editor is responsive.

Source: <https://joelkallman.blogspot.com/2017/12/how-do-i-create-responsive-rich-text.html>

Configure Rich Text Editor with Javascript

The rich text editor is based on the Javascript CKEditor library. Page Designer does not offer too much configuration options, but the CKEditor library offers a huge variety of configuration options.

In Page Designer, go to the properties of Rich Text Editor and navigate to the “Javascript Initialization Code” attribute and then add the configuration options of CKEditor. A typical example for such a Javascript initialization function is as follows.

```
function ( configObject ) {
    configObject.{attribute-name-1} = {attribute-value-1};
    configObject.{attribute-name-2} = {attribute-value-2}; ...
    return configObject;
}
```

Let's see a simple example. The following javascript function will set the editor's base colour to green and we'll disable the resizing capability, so end users won't be able to resize the editor any more. After saving the function, reload the page to see the effects.

```
function ( configObject ) {
    configObject.uiColor          = "#AADC6E";
    configObject.resize_enabled = false;
    return configObject;
}
```

You can look into the below link to know the complete API that are available for configuring objects,

https://ckeditor.com/docs/ckeditor4/latest/api/CKEDITOR_config.html

Source: <https://blogs.oracle.com/apex/how-to-fully-leverage-the-rich-text-editor>

SELECT LIST

The page item Select list will display using a list of values based on some inputs. The input should be required to display for this. Select lists are rendered using the HTML form element `<select>`. The values in a select list are determined using a named list of values or a list of values defined at the item level. The user can also specify the NULL as display value and NULL as return value.

It consists of some settings attributes as follows:

Value Required

This option helps the user that if Select Yes chosen, and the page item is visible, Oracle Application Express automatically performs a NOT NULL validation when the page is submitted.

Page Action when Value Changed

This Option consists of some features like None, Submit Page, Redirect and Set Value, Redirect to Page (based on selected value), or Redirect to URL (based on selected value).

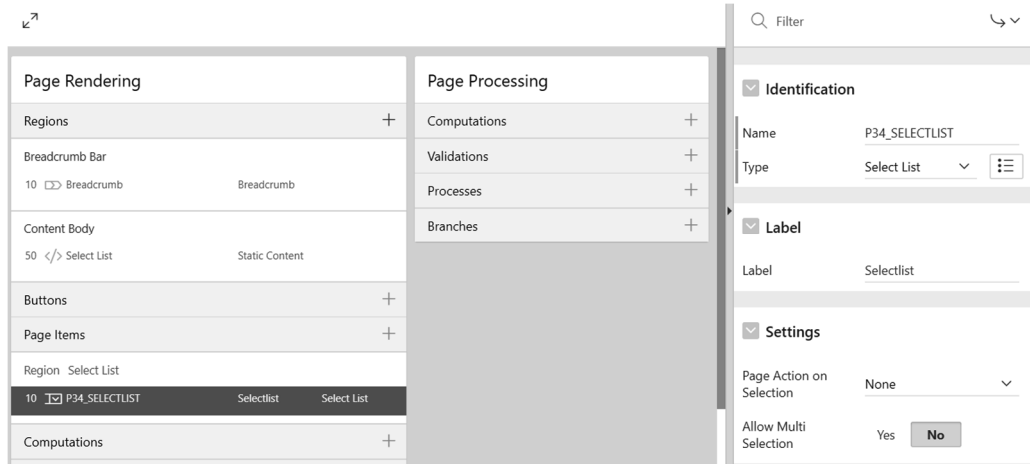
Allow Multi Selection

This option enables the user to select multiple values in the Select List. Use the Height attribute to define how many entries of the Select List should be visible at once.

Execute Validations

This option helps to display if Page Action when Value Changed is set to Submit Page. It controls the execution of validations when a page is submitted. Set this attribute to No to prevent the execution of validations and associated error messages. Set this attribute to Yes to execute all page and built-in validations and associated error messages when the page is submitted.

Step 1: Create a page item and select type as “Select list”.



Step 2: Go to List of Values Section. It consists of four types as

- ☐ Shared Component
- ☐ SQL Query
- ☐ Static Values
- ☐ PLSQL Function Body Return SQL Query

Shared Component

List of Values

Type

Shared Component

List of Values

LOGIN_REMEM

Display Extra Values

Yes

No

Display Null Value

Yes

No

Null Display Value

Null Return Value

SQL Query



▼

List of Values

Type

SQL Query

▼

SQL Query

↗

select dname as d, deptno as r
from dept
order by 1 asc

Display Extra Values

Yes

No

Display Null Value

Yes

No

Null Display Value

Static Values

Static Values

Values

Display Value	Return Value		
Display1	Return1	X	↓
Display2	Return2	X	↑
		X	↑

Sort

Cancel

OK

List of Values

TypeStatic Values

Static ValuesDisplay1, Display2

Display Extra Values

YesNo

Display Null ValueYesNo

Null Display Value

Null Return Value

Cascading LOV Parent Item(s)

PLSQL FUNCTION Body Return SQL Query

Code Editor - PL/SQL Function Body returning SQL Query

Validation successful

```
1 DECLARE
2   l_query VARCHAR2(400);
3 BEGIN
4   l_query := 'SELECT DNAME AS D,DEPTNO AS R FROM DEPT';
5   RETURN l_query;
6 END;
```

▼ List of Values

Type

PL/SQL Function Body

PL/SQL Function Body returning SQL Query

```
DECLARE
  l_query VARCHAR2(400);
BEGIN
  l_query := 'SELECT DNAME AS D,DEPTNO
AS R FROM DEPT';
  RETURN l_query;
END;
```

Display Extra Values

Yes

No

Result

Select list- Page Item

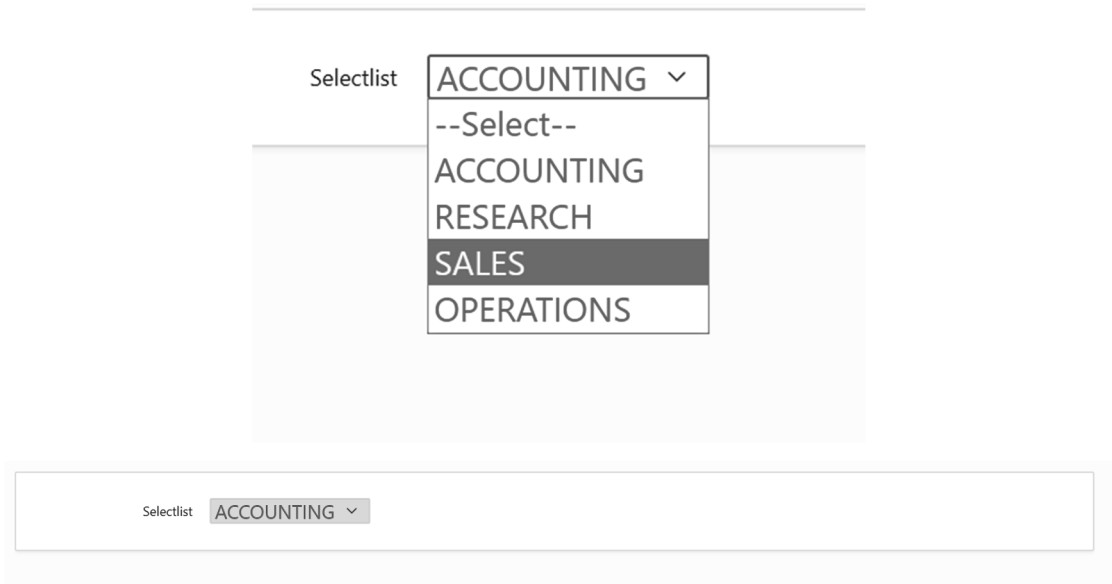
Selectlist

--Select--

Customization

Copy and paste the below script to change the colour, background colour, and font style.

```
.t-Form-inputContainer select.selectlist, .t-Form-inputContainer
select.yes_no, .t-Form-select {
    color: red;
    border-color: #dfdfdf;
    background-color: aqua;
    font-size: large;
}
```



Shuttle

The Shuttle item allows the user to select one or more values. The values listed in the left side are the possible values from the source that are available for the user selection. The selected values will be listed in the right side of the item. The value selection can be done by doing double click on the values in the left side. It also provides the toolbar placed in the middle which includes features such as Reset, Move all, Move, Remove and Remove all. Let's see how to create a simple shuttle item.

Creating Shuttle in Oracle apex is as simple as creating a simple select list field.

Step 1: Create a static content region type.

Step 2: Create an Item either using Tree view or Component view.

Step 3: Change the type from Text to Shuttle.

Step 4: Navigate to List of values and place the select query as given below,

```
select ename d, empno r from emp
```

Step 5: Click Save and Run a page to see the Shuttle.



Customization in Shuttle

The Shuttle is really helpful when there is a multi-selection required. But it becomes complex when it has a lot of values as the possible value from source. To overcome this problem, I came across Jari Laine's blog where he added the filter to the shuttle.

Let's see how to achieve adding filter to the Shuttle item in Oracle Apex 19.1.

Step 1: Create a Shuttle item using the above given steps. In my example, page item name is P18_SHUTTLE.

Step 2: Now on the same page, navigate to “Function and Global Variable Declaration” under Javascript in the page properties and add the code given below,

```
(function ($) {
$.fn.htmlDbShuttlefilter=function(options){
  options=$.extend({},{"label":"Filter"},options);
  return this.each(function(i){
    var $self      = $(this)
    ,filterId      = $self.attr("id") + "_FILTER"
    ,$select       = $self.find("select")
    ,shuttleValues = $select.children("option").map(function(){
      return {text:$(this).text(),value:$(this).val(),option:this}
    })
    ,$filter =
    $("<input/>,{\"type\":\"text\",\"value\":\"\",\"size\":\"255\",\"autocomplete\":\"off\",
    \"id\":filterId})
    .keyup(function(){
      var filterval  = new RegExp("^"+$(this).val()+".*","i")
```



```

,selectedValues = $select.eq(1).children("option").map(function() {
    return $(this).val();
});
$select.eq(0).empty();
$.each(shuttleValues,function(idx,obj){
    if(obj["text"].match(filterval) &&
$.inArray(obj["value"],selectedValues)<0){
        $select.eq(0).append(obj["option"]);
    }
});
})
.width($self.width());
$("<div/>",{ "css":{ "padding-bottom": "5px" } })
.insertBefore($self)
.append(
    $("<label/>",{ "for":filterId})
    .append($("<span/>",{ "css":{ "font-
weight": "bold" } }) .text(options.label))
)
.append("<br/>").append($filter);
$self.find("img[alt='Reset']").click(function(){ $filter.val(""); });
});
}
})(jQuery);

```

Step 3: Then add the code given below in the “Execute and Page loads” section available below to the “Function and Global Variable Declaration” section.

```

$("#P18_SHUTTLE").htmlDbShuttlefilter({});
$(".t-Form-itemWrapper").css("display","inline");

```

Step 4: Click Save and Run the Page.

Final result will be as shown in the below image.

The screenshot shows a web interface with a search filter at the top. Below the filter is a table with two columns. The first column contains the names 'SCOTT' and 'SMITH'. The second column is empty. To the right of the table are navigation buttons: a search icon, a refresh icon, and arrows for navigating between pages (first, previous, next, last).

Source:

<https://jaris.tilaa.cloud/apex/f?p=BLOG:READ:::NO::ARTICLE:201410131041380630>

TEXT FIELD

The page item Text field is displayed as an HTML text field containing a maximum of 30,000 bytes of text. It also includes the option to control the maximum length and display width by editing the Height and Width item attribute.

It consists of some Available Text display options as below:

Text Field

This option defines that it will render as a text field.

Text Field (Disabled, does not save state)

This option helps to display a read-only version of a display value from a list of values by using the item's value in session state to look up the corresponding display value in the associated list of values. The value displayed on the screen is not saved in session state upon submit.

Text Field (Disabled, saves state)

This option helps to display a read-only version of a display value from a list of values by using the item's value in session state to look up the corresponding display value in the associated list of values.

Text Field (always submits page when Enter pressed)

This option helps to display a read-only version of the value in session state. Upon choosing submit, the value displayed is saved in session state.

Text Field with Calculator Popup

This option will render as a text field with an icon next to it. When clicked, the icon displays a small window containing a calculator. Calculations are placed back in the text field.

Step 1: Create a page item and set type as “Text field”

The screenshot shows the configuration interface for a new page item. It is divided into three sections: Identification, Label, and Settings.

- Identification:** Contains fields for Name (P34_TEXT_FIELD) and Type (Text Field). There is a menu icon to the right of the Type field.
- Label:** Contains a field for Label (Text Field).
- Settings:** Contains fields for Subtype (Text) and Trim Spaces (Leading and Trailing). Both have dropdown arrows.

Result

The screenshot shows the final result of the configuration. It displays a breadcrumb trail with the text "Select list- Page Item \ Breadcrumb". Below the breadcrumb, there is a text field labeled "Text Field" with a small input box.

CUSTOMIZATION**Text Box Value to UPPER/LOWER Case**

Navigate to the page where the changes are required. Click on the field/Item for which all capital characters are required. Scroll till you reach Element section and in "HTML Form Element Attributes" paste the below code

```
onKeyUp="this.value=this.value.toUpperCase();"
onChange="this.value=this.value.toLowerCase();"
```

Now Click on Apply Changes button and run the page. For the field this attribute is applied, whenever a key is typed the case is changed. But if you don't want this type of animation and just want all characters to be in upper case,

```
style="text-transform: uppercase;"
onKeyUp="this.value=this.value.toUpperCase();"

```

Text Box Value allows only numbers

Steps: Copy and paste the below code in Page HTML Header

```
<script>
function isNumberss(event) {
    if (event) {
        var charCode = (event.which) ? event.which : event.keyCode;
        if (charCode != 190 && charCode > 31 &&
            (charCode < 48 || charCode > 57) &&
            (charCode < 96 || charCode > 105) &&
            (charCode < 37 || charCode > 40) &&
            charCode != 110 && charCode != 8 && charCode != 46 )
            return false;
        }
        return true;
    }
}
</script>

```

Navigate to the page where the changes are required. Click on the field/Item for which all capital characters are required. Scroll till you reach Element section and in "HTML Form Element Attributes" paste the below code

```
onkeydown="return isNumberss(event);"
```

Restrict Special Characters and Spaces in Textbox

Copy and paste the below code in Page HTML Header Section

```
<script type="text/javascript">
    function avoid_special(event){
    if(!((event.keyCode >= 65) && (event.keyCode <= 90) || (event.keyCode >=
    97) && (event.keyCode <= 122) || (event.keyCode >= 48) && (event.keyCode
    <= 57))) {
        event.returnValue = false;
        return;
    }
    event.returnValue = true;
    }
</script>

```

Navigate to the page where the changes are required. Click on the field/Item for which all capital characters are required. Scroll till you reach Element section and in "HTML Form Element Attributes", paste the below code

```
onkeypress="return avoid_special(event) "
```

Text field with Auto Complete

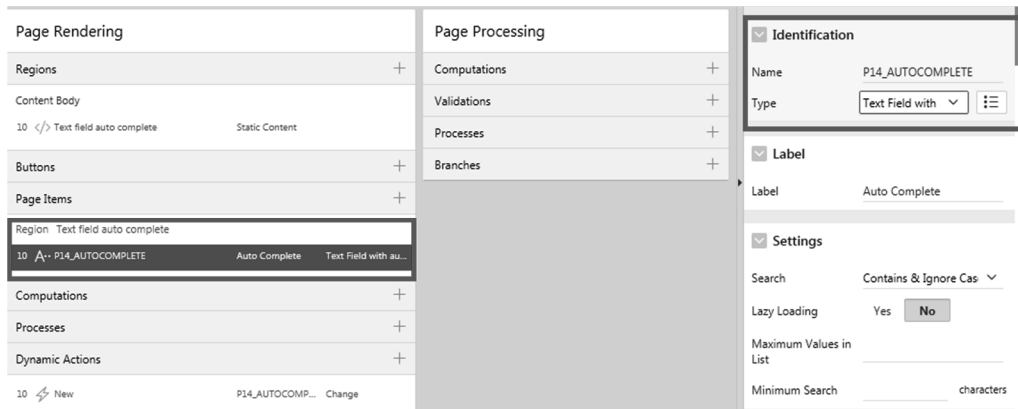
Text field with auto complete displays a list of values, based on the LOV, and user can also enter the required text for the processing that are not on the list. Whenever user types a character, it will list all the possible words from LOV.

It is similar to select list. In select list, we are not able to type and search. But in text with auto complete we can search the values by entering the text, and if we have a very huge list, then text with auto complete will be very handy.

While giving the LOV Query, in select list we will provide display value and return value, but in Auto complete we will just provide the display value only.

In the screenshots given below, we will see how we can create an Auto Complete text field and its key attributes.

Create a new Item, with type “Text field with auto complete”.



Page Designer item attributes

Settings

Search – Select how the search against the entered value is performed. When using Lazy Loading this setting can have an impact on query performance. Selecting **Contains & Case Sensitive** or **Contains & Ignore Case** prevents the use of column indexes.

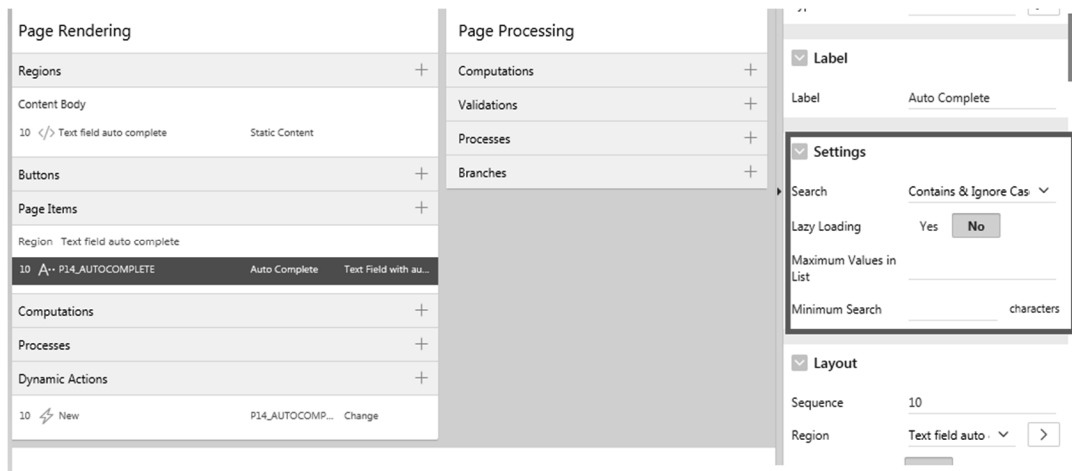
Lazy Loading – Specify whether matching values are retrieved from the database using Lazy Loading techniques. Lazy loading retrieves matching records from the database each time the user types a character. Otherwise, all of the list values are loaded on page load. Oracle recommends using Lazy Loading when you have a large select list, as loading a large number of records may impact the time to initially load the page.

Maximum Values in List – Enter the maximum number of items to display in the select box. This value can help avoid performance problems by returning too many rows, and also prevent a very large return set when only a few letters have been entered.

Minimum Search Character – Enter the minimum number of characters to search in the select box. E.g. If we set the value as 2, then the searching will be performed only after the second character has been entered.

Validation

Value Required – If select **Yes** and the page item is visible, Oracle Application Express automatically performs a NOT NULL validation when the page is submitted. See Help for more details.



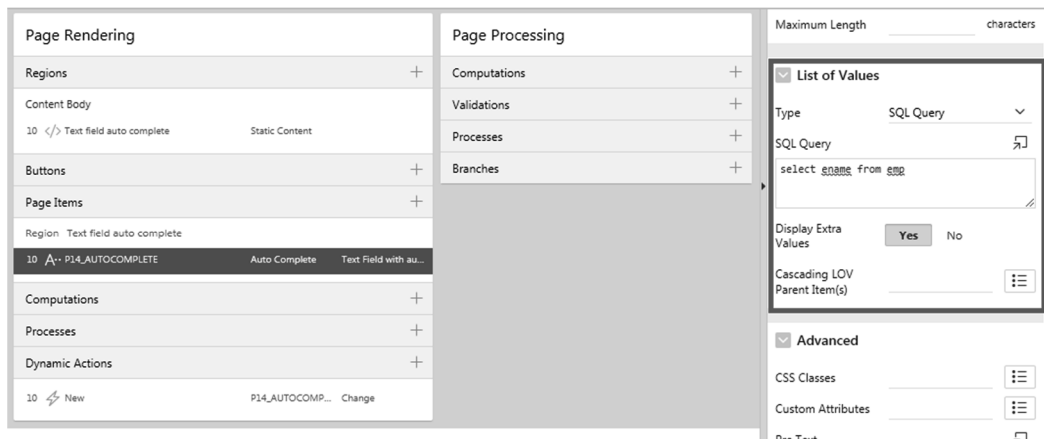
List of Values

Type:

Select the source type for the Text field with Auto Complete. Now we have chosen the source type as “SQL Query”.

Place the below code in SQL Query box.

```
SELECT ename
FROM emp;
```



Click Save and Run the Page.

Output

The screenshot displays an Oracle APEX application interface. On the left is a dark sidebar with a navigation menu. The main content area is titled 'Text field auto complete'. It features a text input field with the letter 'a' entered. Below the input field is a dropdown list showing the following names: BLAKE, CLARK, ALLEN, WARD, MARTIN, ADAMS, and JAMES.

Reference : <https://docs.oracle.com/en/database/oracle/application-express/19.1/htmldb/about-item-types.html#GUID-8CAA1BA5-DCA8-4DF0-A841-3DC40E67CA9C>

Working with Textarea

A textarea can hold a multi-line text input and there is no maximum length for an item. You can control the height and width by editing the Height, Width and Maximum width attributes in Oracle Apex. Creating Textarea in Oracle apex can be done using two ways by Creating Item/using Textarea function. Let's see how to create a simple Textarea in the html region.

Step 1: Create a static content region type.

Step 2: Create an Item either using Tree view/ Component view.

Step 3: Change the type from Text to Textarea.

Step 4: Navigate to setting attribute to adjust textarea style like width, height and so on.

Step 6: Click Save and Run a page to view the created textarea.

Customization in Textarea

Set Auto height

There is an Auto height option in the textarea which works perfectly in IE as expected. But it is not working in other browsers. So this can be fixed by adding simple javascript code. Let's see it in the following steps,

Step 1: Create a Textarea item using the above steps. In my example, page item name is P22_TEXTAREA.

Step 2: Click on Page header and navigate to Javascript section. Place the below given code in the "Function and Global Variable Declaration"

```
function adjust_height(element) {
    element.style.height = (element.scrollHeight)+"px";
}
```

Step 3: Then go to Execute when Page Loads and place code shared below,

```
$("#P22_TEXTAREA").attr("oninput", "auto_grow(this)");
$("#P22_TEXTAREA").css("height", "5px");
```

That's it, now textarea will adjust its height based on the input given to it.

Add Background Image

Here we are going to see adding css to generate a highlighting effect within a textarea. Below example shows you how to change a simple textarea to look like a piece paper effect. The effect is complemented by the use of the old-fashioned typewriter-style font.

Step 1: Create a Textarea item using the above steps. In my example, page item name is P22_TEXTAREA.

Step 2: Click on Page header and navigate to CSS section and place the below given code in the "Inline" attribute,

```
#P22_TEXTAREA {
    width:500px; overflow:hidden; background-color:#FFF;
    color:#222; font-family:Courier, monospace; font-weight:normal;
    font-size:24px; resize:none; line-height:40px; padding-left:100px;
    padding-right:100px; padding-top:45px; padding-bottom:34px;
    background-
    image:url(https://static.tumblr.com/maopbtg/E9Bmgtoht/lines.png),
```

ORACLE APEX

```
url (https://static.tumblr.com/maopbtg/nBUmgtogx/paper.png);  
background-repeat:repeat-y, repeat; -webkit-border-radius:12px;  
border-radius:12px; -webkit-box-shadow: 0px 2px 14px #000;  
box-shadow: 0px 2px 14px #000; border-top:1px solid #FFF;  
border-bottom:1px solid #FFF;  
}
```

Step 3: Click Save and Run a page.

The textarea will be like the image shown below,



Source : <https://codepen.io/levchenkod>

BUTTONS

Button in Oracle apex can perform four types of actions like submitting a page, redirect to another page in the application, add custom URL and define dynamic action. Creating Button in Oracle apex can be done either using Component view or Tree View. Refer the following link to create a simple button,

https://docs.oracle.com/database/121/HTMDB/bldapp_but.htm#HTMDB25438

Adding css to the button

If you are looking to attract the user's attention, then following content will give you basic idea of adding css to enhance the look and feel of button on screen. Let's see the following topics in detail,

- ☐ Convert text link to button in Report
- ☐ Glow buttons with CSS on hover
- ☐ Change shape of the button on hover

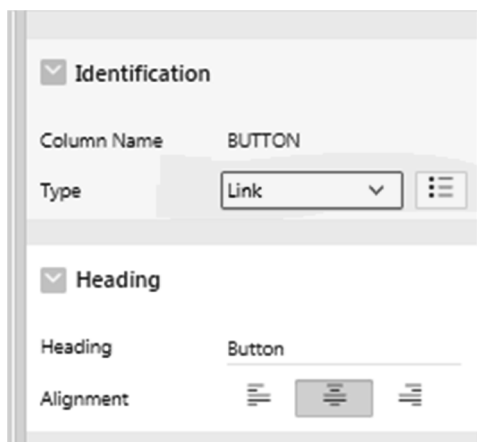
Convert text link to button in Report

Creating button in Region is straight forward and also simple. Let see how to achieve adding button into each row of the Report, either Interactive Report or Classic Report.

- ☐ Create a region and select type as Classic Report.
- ☐ Select source type as SQL Query and place your code.
- ☐ Sample code:

```
SELECT dname department, ename Name , job , sal Salary, NULL LINK
FROM emp e, dept d
WHERE e.deptno = d.deptno
order by 1, 2
```

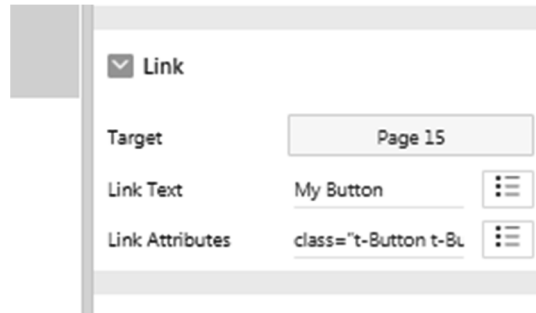
Go to column Link in the report column section (left pane of the page designer) and change the type to Link (right pane of the page designer).



Then scroll down to link section (right pane of the page designer). Place css code as seen in the below screen shot,

ORACLE APEX

Link attribute : `class="t-Button t-Button--hot"`



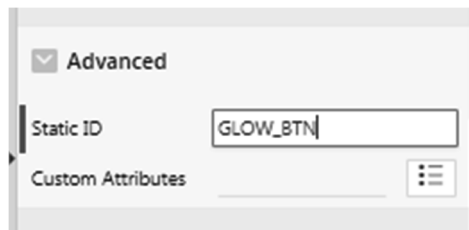
Source: <https://apex.oracle.com/pls/apex/f?p=42:6100:::NO:::>

Note: Same step is also applicable for Interactive report.

Glow buttons with CSS on hover

Adding effects to the items in the screen will always attract the user to your website or application. Let's see how to do the same in Oracle Apex.

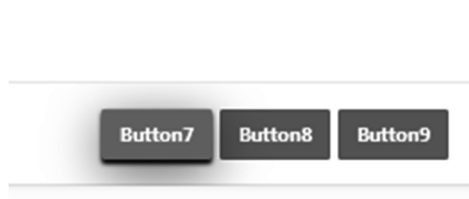
- ☐ Create a simple button on a page.
- ☐ Assign a static id to it which can be seen under Advanced section of button attribute.



- ☐ Click page header and go to css inline section. Then place the following code in it.

```
#GLOW_BTN:hover { transform: scale(1);  
    box-shadow: 0 2px 4px #000, 0 8px 32px rgba(0, 0, 0, 0.5); }
```

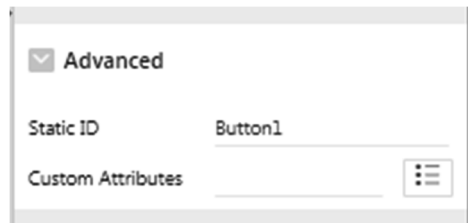
- ☐ Click Save.
- ☐ Now button on page looks as shown in the below figure on hover.



Change shape of the button on hover

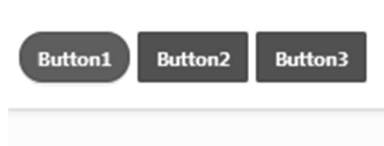
To change the shape of your button, you can apply a simple CSS. The following steps will enable you to achieve it.

- ☐ Create a simple button on a page.
- ☐ Assign a static id to it which can be seen under Advanced section of button attribute.



- ☐ Click page header and go to css inline section. Then place the following code in it.
- ```
#Button1:hover {-webkit-border-radius: 25px; -moz-border-radius: 25px; border-radius: 25px; }
```
- ☐ Click Save.

Now button on page looks as shown in the below figure on hover.



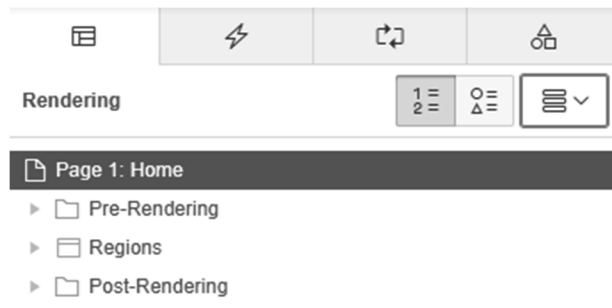


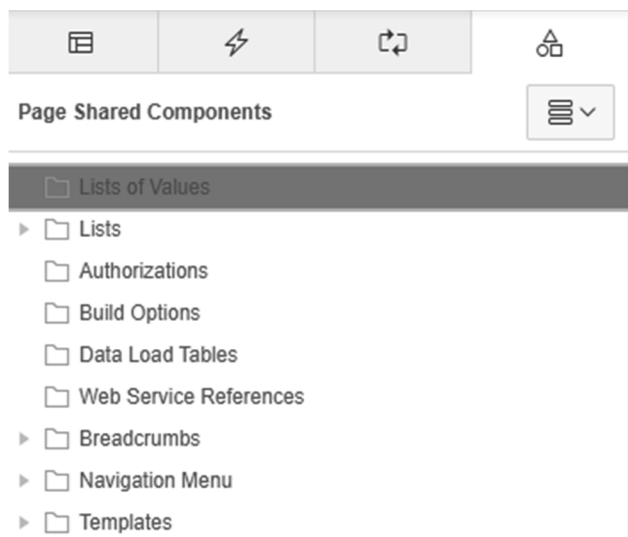
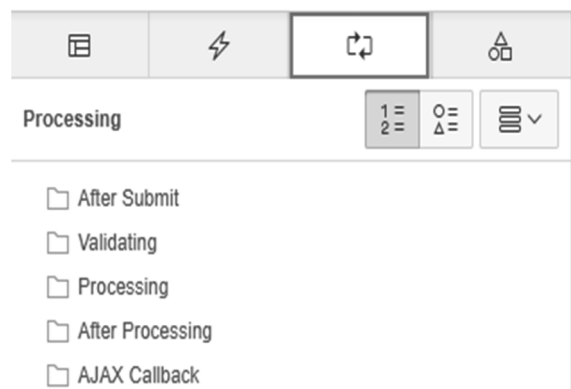
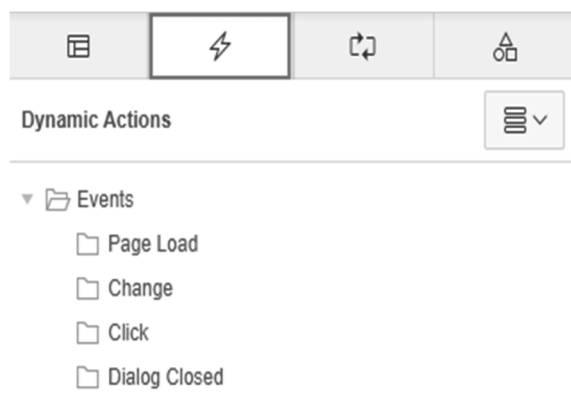
## PAGE DESIGNER

---

The Page Designer window features three panes: a Left pane, a Central pane, and a Right pane.

Usually we segregate this as Page rendering, Page Processing and Shared components. This was the case before 5.0 version. In later versions Oracle has introduced page layout designer which enables drag and drop option for creating page components which was taking a bit longer without this option. The left pane enables the developer to add page components and dynamic action for all the page events and page processing at different stages of page action.







The last but not least, we have the shared component list which is being used in the page and can be edited from the page itself.

## **Centre Pane**

### **Grid Layout**

The grid layout is the place where the developer can place the page components in the different page sections, like after header, navigation, breadcrumb, etc., and rearrange by dragging it to the desired page, which makes the task easy and fast, enabling the developer to concentrate on the other important sections like security, instead of spending time in arranging the items with html codes.

### **Message**

The message section was introduced in 18 version, where instead of having errors one after the other, the developer can see all the error messages at once.

### **Page Search**

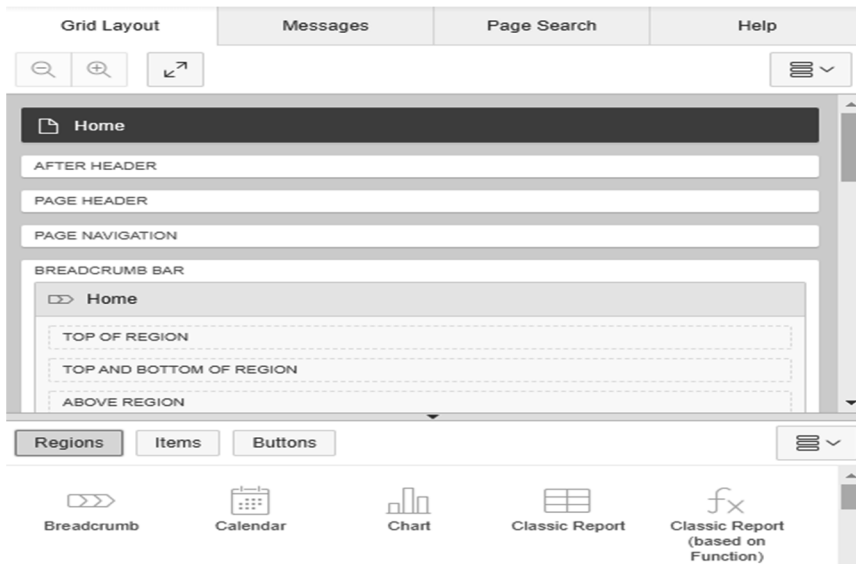
The page search enables the developer to search the page instantly. Say, the developer wants to know whether the particular table is used in the page or not. Assume that this feature is not available, the developer needs to open each and every component in the page to see whether the table is available or not.

### **Help**

This section tells the developers what is the selected component about and what are the details of the options in the component.

### **Right Pane**

This pane is the heart of the component, and this enables the developers to change the attributes of the page components.



## SHARED COMPONENTS

---

### APPLICATION LOGIC

The Application Logic is one of the modules in Shared components. It consists of different types of features as below:

- ☐ Application Definition Attributes
- ☐ Application Items
- ☐ Application Process
- ☐ Application Computations
- ☐ Application Settings
- ☐ Build options

### Application Definition Attributes

The Application Definition attributes consist of main four types as

- ☐ Definition
- ☐ Security
- ☐ Globalization
- ☐ User Interface

### Definition

This option helps the user to Name to define basic characteristics of your application, including the application name, an optional alphanumeric alias, and a version number.

It consists of some following types as follows:

- ☐ Name
- ☐ Properties
- ☐ Application Icon
- ☐ Availability
- ☐ Error Handling
- ☐ Global Notification
- ☐ Substitutions
- ☐ Build Options

Name

Name

Application

103274

\* Name

JeganDev-Apex(Run)

Application Alias

103274

\* Version

Release 1.0

Application Group

- Unassigned -

Properties

Properties

Logging

Yes

Debugging

Yes

Allow Feedback

Yes

No

Compatibility Mode

19.1

Application Email From Address

Proxy Server

Application Icon

Application Icon

Icon File Name

app-icon.svg

## Availability

Availability

Status

Available with Developer Toolbar

?

Build Status

Run and Build Application

?

Message for unavailable application

This application is currently unavailable at this time.

?

Restrict to comma separated user list (status must equal Restricted Access)

?

## Error Handling

Error Handling

Default Error Display Location

Inline with Field and in Notification

?

Error Handling Function

?

## Global Notification

Global Notification

Message to be displayed in page #GLOBAL\_NOTIFICATION# substitution string

?

## Substitutions

Substitutions

Use static substitution strings for phrases or labels that occur in many places within an application. To create a substitution string, enter the string name in the Substitution String column and the string value in the Substitution Value column.

| Substitution String | Substitution Value |
|---------------------|--------------------|
| APP_NAME            | JeganDev-Apex      |
|                     |                    |
|                     |                    |

## Build Options

Build Options

No build options defined.

### Security

This option helps the user to use this page to set application-wide security settings. Edit application components directly to manage more granular settings.

It consists of some following features as below:

#### Authentication

Authentication is the process of establishing each user's identity before they can access your application. You may define multiple authentication schemes for your application, however only one scheme can be current. The authentication logic of the current scheme is used when your application is run.

#### Authorization

Application authorization schemes control access to all pages within an application. Unauthorized access to the application, regardless of which page is requested, will cause an error page to be displayed.

#### Session Management

This option helps to control the application session management.

#### Session State Protection

Enabling Session State Protection can prevent hackers from tampering with URLs within your application. URL tampering can adversely affect program logic, session state contents, and information privacy.

To enable Session State Protection for your application, select Enabled from the Session State Protection list. Enabling Session State Protection turns on session state protection controls defined at the page and item level. To configure Session State Protection, click Manage Session State Protection.

Browser Security

This option helps to control the application Browser Security Management.

Database Session

This option helps to control the application Database Session management.

Authentication

Application103274?

Public UserAPEX\_PUBLIC\_USER?

Authentication SchemeApplication Express Authentication?

Authorization

Authorization Scheme- No application authorization required - ?

Run on Public PagesYesNo?

Source for Role or Group SchemesAccess Control User Role Assignments?

Session Management

Session Management

Rejoin SessionsDisabled?

Deep LinkingDisabled?

Maximum Session Length in Seconds?When Maximum Session Length is not set, the workspace level or instance level value will be used (currently, 43200 seconds).

Session Timeout URL?

Maximum Session Idle Time in Seconds?When Maximum Session Idle Time is not set, the workspace level or instance level value will be used (currently, 7200 seconds).

Session Idle Timeout URL?

## Session State Protection

Session State Protection 

Enabled

 ?

Allow URLs Created After **(null)** ?

Bookmark Hash Function **SHA-2, 512 bit (requires 12c)** ?

## Browser Security

Browser Security

Cache 

Disabled

 ?

Embed in Frames 

Deny

 ?

HTML Escaping Mode 

Extended

 ?

HTTP Response Headers  ?

## Database Session

Database Session

\* Parsing Schema 

KIRTHIK

 ?

Initialization PL/SQL Code  ?

Cleanup PL/SQL Code  ?

Runtime API Usage 

☐ Modify This Application

☐ Modify Other Applications

☐ Modify Workspace Repository

 ?

## Globalization

This option helps the user to identify the globalization attributes of your application. It consists of application translations.



Globalization

Application Primary Language

English (en)

Application Language Derived From

Application Primary Language

Document Direction

Left-To-Right

Application Date Format

Application Date Time Format

Application Timestamp Format

Application Timestamp Time Zone Format

Character Value Comparison

Character Value Comparison Behavior

Database session NLS setting (default)

Automatic Time Zone

Yes

No

Automatic CSV Encoding

Yes

No

No translations found.

User Interface

This option helps the user to change the UI Settings of the application.

It consists of following types as:

- ☐ User Interfaces
- ☐ General Properties
- ☐ Logo
- ☐ Favi-Icon
- ☐ User Interface Detection

User Interfaces

| User Interfaces                                                                     |         |                                                                                     |         |             |             |                      |
|-------------------------------------------------------------------------------------|---------|-------------------------------------------------------------------------------------|---------|-------------|-------------|----------------------|
|                                                                                     | Name    | Type                                                                                | Default | Auto Detect | Global Page | Theme                |
|                                                                                     |         |                                                                                     |         |             |             | Theme Style          |
|  | Desktop |  | ✓       |             | 0           | Universal Theme - 42 |
|                                                                                     |         |                                                                                     |         |             |             | Vita                 |

## General Properties

General Properties

Static File Prefix

?

Image Prefix

?

Media Type

?

## Logo

Logo

Logo Type:

Image

☒ Text

?

Logo:

JeganDev-Apex

?

Logo Attributes:

⋮

?

## Favi-Icon

Favicon

Favicon HTML

?

## User Interface Detection

User Interface Detection

CSS File URLs

?

## Application Items

The User can use application Items to maintain session state. Application Items can be set using computations, processes, or by passing values on a URL. Use "On New Instance" computations to set the value of items once for a session. Use Application Items to maintain session state that is not displayed and is not specific to any one page.

It consists of the following types as

☐ Name

Name

Application: 103274 JeganDev-Apex(Run) ?

\* Name  ?

\* Scope  ?

☐ Security

Security

Session State Protection  ?

☐ Configuration

Configuration

Build Option  ?

☐ Comments

Comments

Comments

?

## Application Process

Application Processes run PL/SQL logic at specific points for each page in an application or as defined by the conditions under which they are set to fire. Note that "Ajax Callback" processes fire only when called from Ajax or by "Run Ajax Callback" processes defined for pages. It consists of some following types as

Name

Name

Application:

103274 JeganDev-Apex(Run) ?

\* Sequence

1 ?

\* Process Point

Ajax Callback: Run this application process when requested by a page process. ?

\* Name

DOWNLOADPDF ?

Type

PL/SQL Anonymous Block ?

Source

Source

\* PL/SQL Code ?

1

begin

2

for file in (select \*

3

from apex\_application\_temp\_files

4

where id = :FILE\_ID) loop

5

--

6

sys.http.init;

7

sys.owa\_util.mime\_header( file.mime\_type, FALSE );

8

sys.http.p('Content-length: ' || sys.dbms\_lob.getlength( file.blob\_content));

9

sys.http.p('Content-Disposition: attachment; filename="' || file.filename || '"');

10

sys.http.p('Cache-Control: max-age=3600'); -- tell the browser to cache for one hour, adjust as necessary

11

sys.owa\_util.http\_header\_close;

12

sys.wpg\_docload.download\_file( file.blob\_content );

13

apex\_application.stop\_apex\_engine;

14

end loop;

15

end;

16

☐ Do not validate PL/SQL code (parse PL/SQL code at runtime only).

Conditions

Conditions

Condition Type

- Select Condition Type -

?

PL/SQL item / column=value item / column not null item / column null request=e1 exists never none

Authorization

Authorization

Authorization Scheme

Must Not Be Public User

?

Configuration

Configuration

Build Option

- No Build Option -

?

Comments

Comments

Comments

?

Application Process

The user can use Application Computations to assign values to application and page items.  
It consists of below types as

Item

Item

Application: 103274 JeganDev-Apex(Run) ?

Sequence

10

?

\* Computation Item

⋮

?

Frequency

Frequency

Computation Point

On New Instance (new session)

?

Computation

Computation

Computation Type Static Assignment ?

\* Computation

Computation Error Message

Authorization

Authorization

Authorization Scheme - No Authorization Required - ?

Conditions

Conditions

Condition Type - Select Condition - ?

PL/SQL item / column=value item / column not null item / column null request=e1 exists never none

Configurations

Configuration

Build Option - No Build Option - ?

Comments

Comments

Comments

## Application Settings

The Application Settings enable developers to define application level configuration options.

You can use the following APIs within your application to access your application settings:

To set a value for a setting: `apex_app_setting`.

```
set_value() .
```

To get the value of a setting: `apex_app_setting`.

```
get_value() .
```

×

Attributes

Application: **103274 JeganDev-Apex(Run)**

\* Name  ?

Value  ?

Value Required Yes **No**

Valid Values  ?

On Upgrade Keep Value Yes **No** ?

Comments

Cancel

Create Application Setting

## Build Options

The Build options enable developers to conditionally enable and disable application components and functionality when the application is installed or at run time using the `GET_BUILD_OPTION_STATUS` and `SET_BUILD_OPTION_STATUS` APIs.

The user can apply build options to most components (such as pages, regions, items, validations, and so on) to include or exclude them in the run time application:

## **Creating Build Options**

This option helps the user to create build options by selecting the application and accessing Build Options on the Shared Components page.

## **Including or Excluding Build Options**

The user can specify Include to enable a component and include it with the application. Specify Exclude to disable a component and exclude it from the application.

## **Selecting a Build Option**

This option helps the user to apply build options to a page, component, page control, or shared component in Page Designer.

## **Deleting Build Options and Associated Components**

It helps the user to delete a build option by first removing the associated components and then deleting the build option.

## **Viewing Build Option Reports**

This option helps to access the Utilization report where build options are utilized in the current application. View the History report to see what modifications have been made to build options in the current application.

## **Exporting Build Options or Build Option Status**

This option helps the user to export build option status to toggle build options on or off within another environment.

It consists of two types as below:

- ☐ Include

It defines that components are enabled and part of the application.

- ☐ Exclude

It defines that components are disabled and not part of the application.



## SECURITY

The common discussion around Oracle apex is when you say it's browser based, the next question that comes to you would be, "How secured is your application?". The best answer we can give is, "Oracle store is built with Oracle apex." Yes you heard it right; it is one of the secured applications where you can perform even financial transactions using apex. There are different features to enhance the security. We shall see some of them.

- ☐ Authentication
- ☐ Authorization
- ☐ Session Management
- ☐ Session State Protection

### Authentication

It is a process or action to verify the identity of the user using password. This can be done in different ways as below:

- ☐ Application express accounts – this uses user accounts and passwords that are created and managed within Oracle Application Express user repository. Your application is authenticated against these accounts.
- ☐ Custom – as always there will be different needs for different users, hence this enables them to sculpture their own authentication using `plssql` program.
- ☐ Database account – this uses the schema username and password from the database where apex is installed. This acts more like no authentication is selected
- ☐ HTTP Header variable – this is used only for the single sign on from some other application.
- ☐ LDAP directory – when you have LDAP server you can use the same credentials maintained in the server.
- ☐ Oracle Server single sign on—this uses the credentials maintained in Oracle server.

### Authorization

Many of the users still get puzzled with authentication and authorization. The basic difference is, authentication is like a lock in your laptop or desktop, which enables the user

to login and use the system, but authorization is like access given for each user to use only software allowed to them. We can define the authorization at each item level also.

| <b>Authorization Scheme Types</b>                               | <b>Description</b>                                                                                                                            |
|-----------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| Exists SQL Query                                                | Enter a query that causes the authorization scheme to pass if it returns at least one row and causes the scheme to fail if it returns no rows |
| NOT Exists SQL Query                                            | Enter a query that causes the authorization scheme to pass if it returns no rows and causes the scheme to fail if it returns one or more rows |
| PL/SQL Function Returning Boolean                               | Enter a function body. If the function returns true, the authorization succeeds.                                                              |
| Item in Expression 1 is NULL                                    | Enter an item name. If the item is null, the authorization succeeds.                                                                          |
| Item in Expression1 is NOT NULL                                 | Enter an item name. If the item is not null, the authorization succeeds.                                                                      |
| Value of Item in Expression 1 Equals Expression 2               | Enter an item name and value. The authorization succeeds if the item's value equals the authorization value.                                  |
| Value of Item in Expression 1 Does NOT Equal Expression 2       | Enter an item name and a value. The authorization succeeds if the item's value is not equal to the authorization value.                       |
| Value of Preference in Expression 1 Does NOT Equal Expression 2 | Enter a preference name and a value. The authorization succeeds if the preference's value is not equal to the authorization value.            |

| <b>Authorization Scheme Types</b>                       | <b>Description</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| Value of Preference in Expression 1 Equals Expression 2 | Enter a preference name and a value. The authorization succeeds if the preference's value equals the authorization value.                                                                                                                                                                                                                                                                                                                                                                |
| Is In Group                                             | <p>Enter a group name. The authorization succeeds if the group is enabled as a dynamic group for the session. See "APEX_AUTHORIZATION.ENABLE_DYNAMIC_GROUPS" in Oracle Application Express API Reference.</p> <p>If the application uses Application Express Accounts Authentication, this check also includes workspace groups that are granted to the user. If the application uses Database Authentication, this check also includes database roles that are granted to the user.</p> |
| Is Not In Group                                         | Enter a group name. The authorization succeeds if the group is not enabled as a dynamic group for the session.                                                                                                                                                                                                                                                                                                                                                                           |

Source: [https://docs.oracle.com/html/E39147\\_04/sec\\_authorization.htm](https://docs.oracle.com/html/E39147_04/sec_authorization.htm)

## Session Management

Session state management is done with the session ID and this session values will be maintained by the item in the browser and developer can restrict the values being changed by the users.

## Session state protection

Session state can be protected adding checksum to the url

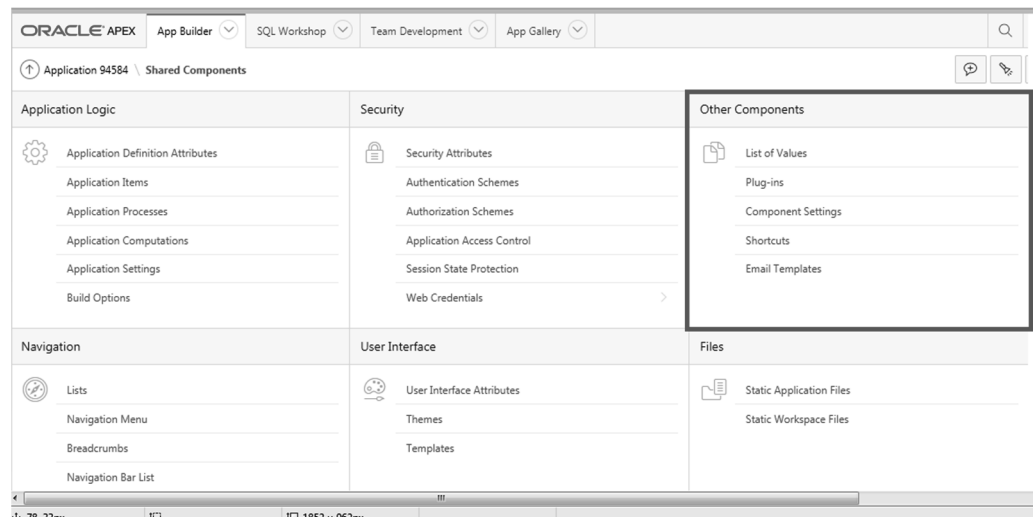
|                                   |                                   |   |
|-----------------------------------|-----------------------------------|---|
| Page Access Protection            | - Do not update pages -           | ? |
| Page Data Entry Item Protection   | Checksum Required - Session Level | ? |
| Page Display-Only Item Protection | Checksum Required - Session Level | ? |
| Application Item Protection       | Checksum Required - Session Level | ? |

OTHER COMPONENTS

The following are the list of other components that are under shared components.

- ☐ List of values
- ☐ Plug-ins
- ☐ Component settings
- ☐ Shortcuts
- ☐ Email Templates

Below we can see how we create the components and their details.



List of Values

A list of values (LOV) is a static or dynamic set of values used to display a popup list of values, select list, check box, or radio group.

In the following example we can create sample LOV with name as “Test” and place the SQL Query in Query box.

E.g.

```
select ename as d,
 empno as r
 from emp
 order by 1
```

List of Values: TEST

Cancel

Delete

Apply Changes

| Show All                                                                   | Name | Subscription | Source: Lists of Values Query |
|----------------------------------------------------------------------------|------|--------------|-------------------------------|
| <div> <div>* List of Values Name</div> <div>TEST</div> <div>?</div> </div> |      |              |                               |

Subscription

Reference Master List of Values From

Refresh

This is the "master" copy of this List of Values.

No Lists of Values subscribe to this List of Values.

Source: Lists of Values Query

Query ?

↶

↷

🔍

↔

📄

A

✓

⚙️

1 select ename as d,

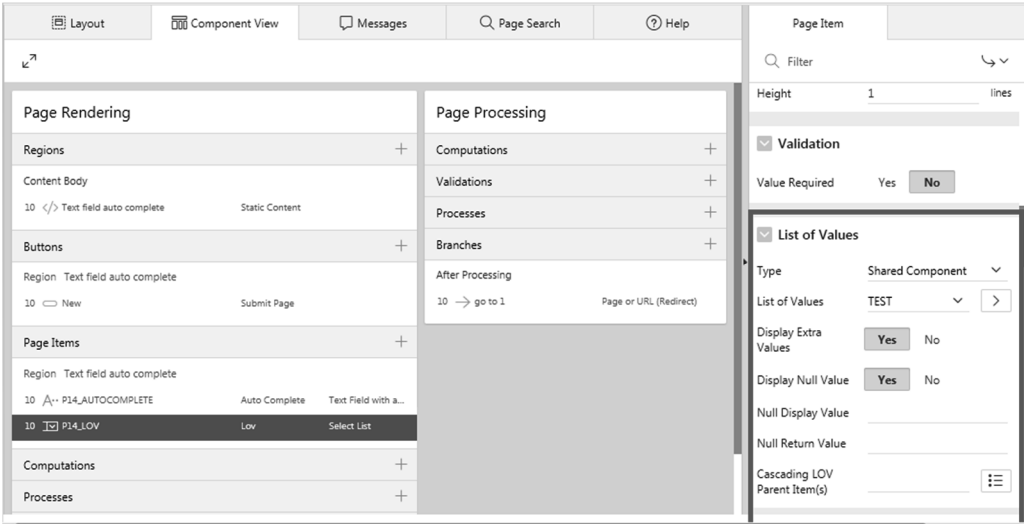
2 empno as r

3 from emp

4 order by 1

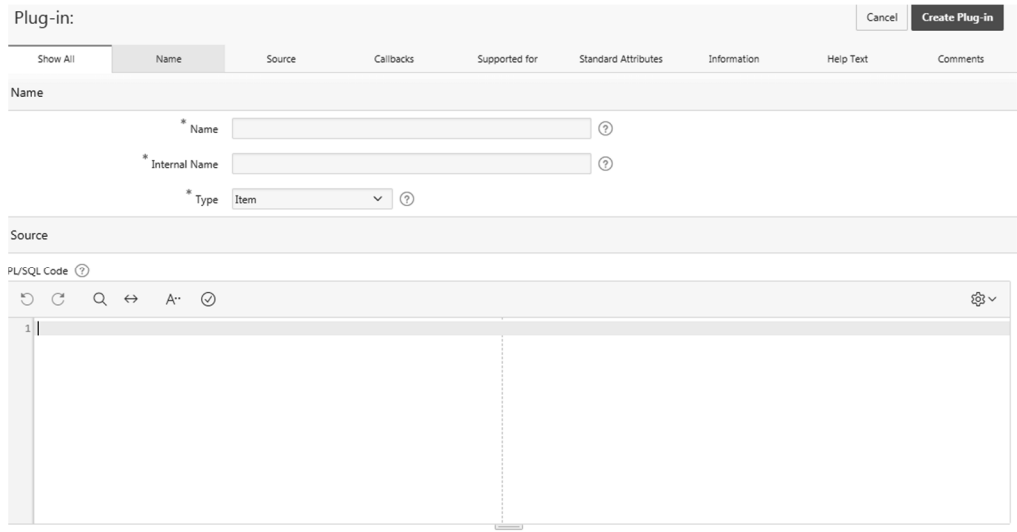
We can include the Shared component LOV on the Page level item creation.

In List of Values, set the type as “Shared Components” and further in List of values set as “Test”.



Plug-ins

App Builder includes built-in item types, region types, dynamic actions, and processes. Use plug-ins to add new declarative types into your application.



We can create a new plugin and assign the name for it.

In PLSQL Code, we can set the process code for the plugin and that can be used in regions type, Item type, dynamic actions and processes.



We can import the other plugins by using the option “Import” and it has been included as Application level, and it will be available in all pages based on the category how the plugin has been created.

⬆ Import

### Import

Select the file you wish to import to the export repository. Once imported, you can install your file.

If the imported file is a productivity or sample application export, the installation wizard will allow you to run the app installation scripts after installing the application definition.

\* Import file

Choose File

No file chosen

\* File Type:

Database Application, Page or Component Export

Worksheet Application Export

**Plug-in**

Theme Export

User Interface Defaults

Team Development Feedback

CSS Export [Deprecated]

Image Export [Deprecated]

File Export [Deprecated]

File Character Set

Unicode UTF-8

Cancel

Help

Next >

Component Settings

Use Component Settings to set application-level values for built-in Oracle Application Express components and installed plug-ins.

| Component Settings      |        |             | History                              |
|-------------------------|--------|-------------|--------------------------------------|
| Q ▾                     |        |             | Go <div></div> <div></div> Actions ▾ |
| Name ↑                  |        |             | Reset                                |
| Calendar                | Region | 4 weeks ago |                                      |
| Color Picker            | Item   | 4 weeks ago |                                      |
| Interactive Grid        | Region | 4 weeks ago |                                      |
| Interactive Report      | Region | 4 weeks ago |                                      |
| Region Display Selector | Region | 4 weeks ago |                                      |
| Rich Text Editor        | Item   | 4 weeks ago |                                      |
| Switch                  | Item   | 4 weeks ago |                                      |

1 - 7

Shortcuts

Use shortcuts to avoid repetitive coding of HTML or PL/SQL functions. You can create a shortcut to define a page control such as a button, HTML text, a PL/SQL procedure, or HTML. Once you define a shortcut, it is stored in a central repository so you can reference it from various locations within your application.

Shortcuts are a repository of shared static or dynamic HTML. Shortcuts are substitution strings that are expanded using the syntax: "SHORTCUT\_NAME". Shortcuts are used in the following locations:

- ☐ Region Source for regions of type HTML\_WITH\_SHORTCUTS
- ☐ Region Templates, Region Headers and Footers
- ☐ Item Labels
- ☐ Item Default Value
- ☐ Item Post Element Text
- ☐ Item Help Text
- ☐ HTML Header of a page

Shortcut

Cancel

Delete

Apply Changes

| Show All                                                                                                     | Name | Subscription | Source | Configuration | Comments |
|--------------------------------------------------------------------------------------------------------------|------|--------------|--------|---------------|----------|
| Name                                                                                                         |      |              |        |               |          |
| <div>* Name</div> <div>DELETE_CONFIRM_MSG</div>                                                              |      |              |        |               |          |
| Subscription                                                                                                 |      |              |        |               |          |
| <div>Reference Master Shortcut From</div> <div></div> <div><input checked="" type="checkbox"/> Refresh</div> |      |              |        |               |          |
| <div>This is the "master" copy of this shortcut.</div> <div>No shortcuts subscribe to this shortcut.</div>   |      |              |        |               |          |
| Source                                                                                                       |      |              |        |               |          |
| <div>* Type</div> <div>Text with JavaScript Escaped Single Quotes</div>                                      |      |              |        |               |          |
| <div>Shortcut</div> <div>Would you like to perform this delete action?</div>                                 |      |              |        |               |          |



## Email Templates

Create templates to define the HTML format and Plain Text formats for the emails you wish to send from an application.

Before we send email from an application, your Instance administrator must log in to Oracle Application Express Administration Services, navigate to the Instance Settings page and configure Email attributes.

- ☐ Template Name – Enter a descriptive name for this template.
- ☐ Static Identifier – Static string identifier used to refer to the template when calling the APEX\_MAIL APIs.
- ☐ Email Subject – Enter the text to display for the email subject.
- ☐ Define the Header, Body, and Footer. Body supports basic HTML markup.
- ☐ Under Advanced, optionally click Load Default HTML.
- ☐ The default HTML appears in the HTML Template. If needed, edit the default HTML Template.
- ☐ Under Plain Text Format, enter the appropriate template defaults.
- ☐ Under Comments, optionally enter comments that describe this template.
- ☐ Click Create Email Template.
- ☐ Next, create a button and process to call the APEX\_MAIL API to send email.

Reference: <https://docs.oracle.com/en/database/oracle/application-express/18.2/htmldb/shared-components-page.html#GUID-A3FBCD13-FF7A-45F0-8DCB-B45C18EFCAC7>

## NAVIGATION

The Navigation region on the shared component has the following links,

- ☐ Lists
- ☐ Navigation Menu
- ☐ Breadcrumbs
- ☐ Navigation Bar List

### Lists

The collection of links is a list. A list will be available under Navigation region on the Shared Components. The appearance of the list display in the page is based on the list templates. Each list can be controlled by adding a display condition to it. Let's see how to create a list.

Two types of lists can be created in Oracle Application Express:

- ☐ Static Lists
- ☐ Dynamic Lists.

### Static Lists

A Static List is based on predefined display and return values. When you wish to create a static list, a list entry label and target (either a page or URL) attributes need to be defined. A static list can be creating using the following option, creating from scratch, by copying existing entries, or by adding the list entries. Each list will have display condition, which enables to control its display.

The following describes how to create a Static List:

- ☐ Navigate to the application shared components.
- ☐ Create the list using the Create List Wizard.
- ☐ Mention whether to create the list from scratch or by copying an existing list.
- ☐ If the option from scratch is chosen, then enter the details for the Name, Select type as Static and click next.

- ☐ Define list entries and click next.
- ☐ Specify the details for the following and click Create list.
  - Create List regions
  - Region Position
  - Region Template
  - List template

## Dynamic List

A Dynamic list is based on an SQL query or a PL/SQL function executed at runtime. Using Dynamic list, a list can be created dynamically which will also support for mobile frameworks.

The list definition displays a specific type of page item, such as progress bars, sidebar, bullet navigation list, or navigation menu. You can control how a list displays through templates.

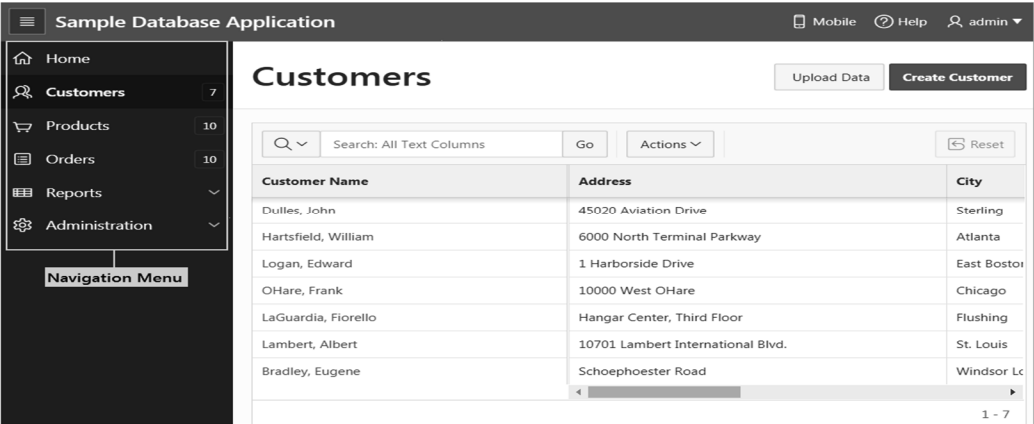
The following describes the process to create Dynamic List:

- ☐ Navigate to the application shared components.
- ☐ Create the list by using the Create List Wizard.
- ☐ Mention whether to create the list from scratch or by copying an existing list.
- ☐ If the option "from scratch" is chosen, then enter the details for the Name, Select type as Dynamic and click next.
- ☐ Enter an SQL query or a PL/SQL function returning an SQL query.
- ☐ Add the list to a page by creating a List region.

## Navigation Menu

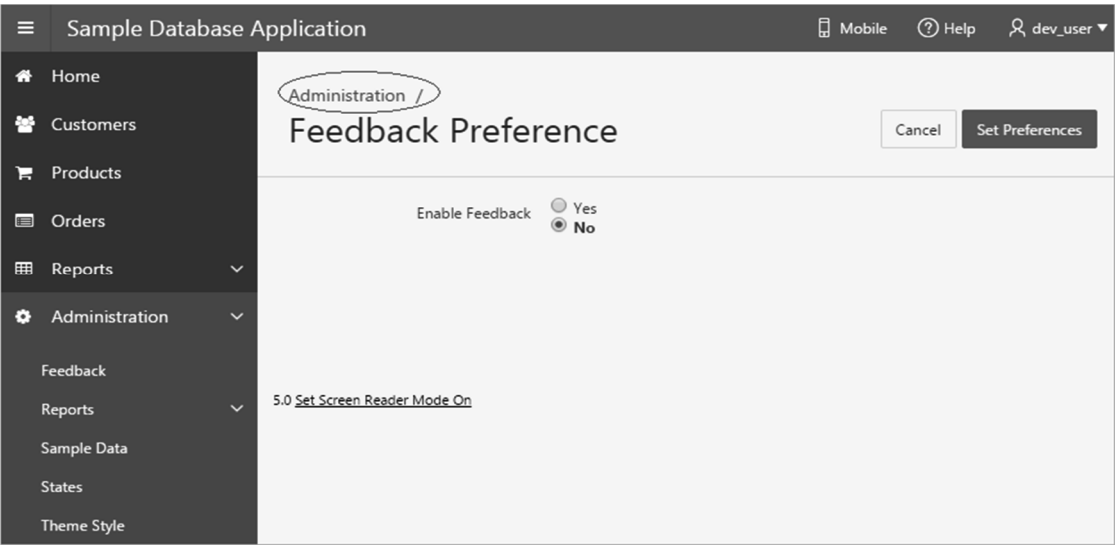
A navigation menu is a list link that enables users to navigate the pages in an application. Navigation menus are only supported in the applications using the Universal Theme. A navigation menu is a list with hierarchical list entries. When you create an application, the Create Application Wizard automatically creates a navigation menu for you and populates it with list entries linking to the application pages you have created. The example below shows the Sample Database Application with a navigation menu displaying as a sidebar. In

this example, the navigation menu includes list entries for the application pages: Home; Customers; Orders; Reports; and Administration.



Breadcrumbs

A breadcrumb is a hierarchical list of links that shows where the user is within the application from a hierarchical perspective. It helps the users to click a specific breadcrumb link to view the page instantly. We can use breadcrumbs as a second level of navigation which will always place at the top of the page, complementing other user interface elements such as tabs and lists.



## Navigation Bar List

Navigation bar lists enable users to create a simple navigation path for migrating between pages in an application. The section of a navigation bar depends upon the associated page template. A list entry can be an image, an image with text beneath it, or text.

Source: <https://docs.oracle.com/en/database/oracle/application-express/19.1/htmldb/shared-components-page.html#GUID-D5A5F67A-DF1B-48C8-9CD5-12CDA4D14DA4>

## USER INTERFACE

The following is a list of links under User interface on the Shared Components page.

- ☐ User Interface Attributes
- ☐ Themes
- ☐ Templates

### User Interface Attributes

Use User Interface Details to define the specific settings for the selected user interface type.

Attributes enable you to select the home and login pages that users are redirected to when they access this user interface. You can also set the current theme style and global page.

Using the Navigation Menu you can select the list and position used to render the navigation menu as well as setting the navigation menu list template.

Navigation Bar settings enable you to select list and list templates. Selecting classic implementation uses tabs instead of a list.

The JavaScript, Cascading Style Sheet and Concatenated Files attributes can be used to select alternate content delivery networks, load additional CSS files and to optimize the way JavaScript and CSS files are loaded.

User Interface Details

Cancel

Delete

Apply Changes

Show All

Identification

Attributes

Navigation Menu

Navigation Bar

JavaScript

Cascading Style Sheets

Concatenated Files

Attributes

Auto Detect

Yes

No

?

Default

Yes

No

?

Enable End Users to choose Theme Style

Yes

No

?

Add "Built with APEX" to Footer

Yes

No

?

Home URL

f?p=&APP\_ID:1:&SESSION.

⋮

?

Login URL

f?p=&APP\_ID:LOGIN\_DESKTOP:&SESSION.

⋮

?

Theme:

Universal Theme

?

Theme Style:

Vita

▼

Global Page:

0

?

E.g. We can change the theme style and set the home page and login page details.

By default its Universal theme and theme style is “Vita”. We can also change the theme style.

New Theme Style: Vita - Dark

User Interface Details

Cancel

Delete

Apply Changes

Show All

Identification

Attributes

Navigation Menu

Navigation Bar

JavaScript

Cascading Style Sheets

Concatenated Files

Attributes

Auto Detect

Yes

No

?

Default

Yes

No

?

Enable End Users to choose Theme Style

Yes

No

?

Add "Built with APEX" to Footer

Yes

No

?

Home URL

f?p=&APP\_ID:1:&SESSION.

⋮

?

Login URL

f?p=&APP\_ID:LOGIN\_DESKTOP:&SESSION.

⋮

?

Theme:

Universal Theme

?

Theme Style:

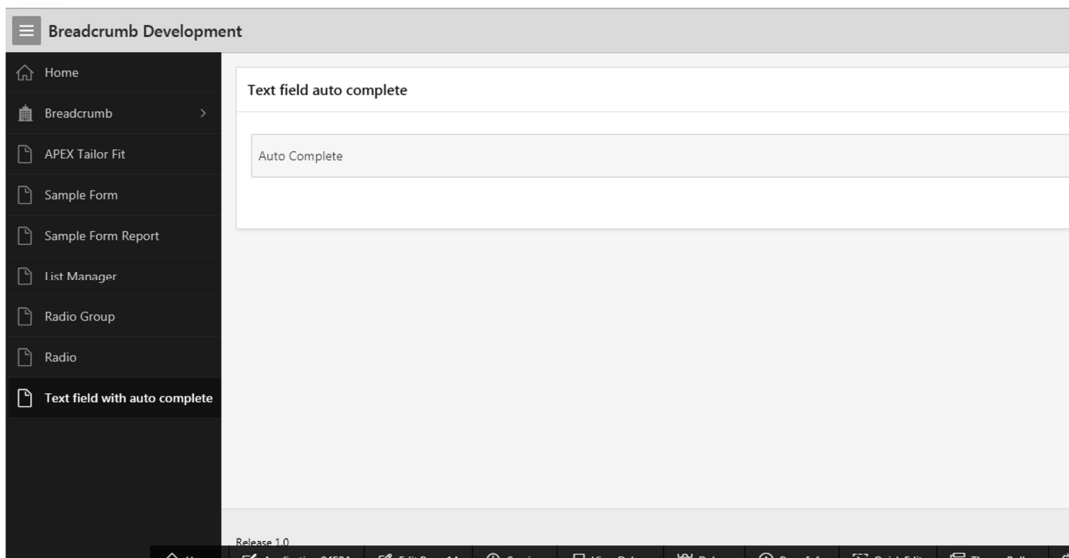
Vita - Dark

▼

Global Page:

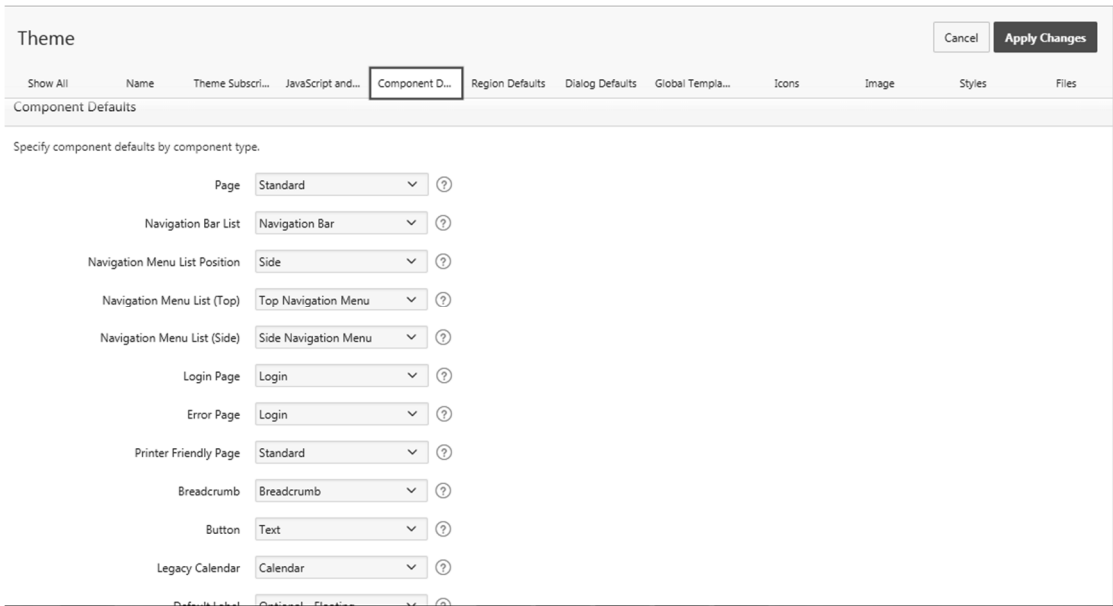
0

?



Themes

A theme is a named collection of templates that defines the application user interface.



If you want to edit the theme then we can copy the theme and we can customize the theme that we want to edit.

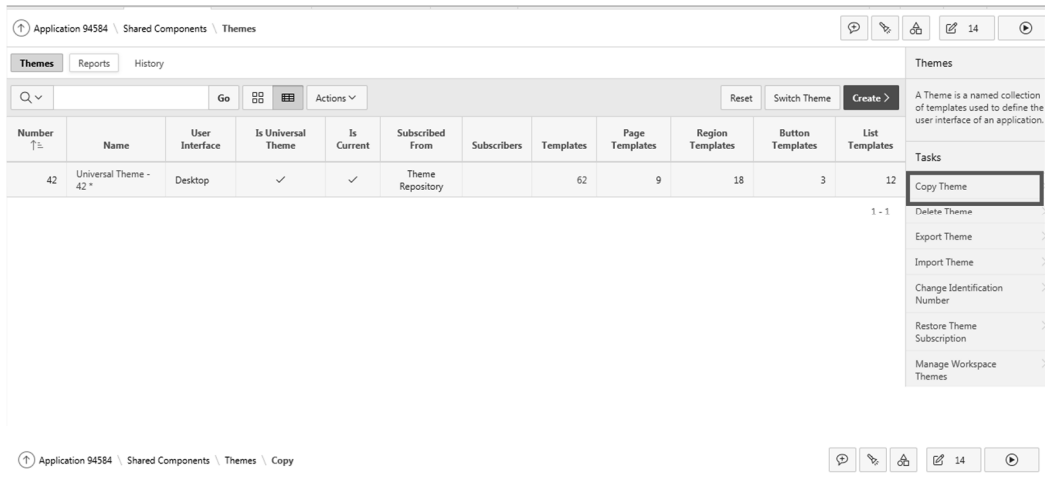
There are multiple options to edit the theme by using the below methods.

ORACLE APEX

- ☐ Copy theme
- ☐ Delete theme
- ☐ Import theme
- ☐ Export theme

Copy theme

We can copy the theme by selecting the copy theme button.



Copy Theme

Copy Theme

Each theme is identified by a numeric identification number (ID). Use this to make a copy of an existing theme and specify a new theme ID. Copying a theme is useful if you wish to export a theme with a different ID.

Application: 94584 Breadcrumb Development

Copy from Theme: 42 Universal Theme

\* Copy to this Theme ID: 100

Subscribe Theme: Yes No

Cancel Next >

| Theme copied. |                        |                |                    |            |                  |             |           |                |                  |                  |                |
|---------------|------------------------|----------------|--------------------|------------|------------------|-------------|-----------|----------------|------------------|------------------|----------------|
| Number        | Name                   | User Interface | Is Universal Theme | Is Current | Subscribed From  | Subscribers | Templates | Page Templates | Region Templates | Button Templates | List Templates |
| 42            | Universal Theme - 42 * | Desktop        | ✓                  | ✓          | Theme Repository | 1           | 62        | 9              | 18               | 3                | 12             |
| 100           | Universal Theme - 100  | Desktop        | ✓                  |            | 94584            |             | 62        | 9              | 18               | 3                | 12             |



Delete Theme

We can delete the theme by clicking the button “Delete Theme” and click Next and Delete Theme to delete the theme.

Application 94584 \ Shared Components \ Themes \ Delete

14

Delete Theme

Delete Theme

Application: 94584 - Breadcrumb Development

\* Delete Theme: 100. Universal Theme

> Delete Theme

Cancel

Next >

Import Theme

We can import the theme by clicking the Import theme button.

Application 94584 \ Shared Components \ Themes

14

Themes

Reports

History

Q

Go

Actions

Reset

Switch Theme

Create >

| Number | Name                   | User Interface | Is Universal Theme | Is Current | Subscribed From  | Subscribers | Templates | Page Templates | Region Templates | Button Templates | List Templates |
|--------|------------------------|----------------|--------------------|------------|------------------|-------------|-----------|----------------|------------------|------------------|----------------|
| 42     | Universal Theme - 42 * | Desktop        | ✓                  | ✓          | Theme Repository | 1           | 62        | 9              | 18               | 3                | 12             |
| 100    | Universal Theme - 100  | Desktop        | ✓                  |            | 94584            |             | 62        | 9              | 18               | 3                | 12             |

1 - 2

Themes

A Theme is a named collection of templates used to define the user interface of an application.

Tasks

Copy Theme

Delete Theme

Export Theme

Import Theme

Change Identification Number

Restore Theme Subscription

Manage Workspace Themes

Export theme

We can Export the theme by clicking the Export theme button.

Application 94584 \ Shared Components \ Themes

Themes

Reports

History

Q

Go

Actions

Reset

Switch Theme

Create

| Number | Name                   | User Interface | Is Universal Theme | Is Current | Subscribed From  | Subscribers | Templates | Page Templates | Region Templates | Button Templates | List Templates |
|--------|------------------------|----------------|--------------------|------------|------------------|-------------|-----------|----------------|------------------|------------------|----------------|
| 42     | Universal Theme - 42 * | Desktop        | ✓                  | ✓          | Theme Repository | 1           | 62        | 9              | 18               | 3                | 12             |
| 100    | Universal Theme - 100  | Desktop        | ✓                  |            | 94584            |             | 62        | 9              | 18               | 3                | 12             |

1 - 2

Themes

A Theme is a named collection of templates used to define the user interface of an application.

Tasks

Copy Theme

Delete Theme

Export Theme

Import Theme

Change Identification Number

Restore Theme Subscription

Manage Workspace Themes

Switch Theme

Switch theme option is used to switch the old theme to new theme.

Application 94584 \ Shared Components \ Themes \ Switch Theme

14

Switch Theme

Identify Theme

A theme is a collection of templates. When you switch a theme, all templates assigned to components within one theme are assigned to templates in another theme. Application Express accomplishes template mapping through the assignment of template class identifiers.

Application: 94584 - Breadcrumb Development

\* Currently Active Theme: - Select Old Theme -

\* Switch to Theme: - Select New Theme -

Cancel

Create Theme

Next

Templates

The Application Express engine constructs the appearance of each page in an application using Templates. Templates control the look and feel of specific constructs within your application, such as pages, regions, items, and menus.

To edit template attributes, select the template name. To preview a template, click the Run icon.

| Application 94584 \ Shared Components \ Templates                                                                      |                |            |                |                         |         |       |      |  |
|------------------------------------------------------------------------------------------------------------------------|----------------|------------|----------------|-------------------------|---------|-------|------|--|
| <div> <div>Templates</div> <div>Subscription</div> <div>Publish</div> <div>Utilization</div> <div>History</div> </div> |                |            |                |                         |         |       |      |  |
| <div> <div>Q v</div> <div>Go</div> <div>Actions v</div> <div>Reset</div> <div>Create &gt;</div> </div>                 |                |            |                |                         |         |       |      |  |
| Type ↑≡                                                                                                                | Name           | References | Updated        | Updated By              | Default | Theme | Copy |  |
| Breadcrumb                                                                                                             | Breadcrumb     | 1          | 11 minutes ago | kpandi.balaji@gmail.com | ✓       | 100   |      |  |
| Breadcrumb                                                                                                             | Breadcrumb     | 3          | 4 weeks ago    | kpandi.balaji@gmail.com | ✓       | 42    |      |  |
| Button                                                                                                                 | Icon           | 0          | 4 weeks ago    | kpandi.balaji@gmail.com |         | 42    |      |  |
| Button                                                                                                                 | Icon           | 0          | 11 minutes ago | kpandi.balaji@gmail.com |         | 100   |      |  |
| Button                                                                                                                 | Text           | 18         | 4 weeks ago    | kpandi.balaji@gmail.com | ✓       | 42    |      |  |
| Button                                                                                                                 | Text           | 1          | 11 minutes ago | kpandi.balaji@gmail.com | ✓       | 100   |      |  |
| Button                                                                                                                 | Text with Icon | 0          | 11 minutes ago | kpandi.balaji@gmail.com |         | 100   |      |  |
| Button                                                                                                                 | Text with Icon | 0          | 4 weeks ago    | kpandi.balaji@gmail.com |         | 42    |      |  |
| Label                                                                                                                  | Hidden         | 3          | 4 weeks ago    | kpandi.balaji@gmail.com |         | 42    |      |  |
| Label                                                                                                                  | Hidden         | 0          | 11 minutes ago | kpandi.balaji@gmail.com |         | 100   |      |  |
| Label                                                                                                                  | Optional       | 0          | 4 weeks ago    | kpandi.balaji@gmail.com |         | 42    |      |  |

In the example below we can change the button template.

Button templates enable developers to customize the look and feel of a button. To build a button, you can use multiple images or HTML tags. Using button templates is optional.

To edit the template, we can copy the template and set the new name for the template and then we can edit based on what we need.

Below are the template types for the buttons like Normal Template and Hot Template.

We can change the HTML to edit the button's look and feel. Once we have created the new template, it will be reflected on all the pages.

If we want to change the button look based on what we customize, we can select the newly created template name for the button.



Reference: <https://docs.oracle.com/en/database/oracle/application-express/18.2/htmdb/shared-components-page.html#GUID-A3FBCD13-FF7A-45F0-8DCB-B45C18EFCAC7>

## FILES

The following describes the files links on the shared component.

- ☐ Static Application Files
- ☐ Static Workspace Files

### Static Application Files

The Static Application Files link enables you to upload, view, download and delete the files which include images, CSS files, js files and other files. This can be managed within application level.

To practise the above mentioned action, follow the steps given below:

To Upload,

- ☐ Navigate to the shared component page in application.
- ☐ Select Static Application files under Files.
- ☐ Click Upload to upload file.

- ☐ Enter the name of the directory in the directory field where the file should be stored. If nothing is specified, then by default the file will be stored in the root directory.
- ☐ Select file to be uploaded by specifying in File.
- ☐ Mention the character set encoding for the file if required.
- ☐ Select the option Yes/No for unzip file.
- ☐ Click Upload.

To View and Delete,

- ☐ Navigate to the shared component page in application.
- ☐ Select Static Application files under Files.
- ☐ A Report appears. A Column File Name will act as link to edit the files.
- ☐ Click any of the filename.
- ☐ Information will be displayed in the pop up.
- ☐ To delete, click Delete button.
- ☐ To exit, click Exit button.

To Download,

- ☐ Navigate to the shared component page in application.
- ☐ Select Static Application files under Files.
- ☐ A Report appears with the following columns,
  - File Name
  - Mime Type
  - File Size
  - Reference
  - File
- ☐ Locate the file to be downloaded and click the Download link.

### **Static Workspace Files**

The Static Workspace Files link enables you to upload, view, download and delete the files which include images, CSS files, js files and other files. This can be managed by workspace level.

To practise the above mentioned action, follow the steps given below:

To Upload,

- ☐ Navigate to the shared component page in application.
- ☐ Select Static Workspace files under Files.
- ☐ Click Upload to upload file.
- ☐ Enter the name of the directory in the directory field where the file should be stored. If nothing is specified, then by default the file will be stored in the root directory.
- ☐ Select file to be uploaded by specifying in File.
- ☐ Mention the character set encoding for the file if required.
- ☐ Select the option Yes/No for unzip file.
- ☐ Click Upload.

To View and Delete,

- ☐ Navigate to the shared component page in application.
- ☐ Select Static Workspace files under Files.
- ☐ A Report appears. A Column File Name will act as link to edit the files.
- ☐ Click any of the filename.
- ☐ Information will be displayed in the pop up.
- ☐ Click Delete button to delete.
- ☐ To exit, click Exit button.

To Download,

- ☐ Navigate to the shared component page in application.
- ☐ Select Static Workspace files under Files.
- ☐ A Report appears with the following columns,
  - File Name
  - Mime Type
  - File Size
  - Reference
  - File
- ☐ Locate the file to be downloaded and click the Download link.

Source: <https://docs.oracle.com/en/database/oracle/application-express/19.1/htmldb/shared-components-page.html#GUID-B9987FB8-956A-456F-8588-F20B5CD8F569>

## REPORTS

The Report region on the shared component has the following links,

- ☐ Report Queries
- ☐ Report layouts

### Report Queries

Defining a report query on the shared component helps to print report region within the current application. A report Query identifies the info to be extracted. In contrast to SQL statements contained in regions, report queries containing SQL statements will be validated when the query is saved. Note that report queries should be SQL statements, not functions returning SQL statements.

With a report design, you can match a report query and download it as a formatted document. A generic layout is used if no report layout is selected. The purpose of the generic layout is to test and verify a report query. Only the first result set is included in the print document when using the generic layout option and multiple source queries are defined.

You then incorporate these reports with the application to make them accessible to end users. For example, a report query can be associated with a button, list item, branch, or other navigation component that allows you to use URLs as targets. Then selecting that item starts the process of printing.

### Creating a Report Layout

- ☐ Navigate to the shared component of your current application.
- ☐ Click Report Queries under Reports.
- ☐ Click Create.
- ☐ Specify the information for the following,
  - Report Query Name
  - Output Format

- Item
  - View file as
  - Session State
  - Click Next.
- Provide the SQL statement for SQL query section.
- Specify the information to Data Source for Report layout under Download Definition and Click Next.
- Provide the information for the following,
  - Report Layout Source – If ‘Create file based report layout’ is selected then ‘Report layout file’ field is mandatory.
  - Layout Name – Enter a layout name.
  - Report Layout File – Click Browse and select Rich Text Editor.
  - Click Next.
- Final screen will appear, view the information shown in the screen and click Create to create Report Query.

## Report Layout

Using Report Layout, either a classic report region or a report query can be formatted and renders the data in a printer-friendly format. If a report layout is not selected, a default XSL-FO layout will be used. The default layout of XSL-FO is always used to make interactive reports region.

To create a report layout,

- Navigate to the shared components page.
- Click **Report Layout** under Reports.
- Click Create.
- Select a report layout type and click next.
  - Generic Columns (XSL-FO) – Uses the default template to populate the report. You can customize the default code in the next step.
  - Named Columns (XSL-FO) – If Named Columns (XSL-FO) is selected then it requires XSL-FO or RTF file to be uploaded in the next step.



- ☐ Provide the appropriate information for the option shown. Note the option on this page will appear based on the report layout selection done in the previous page.
- ☐ Click Create Layout.

To Edit a report layout,

- ☐ Navigate to the Shared Components page.
- ☐ Click Report layout under Report region.
- ☐ The report will appear with information if the report layout is created already.
- ☐ Select the layout you want to edit.
- ☐ A pop up will appear, do the necessary changes and click Apply Changes. This is for generic column layouts.
- ☐ For named column layouts, click Download and do the changes in the file and upload the revised file.

Source: <https://docs.oracle.com/en/database/oracle/application-express/19.1/htmdb/shared-components-page.html#GUID-C075D396-A6F6-4B1D-835A-58EC4240FFF8>

## GLOBALIZATION

The following are the list of links under Globalization on the Shared Components page.

- ☐ Globalization Attributes
- ☐ Text Messages
- ☐ Translate Application

### Globalization Attributes

We can develop applications that can run concurrently in different languages. Click this link to specify globalization options such as the Application Primary Language and Application Language Derived From attributes.

Application 94584

CancelApply Changes

Application Primary LanguageEnglish (en)?

Application Language Derived FromChinese (Taiwan) (zh-tw)?

Document DirectionChinese (zh)?

Document DirectionCroatian (hr)?

Document DirectionCzech (cs)?

Document DirectionDanish (da)?

Application Date FormatDutch (Belgium) (nl-be)?

Application Date FormatDutch (Netherlands) (nl)?

Application Date FormatEnglish (Australia) (en-au)?

Application Date Time FormatEnglish (Belize) (en-bz)?

Application Date Time FormatEnglish (Canada) (en-ca)?

Application Date Time FormatEnglish (Ireland) (en-ie)?

Application Timestamp FormatEnglish (Jamaica) (en-jm)?

Application Timestamp FormatEnglish (New Zealand) (en-nz)?

Application Timestamp Time Zone FormatEnglish (Philippines) (en-ph)?

Application Timestamp Time Zone FormatEnglish (South Africa) (en-za)?

Character Value ComparisonEnglish (Trinidad) (en-tt)?

Character Value ComparisonEnglish (United Kingdom) (en-gb)?

Character Value ComparisonEnglish (United States) (en-us)?

Character Value ComparisonEnglish (Zimbabwe) (en-zw)?

Automatic Time ZoneEnglish (en)?

Automatic Time ZoneYesNo?

Automatic CSV EncodingYesNo?

No translations found.

We can set the application date format. Once the date format has been set, then it will be set for the whole application. Whenever we select the date from date picker, it will set as the format that we mentioned in globalization.

Document Direction

We can set the document reading direction. By default it is from left to right.

We can change it to from right to left also.

Below is a simple example to change the primary language type to “Chinese”

Application 94584

CancelApply Changes

Globalization

Application Primary LanguageChinese (China) (zh-cn)?

Application Language Derived FromApplication Primary Language?

Document DirectionLeft-To-Right?

Application Date Format?

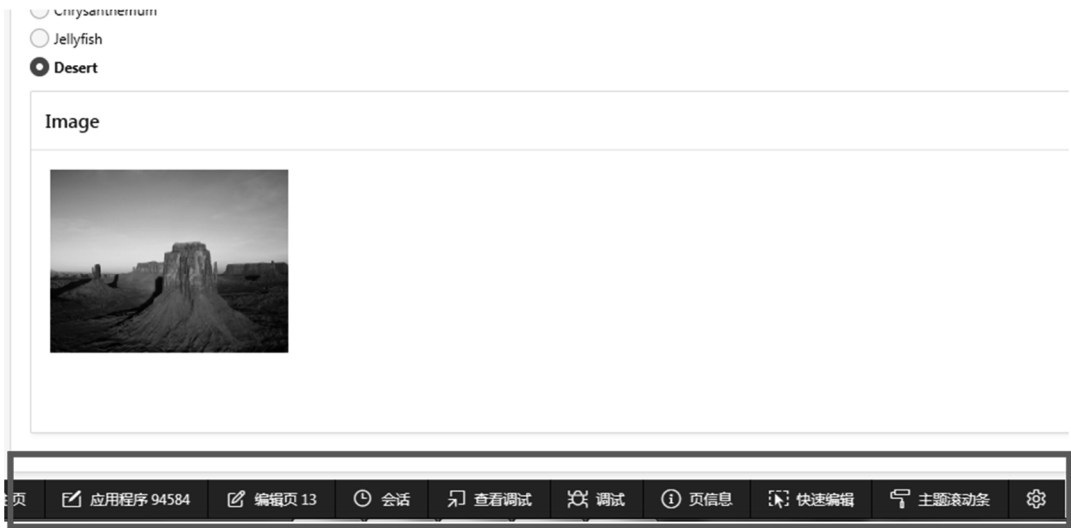
Application Date Time Format?

Application Timestamp Format?

Application Timestamp Time Zone Format?

Character Value Comparison?

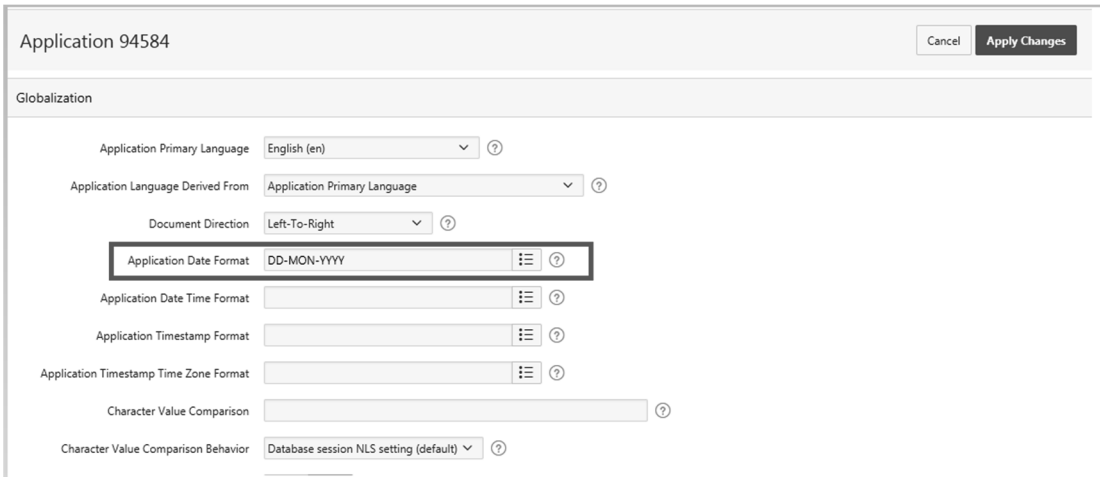
Character Value Comparison BehaviorDatabase session NLS setting (default)?



**Application Date format.**

Once the Application date format has been set, it will be followed for whole application.

By default the date format is “DD-MON-RR”. Now we are changing the format to “DD-MON-YYYY”.



Now create a new date picker item and the format looks as below with format “DD-MON-YYYY”.

Once we create a new application, and set the format as above, it will help to reflect for all the pages.

Date  
29-Sep-2019



## Text Messages

Text Messages can be used to build translatable text strings with substitution variables that can be called from PL/SQL packages, procedures, and functions.

We may need to translate messages if your application:

Includes PL/SQL regions or PL/SQL processes, or calls a PL/SQL package, procedure, or function. If it does, you may need to translate the generated HTML.

Uses a language that is not one of the ten languages into which Oracle Application Express is translated. If it does, you may need to translate messages used in reports.

We can create text message by click the button “Create Text Message”.

Application 94584 \ Shared Components \ Translate \ Text Messages

Q Go Actions Reset Grid Edit Create Text Message

### Create/Edit Text Message

Messages are designed to provide translation services for use in PL/SQL.

Application: 94584 Breadcrumb Development

\* Name

Language English (en)

Used in JavaScript Yes No

\* Text

Cancel Create And Create Another Create Text Message

We create a new text message for the translation and set the name for it.

Translate Application

Applications can be translated from a primary language into other languages. Each translation results in the creation of a new translated application. Each translation requires a mapping which identifies the target language as well as the translated application ID. Translated applications cannot be edited directly in the App Builder.

Once the translation mappings are established, the translatable text within the application is seeded into a translation repository. This repository can then be exported to an XLIFF for translation.

Once the XLIFF file is populated with the translations, one file per language, the XLIFF file is uploaded back into the translation repository. The final step is to publish each translated application from the translation repository.

A translated application will require synchronization when the primary application has been modified since the translated version was last published. Even modifications to application logic will require synchronization. To synchronize, seed and publish the translated application.

Application 94584 \ Shared Components \ Translate \ Language Mappings \ Edit

14

Create/Edit Application Language Mapping

To perform a translation, you create a unique application ID for the translated application. Use this page to map an existing application ID to a translation application ID. Note that when an application mapping is deleted, any corresponding translated applications are also deleted.

Primary Language Application94584 Breadcrumb Development

\* Translation Application101

\* LanguageArabic (ar)

Brazilian Portuguese, Chinese (China), Chinese (Taiwan), English, French, German, Italian, Japanese, Korean, Spanish

Document DirectionLeft-To-Right

Image Directory

Comments

Click “Create”, to create a new translated application.

Application 94584 \ Shared Components \ Translate \ Language Mappings

✓ Action processed.

Go

Actions

Cancel

Create

| Translated Application           | Language    | Application Exists | Published | Translation Comment | Created | Updated |
|----------------------------------|-------------|--------------------|-----------|---------------------|---------|---------|
| 72427836498236469826492634982634 | Arabic (ar) | No                 | -         | -                   | Now     | Now     |

233

Reference:

<https://docs.oracle.com/en/database/oracle/application-express/18.2/htmldb/translating-messages.html#GUID-2B07BBBC-AFF6-493D-B173-81C7C97C26DB>

<https://docs.oracle.com/en/database/oracle/application-express/18.2/htmldb/shared-components-page.html#GUID-270EA688-3883-4D37-8435-914648A25021>

## INTEGRATION WITH OTHER DATABASE WITHOUT DBLINK

---

When we say integration with other database. The first thing that comes to your mind is DBlink. This was the case before webservice. Now we can integrate two or more databases to Apex. Let's see how we achieve it.

Web service is a standardized medium to transfer or transmit data between the client and server applications LAN or WAN.

There are mainly two types of web services.

- ☐ SOAP web services.
- ☐ RESTful web services.

**SOAP web service** – it's an XML based protocol to access the web service and it is lightweight and recommended by W3C consortium, which is the governing body for all web standards. The REST is an abbreviated form of a REpresentational State Transfer

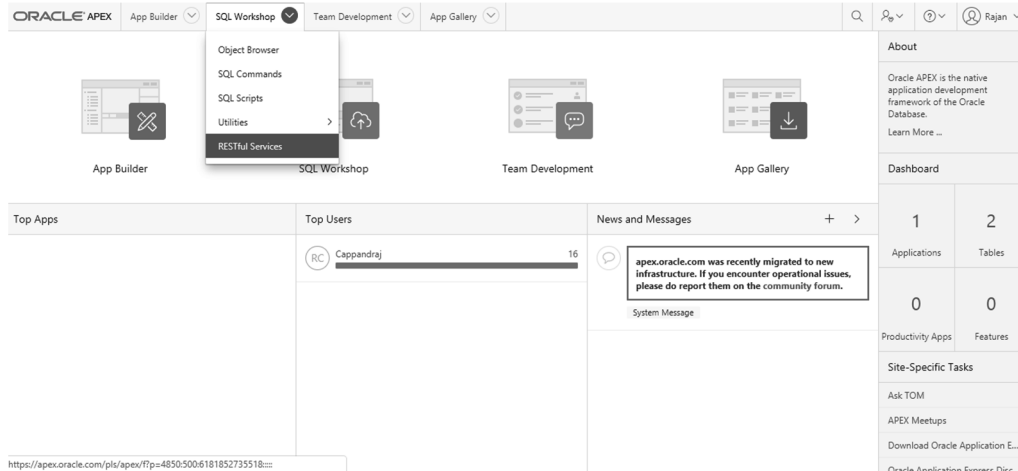
**RESTful web services** – REST is used to build web service which is lightweight and scalable.

- ☐ **POST** – This would be used to create a new employee using the RESTful web service
- ☐ **GET** – This would be used to get a list of all employees using the RESTful web service

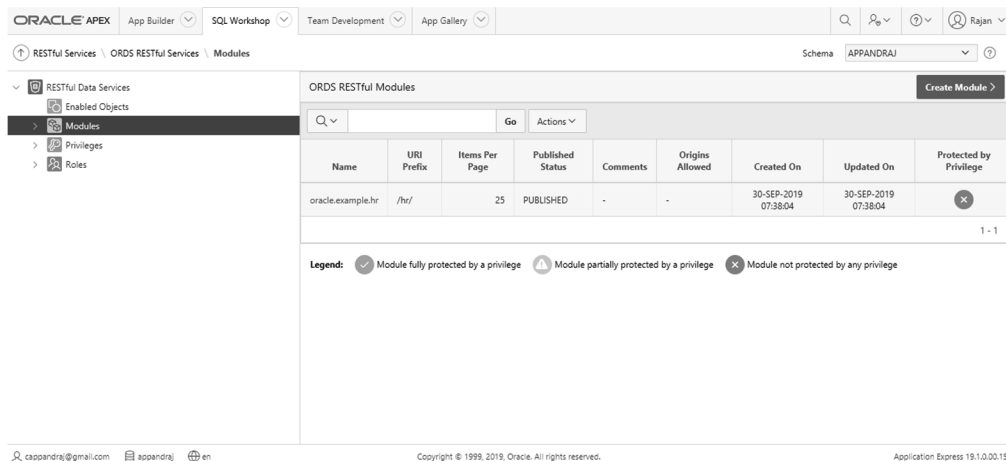
- ❑ **PUT** – This would be used to update all employees using the RESTful web service
- ❑ **DELETE** – This would be used to delete all employees using the RESTful web service

The following are the steps to create endpoint

**Step 1:** From the Oracle Application Express Home page, select the SQL Workshop tab and select the RESTful services.



**Step 2:** From RESTful services page, Select the Create Module Service Option.



**Step 3:** Under ORDS Module Definition Enter the Module Name, Base Path and other required details. Then select the Create Module option.



## INTEGRATION WITH OTHER DATABASE WITHOUT DBLINK

The screenshot shows the Oracle APEX interface for defining a RESTful Service module. The breadcrumb trail is RESTful Services > ORDS RESTful Services > Modules > Module Definition. The schema is APPANDRAJ. On the left, a tree view shows the project structure under RESTful Data Services, with Modules expanded to show 'oracle.example.hr' and its sub-modules. The main area is the 'ORDS Module Definition' form. Fields include: Module Name (book), Base Path (Doyensys), Is Published (Yes), Pagination Size (25), Origins Allowed (empty), and a Comments text area. A legend at the bottom indicates that the 'Template fully protected by a privilege' checkbox is checked. The footer shows the user email, schema name, copyright, and application version.

ORACLE APEX App Builder SQL Workshop Team Development App Gallery

RESTful Services \ ORDS RESTful Services \ Modules \ Module Definition Schema APPANDRAJ

RESTful Data Services  
Enabled Objects  
Modules  
oracle.example.hr  
empinfo/  
employees/  
employees/id  
employeesfeed/  
employeesfeed/id  
empsec/empname  
empsecformat/empname  
version/  
Privileges  
Roles

ORDS Module Definition

\* Module Name book

\* Base Path Doyensys

\* Is Published Yes No

\* Pagination Size 25

Origins Allowed

Comments

Legend: ☒ Template fully protected by a privilege ☐ Template not protected by any privilege

cappandraj@gmail.com appandraj en Copyright © 1999, 2019, Oracle. All rights reserved. Application Express 19.1.0.00.15

**Step 4:** After Module is Created, Select the Create Template option.

This screenshot shows the same 'ORDS Module Definition' form as the previous one, but with additional options at the bottom. The 'Full URL' field is now populated with 'https://apex.oracle.com/pls/apex/doyensys/Doyensys/'. Below the form, there is a 'Generate Swagger Doc' button and a 'Resource Templates' section. The 'Resource Templates' section has a search bar and a 'Go' button. A 'Create Template >' button is visible on the right side of the 'Resource Templates' section. The bottom of the screen shows a message: 'No Templates are currently defined for the selected module.'

Module Created

ORDS Module Definition

\* Module Name book

\* Base Path /Doyensys/

Full URL https://apex.oracle.com/pls/apex/doyensys/Doyensys/

\* Is Published Yes No

\* Pagination Size 25

Origins Allowed

Comments

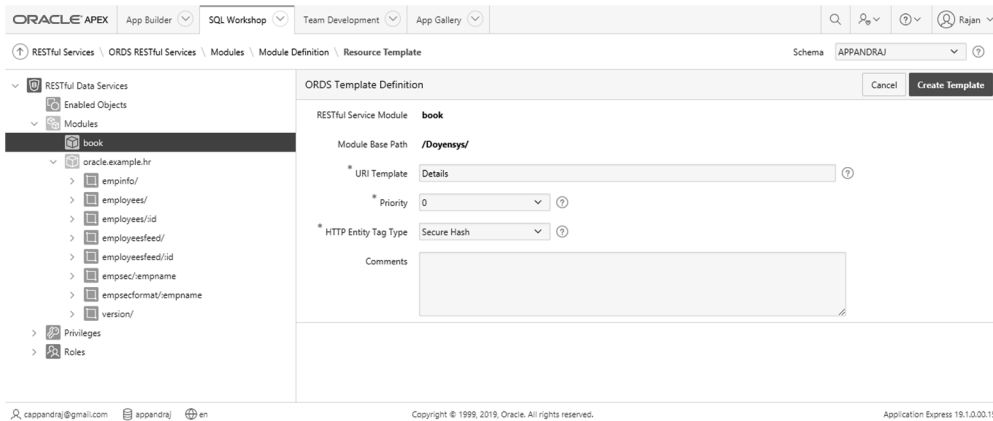
Generate Swagger Doc

Resource Templates

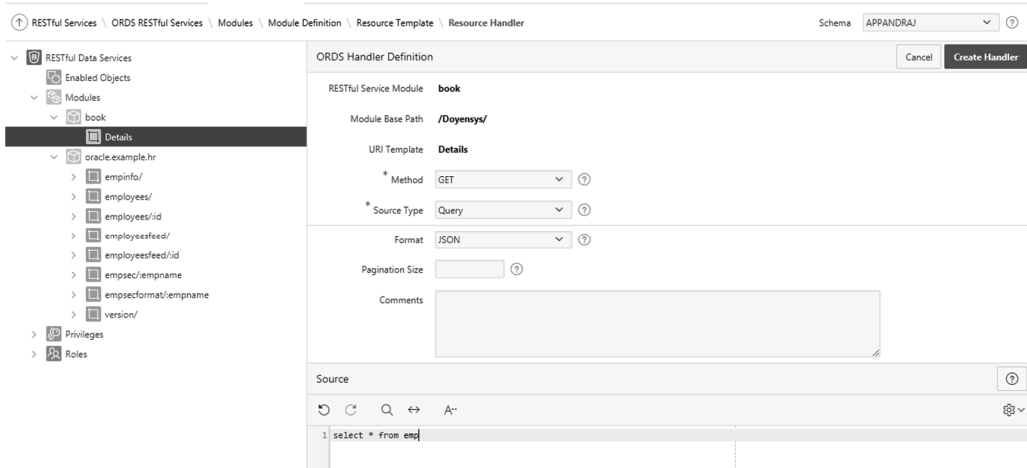
Create Template >

No Templates are currently defined for the selected module.

**Step 5:** Under ORDS Template Definition Enter the URI template, Priority and HTTP Entity tag type. Then select the Create template option.



**Step 6:** After Template is created, Select Create Handler Option.



**Step 7:** Under ORDS Handler Definition select method and source type. Then select the Create Handler option.

**Step 8:** Then Select the Template Name created by you and Copy URL from full URL Section.

← → ↺ [apex.oracle.com/pls/apex/doyensys/Doyensys/Details](https://apex.oracle.com/pls/apex/doyensys/Doyensys/Details)

```
{
 "items": [
 {
 "empno": 7839,
 "ename": "KING",
 "job": "PRESIDENT",
 "mgr": null,
 "hiredate": "1981-11-17T00:00:00Z",
 "sal": 5000,
 "comm": null,
 "deptno": 10
 },
 {
 "empno": 7698,
 "ename": "BLAKE",
 "job": "MANAGER",
 "mgr": 7839,
 "hiredate": "1981-05-01T00:00:00Z",
 "sal": 2850,
 "comm": null,
 "deptno": 30
 },
 {
 "empno": 7782,
 "ename": "CLARK",
 "job": "MANAGER",
 "mgr": 7839,
 "hiredate": "1981-06-09T00:00:00Z",
 "sal": 2450,
 "comm": null,
 "deptno": 10
 },
 {
 "empno": 7566,
 "ename": "JONES",
 "job": "MANAGER",
 "mgr": 7839,
 "hiredate": "1981-04-02T00:00:00Z",
 "sal": 2975,
 "comm": null,
 "deptno": 20
 },
 {
 "empno": 7788,
 "ename": "SCOTT",
 "job": "ANALYST",
 "mgr": 7566,
 "hiredate": "1982-12-09T00:00:00Z",
 "sal": 3000,
 "comm": null,
 "deptno": 20
 },
 {
 "empno": 7902,
 "ename": "FORD",
 "job": "ANALYST",
 "mgr": 7566,
 "hiredate": "1981-12-03T00:00:00Z",
 "sal": 3000,
 "comm": null,
 "deptno": 20
 },
 {
 "empno": 7369,
 "ename": "SMITH",
 "job": "CLERK",
 "mgr": 7902,
 "hiredate": "1980-12-17T00:00:00Z",
 "sal": 800,
 "comm": null,
 "deptno": 20
 },
 {
 "empno": 7499,
 "ename": "ALLEN",
 "job": "SALESMAN",
 "mgr": 7698,
 "hiredate": "1981-02-20T00:00:00Z",
 "sal": 1600,
 "comm": 300,
 "deptno": 30
 },
 {
 "empno": 7521,
 "ename": "WARD",
 "job": "SALESMAN",
 "mgr": 7698,
 "hiredate": "1981-02-22T00:00:00Z",
 "sal": 1250,
 "comm": 500,
 "deptno": 30
 },
 {
 "empno": 7654,
 "ename": "MARTIN",
 "job": "SALESMAN",
 "mgr": 7698,
 "hiredate": "1981-09-28T00:00:00Z",
 "sal": 1250,
 "comm": 1400,
 "deptno": 30
 },
 {
 "empno": 7844,
 "ename": "TURNER",
 "job": "SALESMAN",
 "mgr": 7698,
 "hiredate": "1981-09-08T00:00:00Z",
 "sal": 1500,
 "comm": 0,
 "deptno": 30
 },
 {
 "empno": 7876,
 "ename": "ADAMS",
 "job": "CLERK",
 "mgr": 7788,
 "hiredate": "1983-01-12T00:00:00Z",
 "sal": 1100,
 "comm": null,
 "deptno": 20
 },
 {
 "empno": 7900,
 "ename": "JAMES",
 "job": "CLERK",
 "mgr": 7698,
 "hiredate": "1981-12-03T00:00:00Z",
 "sal": 950,
 "comm": null,
 "deptno": 30
 },
 {
 "empno": 7934,
 "ename": "MILLER",
 "job": "CLERK",
 "mgr": 7782,
 "hiredate": "1982-01-23T00:00:00Z",
 "sal": 1300,
 "comm": null,
 "deptno": 10
 }
],
 "first": "https://apex.oracle.com/pls/apex/doyensys/Doyensys/Details"
}
```



# ABOUT DOYENSYS

---



Doyensys, started in December 2006, is a rapidly growing Oracle technology-based solutions company located in the US with offshore delivery centers in India.

We specialize in Oracle e-Business Suite, Oracle Cloud, Oracle APEX Development, Oracle Fusion, Oracle Custom Development, Oracle Database, and Middleware Administration.

We provide business solutions using cutting-edge Oracle technologies to our customers all over the world. Doyensys uses a viable Global Delivery Model in deploying relevant and cost-effective solutions to its clients worldwide. A winning combination of technical excellence, process knowledge, and strong program management capabilities enables Doyensys achieve global competitiveness by making technology relevant to its customers.

## ABOUT DOYENSYS

We improve business efficiencies through innovative and best-in-class Oracle-based solutions with the help of our highly-equipped technical resources. We are an organization with a difference, which provides innovative solutions in the field of technology with Oracle products. Our clientele across the globe appreciate our laser focus on customer delight, which is our primary success parameter. We have more than 250 resources across the globe. The technical capability of Doyensys stands out from the crowd as we not only provide services of exceptional quality for various Oracle products on time but also take credit for having developed our own products such as DBFullview, EBIZFullview, DBIMPACT, SmartDB, etc.

Our customers are fully satisfied with our services and appreciate our work as we stretch beyond their expectations. We do not compromise on quality for delivery, and the policies of Doyensys revolve around PCITI [Passion, Commitment, Innovation, Teamwork, and Integrity].

Doyensys encourages its employees to participate in Oracle conferences across the globe, and our team has presented papers at various conferences such as AIOUG Sangam, OATUG Collaborate over the years.

The exemplary work of Doyens as a team has created a wonderful environment in the organization. The policies framed by the management are very flexible and employee-friendly, keeping in mind the growth and interest of the organization.



We received 'India's Great Mid-size Workplaces' award (Rank #19) based on the feedback given by our employees in strict confidence and evaluation of various parameters

by Great Place to Work. We are an equal opportunity employer and do not discriminate based on sex, religion, gender, nationality, etc. Our women are given a lot of flexibility to work in the organization, understanding the time that they need to spend with their family.



We are also proud to share that we received the award 'Best Workplaces for Women' from Great Place to Work and were ranked among the top 75 in IT and BPM Best Workplaces.



**Among the TOP 75 of India's Best Workplaces in IT & IT-BPM 2019**

The culture to excel is at the heart of everything Doyens do. We not only share and care for other folks within the organization but also for the folks around the globe.

We have Database and Oracle EBS blogs available on the Doyensys page and are accessible on the internet. These blogs are exemplary work done by Doyens from the

## ABOUT DOYENSYS

knowledge and experience gained by supporting various customers across the globe. There is a habit of creating reusable components for the teams within Doyensys so that a similar piece of work can be helpful for some other project within the organization.

The management is very supportive and encouraging, which is very much visible from the awards [Passion and Commitment, Commitment and Customer Delight, Rookie of the Year] that are given to Doyens, who excel in various categories.

Doyensys is not only a great place to work but is also a great place to learn as employees are always encouraged to explore new technologies and suggest innovative ideas that can benefit the clients. The teams within Doyensys are always encouraged and recognized by the management to add value to the work that is delivered to the customer rather than just doing monotonous work.



## ABOUT THE AUTHORS

---

### RAJAN CHANDRU



Rajan Chandru, a co-author of this book, lives in Chennai and is a Senior Consultant at Doyensys Pvt. Ltd. He is an Oracle Certified Associate with over eight years of rich experience in application development using Oracle Application Express and software development lifecycle. He has worked from APEX 3.0 version and developed applications for financial and pharmaceutical industries on different applications and was involved in data migration and implementation projects.

### SATHYA JEGAN



Sathya Jegan, a co-author of this book, is a Senior Consultant at Doyensys Pvt. Ltd., and is an Oracle Certified Associate. He has over eight years of experience in application development using Oracle Application Express and has worked from APEX 3.0 version, and developed applications for pharmaceutical industries and worked on projects for managing database involved in data migration and implementation projects

## **KARTHIK SENTHILMURUGAN**



Karthik Senthilmurugan is one of the co-authors of this book. He is a Consultant at Doyensys Pvt. Ltd. He has more than four years of experience in application development using Oracle Application Express and has worked from APEX 4.2 version and developed applications for foundry management and pharmaceutical industries on different applications and was involved in data migration and implementation projects.

## **PANDIMUNEESWARAN KALIRAJAN**



Pandimuneeswaran Kalirajan is one of the co-authors of this book. He is a Consultant at Doyensys Pvt. Ltd. He has over five years of experience in application development using Oracle Application Express and has worked from APEX 4.2 version, Oracle forms and reports, and developed applications for foundry management and pharmaceutical industries on different application and was involved in data migration and implementation projects.