Bootcamp TOC

|  |  |
| --- | --- |
| 1 | UNIX and Linux Essentials - Classroom (Campus Only) |
| 2 | XML Fundamentals - Classroom (Campus Only) |
| 3 | Oracle Database 19c: SQL workshop - Classroom (Campus Only) |
| 4 | Fundamentals of Java 17 - Classroom (Campus Only) |
| 5 | JPA with Eclipselink - Classroom (Campus Only) |
| 6 | Oracle WebLogic Server 14c: Administration I - Classroom (Campus Only) |
| 7 | REST Services & Webservices - Classroom (Campus Only) |
| 8 | Micro Services Architecture, Spring Boot features, Kafka (Basics) - Classroom (Campus Only) |
| 9 | Developing Web Applications with JavaScript, HTML5, and CSS  - Classroom (Campus Only) |
| 10 | Ojet and Angular JS, Node JS and Knockout JS - Classroom (Campus Only) |
| 11 | Introduction to Devops - Classroom (Campus Only) |

* Unix
* XML
* Oracle Database
* Java 17
* Eclipse Link for JPA
* Weblogic Server
* WebService
* Microserivce
* Web - HTML, CSS & Javasccript
* OJET
* DevOps

Case Study -> Incremental based approach

Unix

* It is a powerful OS which can be connected by multiple users at a time
* Terminal Based

:wq : Write & Quit

:q : Quits without saving

i : When you open vi editor, type “i” to insert the content

cp: to copy

mv: to rename

touch : to create files

ls : to list

vi : to edit

echo : to print

cat > : to overwrite

cat >> : to append

Search for below software in oracle-emp

1. Git -> present
2. JDK 17 -> present
3. Eclipse IDE for enterprise ->
4. Oracle Database 19c -> You must use pc name which is less than 14 characters & while installing you must use mobile hotspot
5. Weblogic Server
6. VS Code

Install Git -> Git can also be used to try all the Unix related commands, the main purpose of GIT is to collaborate | integrate everyone’s work

XML: Extensible Markup Language, it is mainly used to transform the data as well to configure the applications

Transforming: One Application can share the data in XML and another application parses the XML and converts to the structure it understands

App1 (Java) XML App2 (C#)

Application configuration: When the application want to connect to the database or want to configure server properties then they can use XML as it’s structure is strict.

ex:  
<server>  
 <port>9090</port>  
 <context-path>/api</context-path>  
 <protocol>HTTP</protocol>  
</server>

Rules of XML

1. An XML file must use tags
2. You need to have only one root tag
3. Tags must nest properly, below XML is invalid because of improper nesting

<server>  
 <port>9090 <context-path>/api</port></context-path>  
</server>

1. You can create your own XML tags & can also define/declare those tags in a schema

Two types of XML

1. Well formed that follows all the basic rules like proper nesting, case sensitivity, only one root tag
2. Valid XML: this follows Well formed + XML schema

XML Schema are of 2 types

1. DTD - Document Type Definition - Plain text syntax
2. XSD - XML Schema Definition - Uses XML syntax

XML Schema: These are the rules that declares the XML elements & their attributes, so that you can only use the tags & attributes declared in the Schema

Syntax of DTD:

<!ELEMENT element\_name (#PCDATA or child\_element) >  
<!ATTLIST element\_name attribute\_name value\_type some\_option >

PCDATA: Parsed Character Data -> You can’t use any reserved symbols here  
value\_type of an attribute can be CDATA: Character Data  
some\_option: can be #REQUIRED, #IMPLIED, #FIXED

#REQUIRED: Mandatory

#IMPLIED: Optional

#FIXED: Constants like bank\_name, college\_name,

Wild cards like: \*, +, ? would specify how many times a tag can repeat

“\*” : 0 or more  
“+” : 1 or more  
“?” : 0 or 1

XSD: XML Schema Definition, It is also used to apply schema rules, but it has wider range of types & occurrences, the rule is done using UI

XSD has many types like

* integer, long, float, string
* complex type : a tag that has child tags

RDBMS with Oracle 19C

1. What is RDBMS
2. Keys : Primary key, Composite Key, Candidate Key, Foreign Key, Super Key
3. Normalization in the database, 1NF, 2NF, 3NF
4. ER Model - Understand how to create ER diagrams

Oracle database gives us HR schema in an HR account, which has some tables which you can use to test & learn SQL

By default HR account is locked, you need to unlock.

Root user login: / as sysdba or connect / as sysdba

You need to login as a root user & alter the HR account.

Creating a user in oracle database

Note: You must be a root user, when you create a user it will not have any privilege to login or create any database resource.

Syntax:

create user username identified by password;

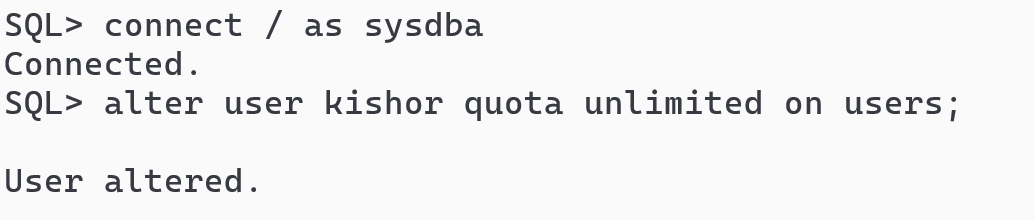
grant connect, resource to username; # this will give permission to login & create database resources

Example:

create user kishor identified by Kishor123;

In ORCLPDB, you need to set the container before you perform any task

alter session set container=orclpdb;



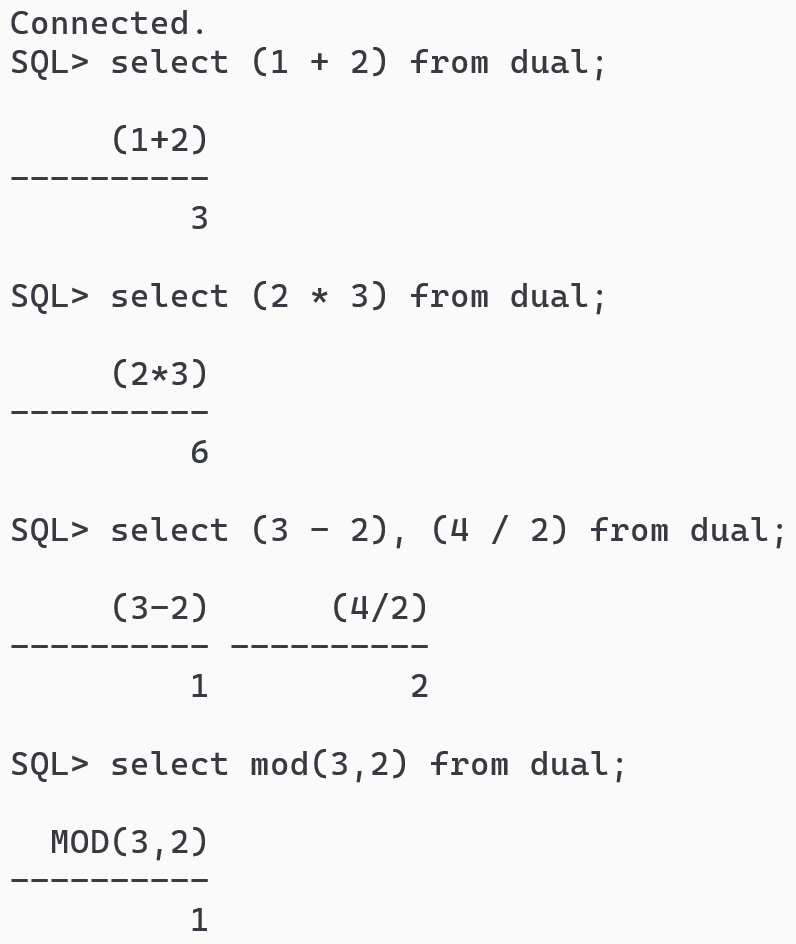
Summary:

1. Logged in as root user: / as sysdba or connect / as sysdba
2. Created the user, grant the permission to login and also set the quota
3. Unlocked the HR schema and grant the permission to login
4. We used user\_tables to identify the tables created by the logged in user

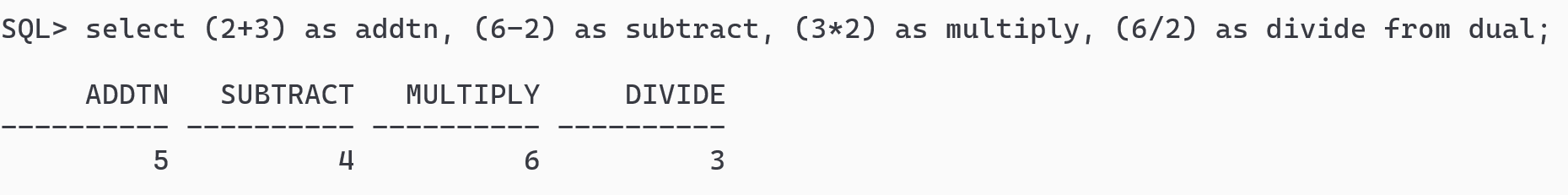
DUAL table

In oracle database it is a table used to get some guaranteed results of one row on any operation

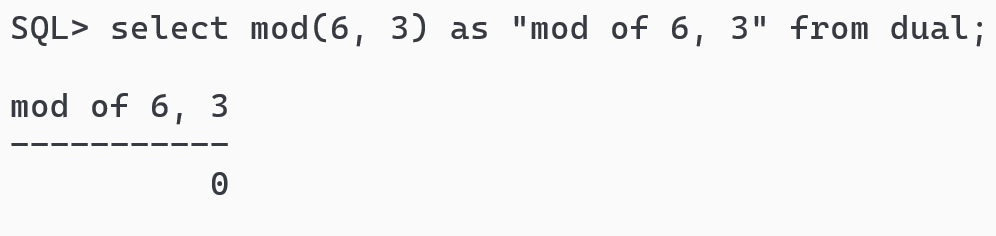
Trying some arithmetic operations using DUAL table



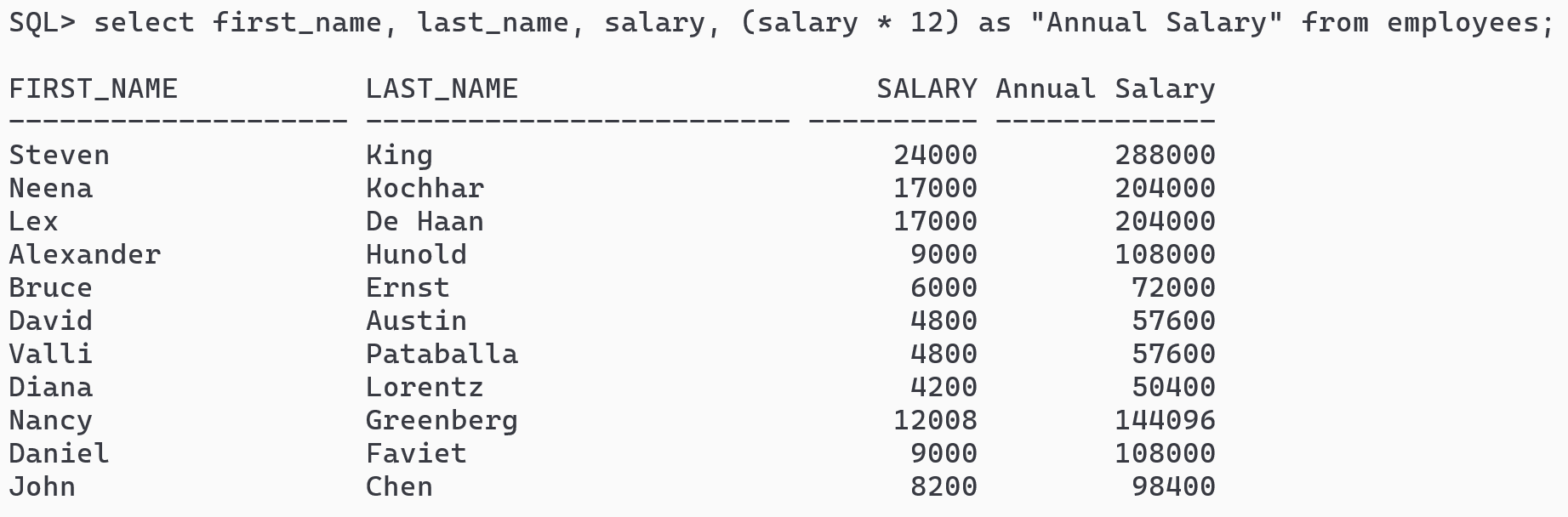
Using alias names to the columns



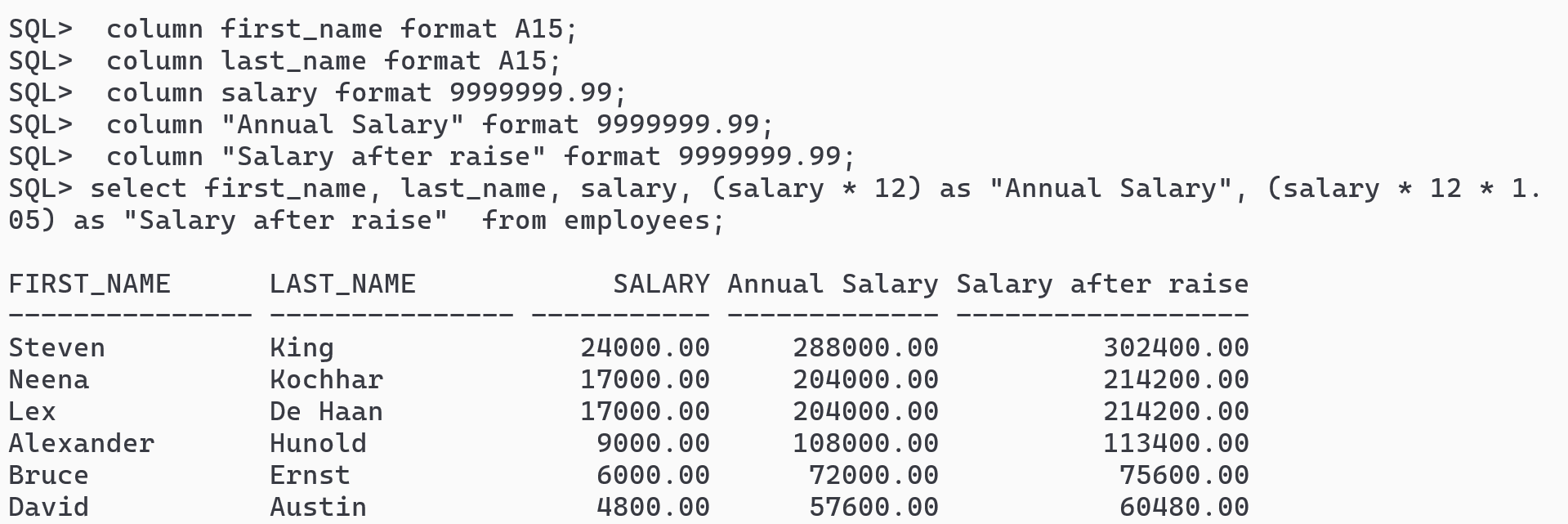
You can use double quotes to provide alias names having multiple words



Applying select & arithmetic operation on HR.EMPLOYEES table



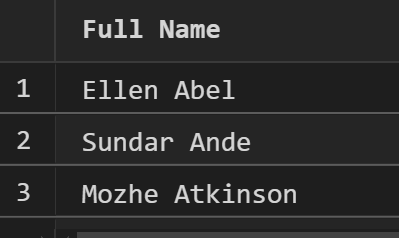
Display the result which gives incremented salary of 5% to the annual salary



Concatenation in oracle

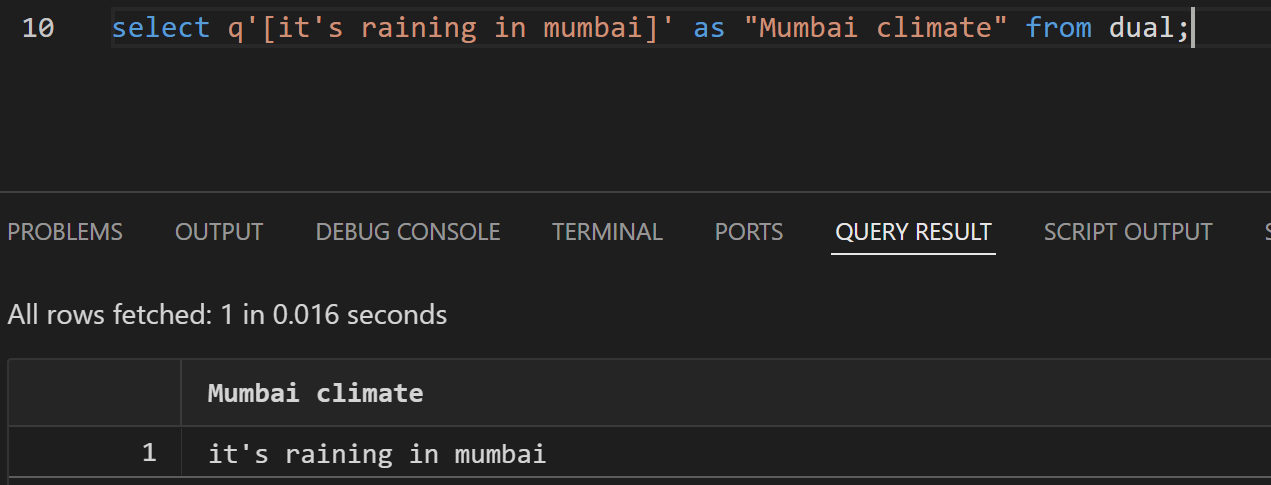


Output:



Alternative quote operator in oracle

To avoid confusions while using lot of quotes in a string you can use q’[strings]’



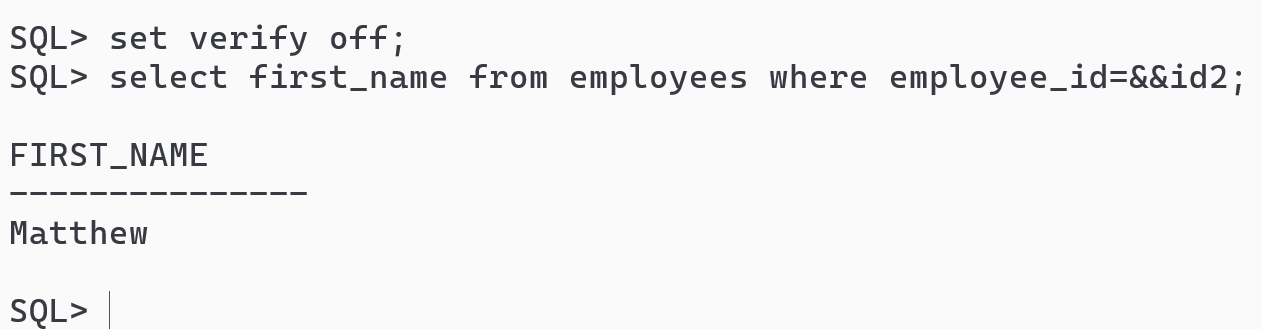
& and &&

& asks for input each time

&& asks for input only once & reuses the value, you can change the value with define.



If you wish to not see the old & new query you can use set verify off and change to on, if you wish to see the old & new query use set verify on



SQL Functions

In Oracle, **SQL functions** are built-in operations that let you **manipulate data** in queries. They're divided into two broad categories:

1. Single-Row functions: Operate on **one row at a time** and return **one result per row**.
2. Aggregate functions: Operate on **multiple rows** and return a **single result per group** or for the whole table.

Single row functions

Character functions: upper, lower, initcap, substr, replace, length, trim, concat

Numeric functions: trunc, round, floor, ceil, sqrt, abs, mod

Conversion functions: to\_char, to\_date

Date functions: sysdate, current\_date, current\_timestamp, add\_months, months\_between

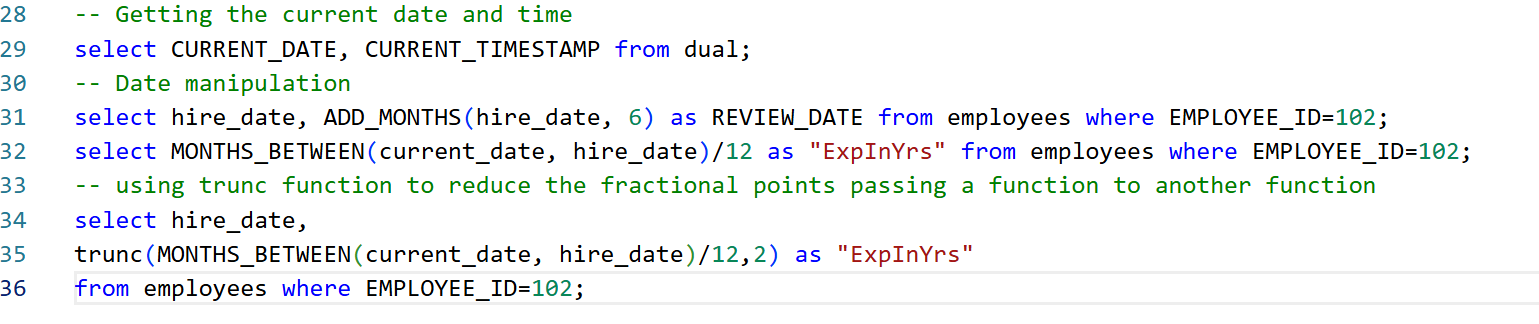
Note: sysdate, current\_date, current\_timestamp are pseudo functions, which doesn’t take parentheses

ex: select current\_date from dual;

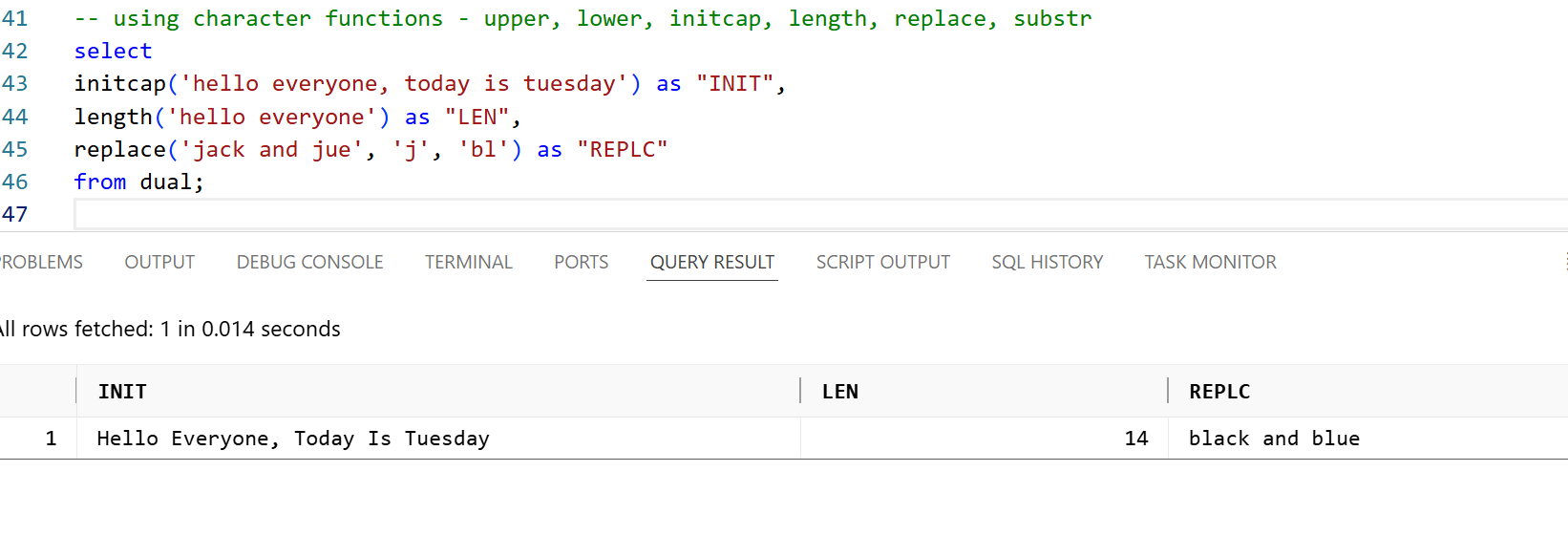
wrong: select current\_date() from dual;

sysdate & current\_date gives the same result, however sysdate uses system clock to get the date, current\_date uses the time zone of the user logged in to get the date

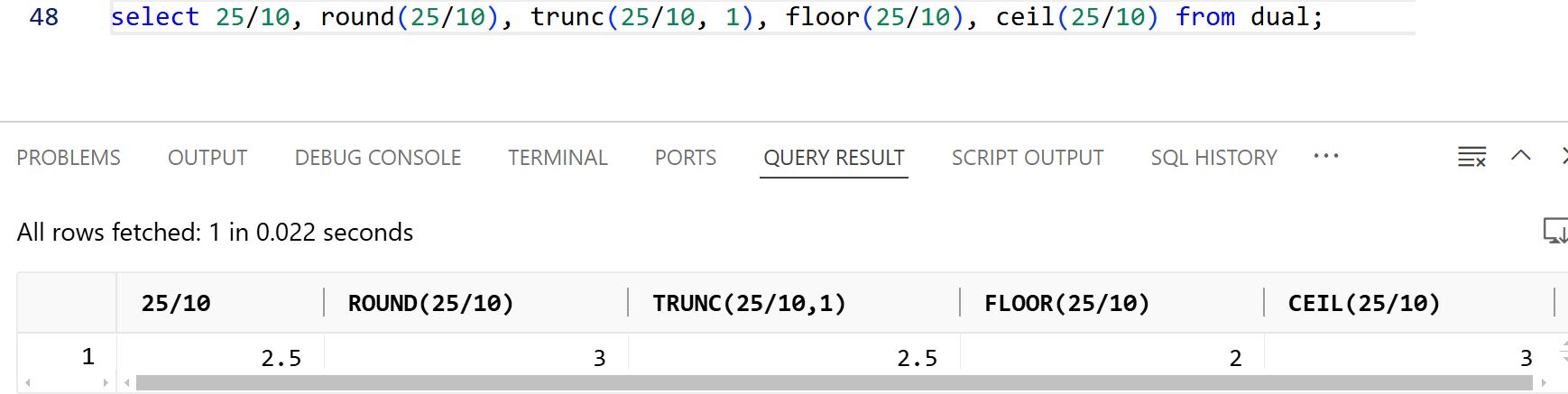
Using Date



Using Character functions



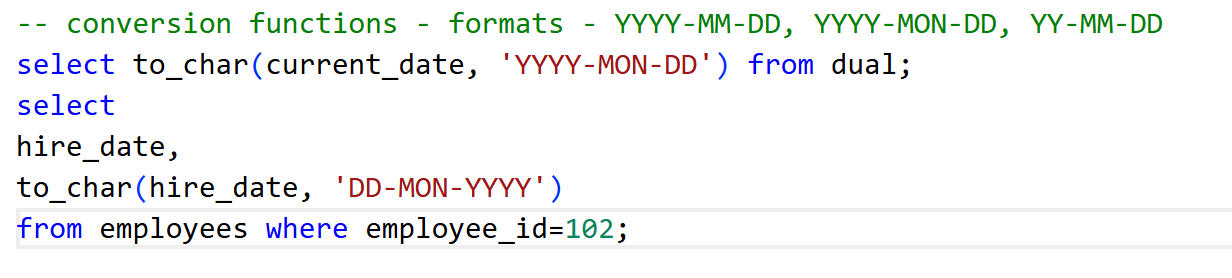
Numeric functions



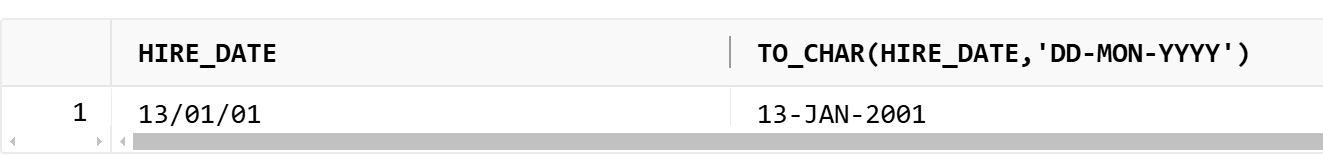
Conversion functions

to\_char: converts date to a string

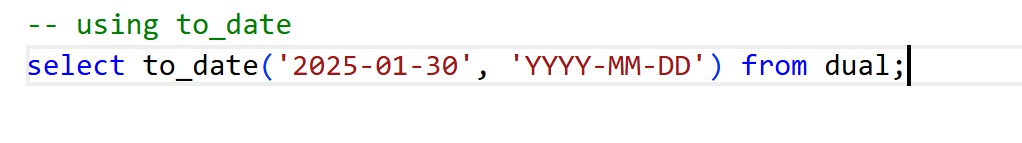
to\_date: converts string to a date



Output:



to\_date: It converts string to date



ISO-Date format

Globally ISO is the standard format which is used by most of the modern applications, the ISO format is dd-MM-yyyy, but oracle is not following ISO format

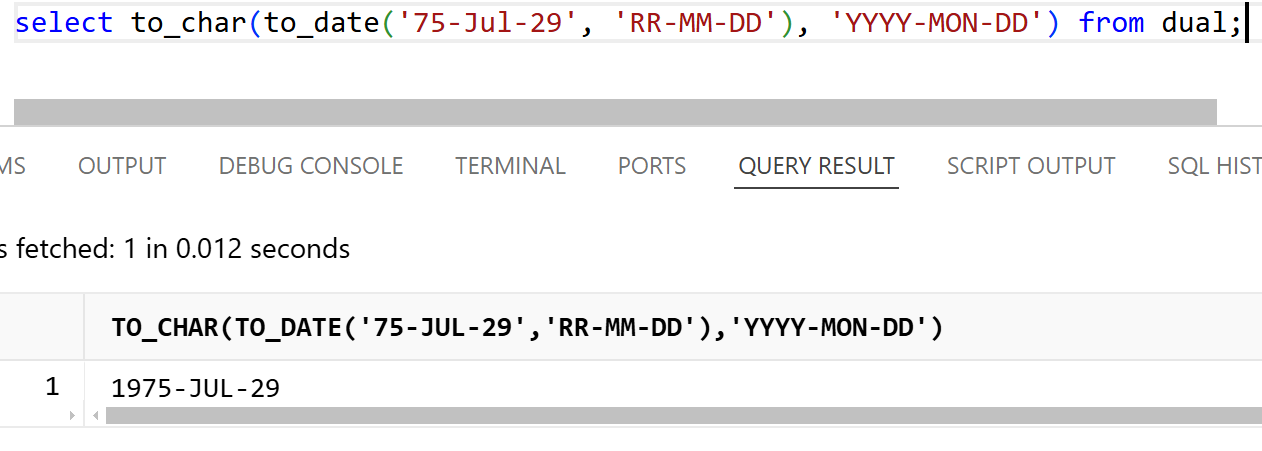
RR date format

The **RR date format** in Oracle is a **two-digit year format** that helps Oracle **guess the correct century** (1900s or 2000s) when you're working with dates.

It is especially useful when dealing with **legacy data** or **Y2K-type logic**, where users may enter only two digits for the year.

0 to 49 interpreted as 2000 to 2049

50 to 99 interpreted as 1950 to 1999

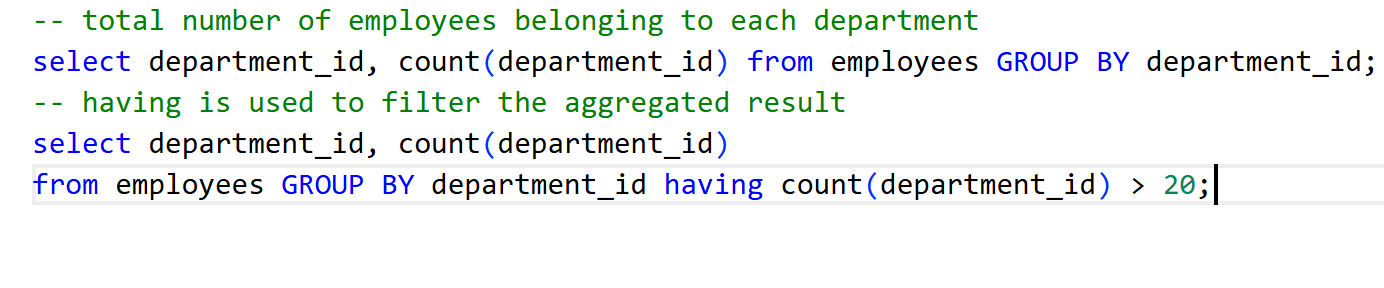


Aggregate functions

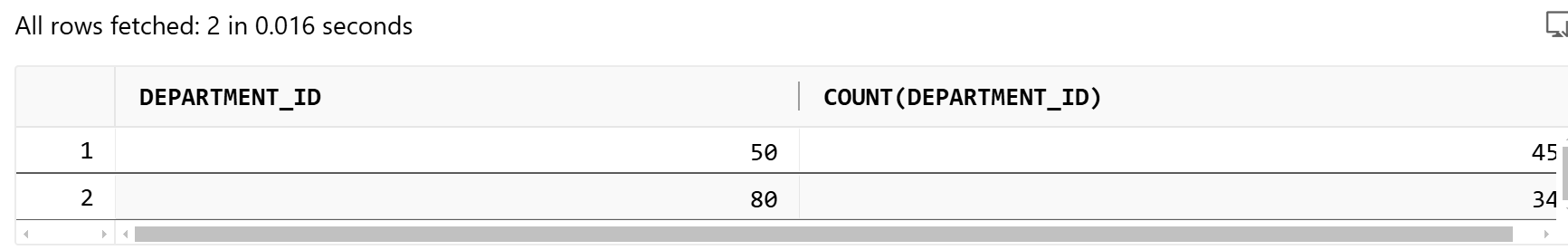
Performs operations on multiple rows & return single result

* count
* sum
* avg
* max
* min

Total number of employees belonging to each department



OUTPUT:



Activities

* Self learning till Chapter - 6
* Keep the queries in the script file.
* Create another script file that will have corresponding MYSQL queries for the queries you try in OracleDB

Evaluation at 4PM

SQL JOINS

JOINS are used to combine the rows from two or more tables based on the JOIN condition.

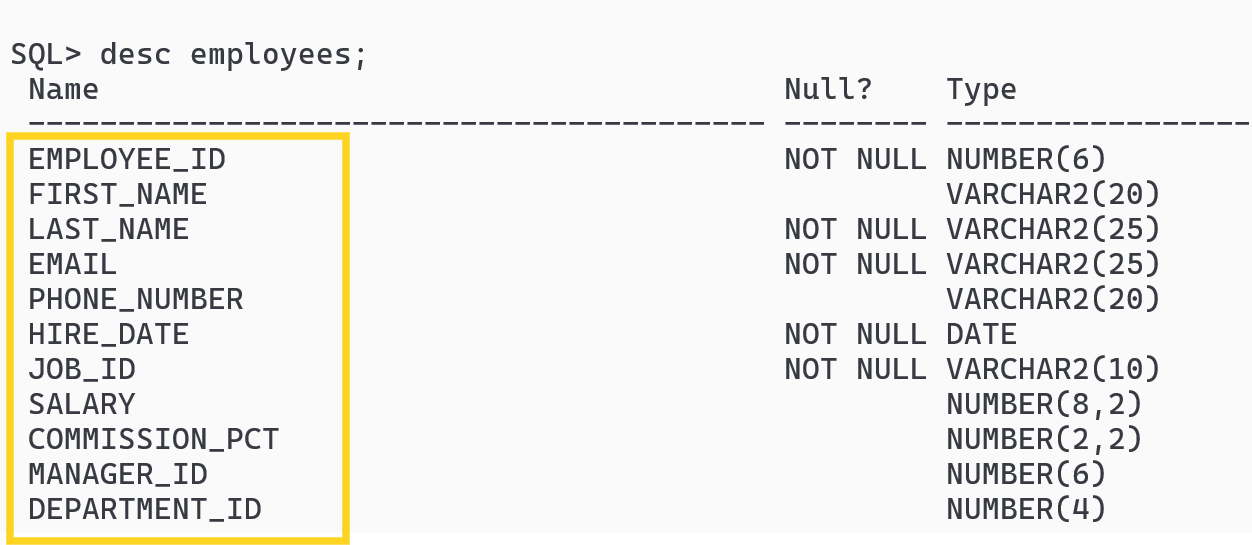
Types of JOINS

1. INNER JOIN
2. LEFT JOIN / LEFT OUTER JOIN
3. RIGHT JOIN / RIGHT OUTER JOIN
4. FULL OUTER JOIN
5. CROSS JOIN
6. SELF JOIN
7. NATURAL JOIN.

To JOIN tables we must have a clear picture about the table columns, let us find columns of below tables

1. EMPLOYEES
2. DEPARTMENTS
3. LOCATIONS
4. JOBS
5. JOBS\_HISTORY

EMPLOYEES

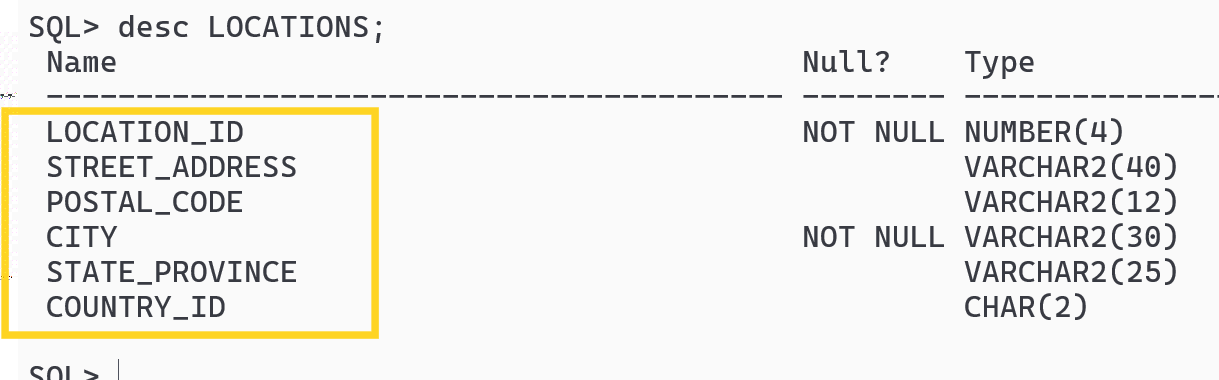


DEPARTMENTS



Common columns between EMPLOYEES & DEPARTMENTS are: DEPARTMENT\_ID, MANAGER\_ID

LOCATIONS



Common columns between DEPARTMENTS and LOCATIONS are: LOCATION\_ID

JOBS



Common columns between EMPLOYEES and JOBS are: JOB\_ID

JOBS\_HISTORY



Common columns between EMPLOYEES & JOB\_HISTORY are: EMPLOYEE\_ID, DEPARTMENT\_ID, JOB\_ID

Common columns between JOBS & JOBS\_HISTORY are: JOB\_ID

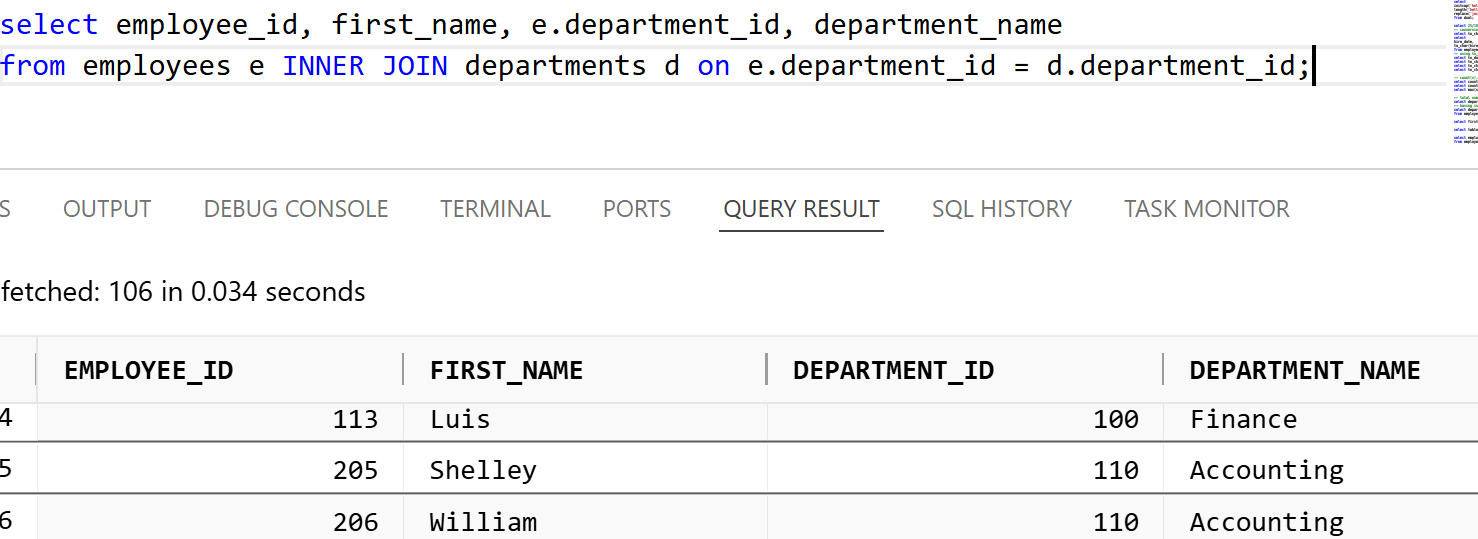
INNER JOIN

It returns only the matching rows from left and right tables based on the JOIN condition.

Syntax: select column1, column2 from table1 INNER JOIN table2 ON condition;

Note: There could be chance that columns with the same name in both the names, hence you need to use table aliases or full table names to select the columns, else you will get ambiguous column name error.

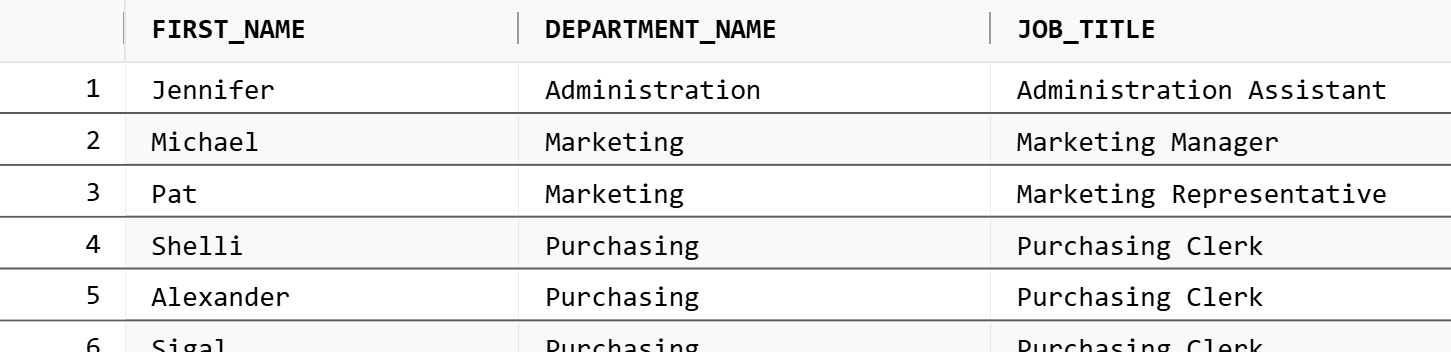
INNER JOIN on employees and departments



INNER JOIN: EMPLOYEES, JOBS and DEPARTMENTS to display each employees JOB title and their department names.

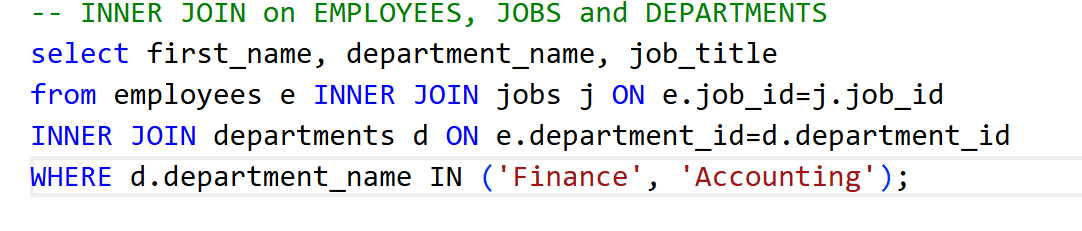


OUTPUT:

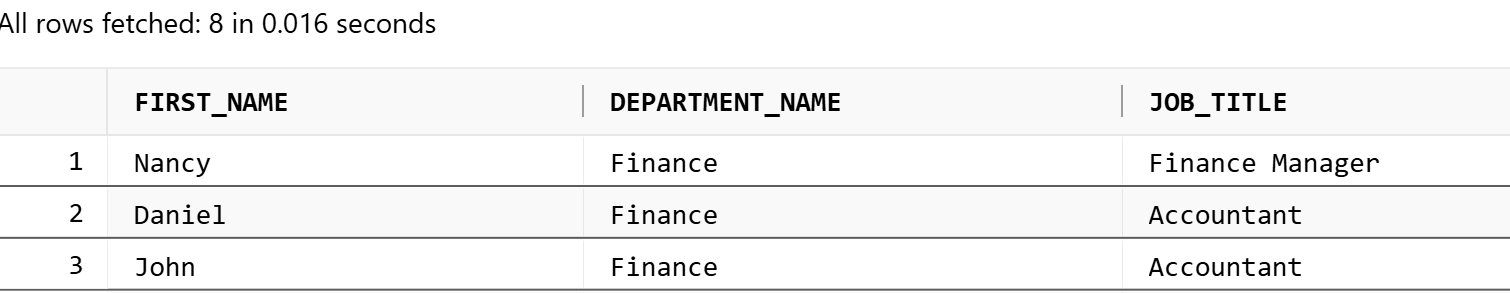


Applying filter condition using WHERE clause

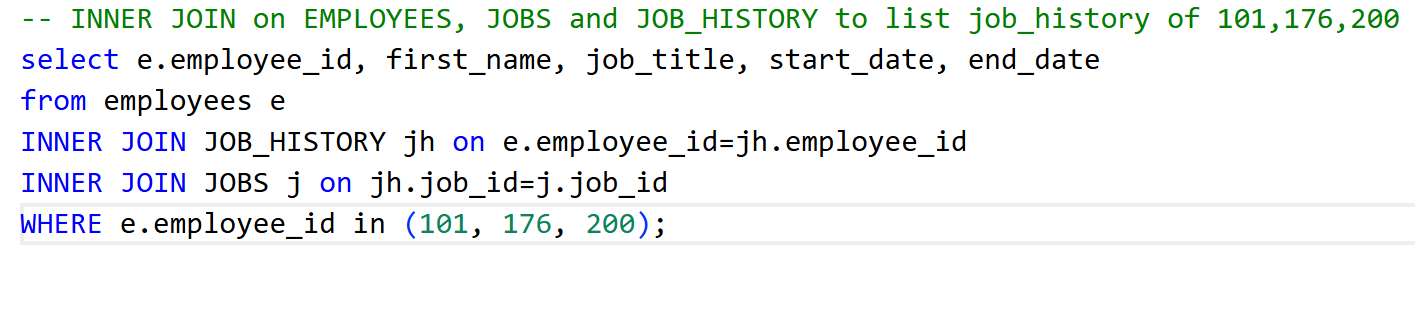
ON is used to apply condition to JOIN, WHERE is used to apply condition to filter



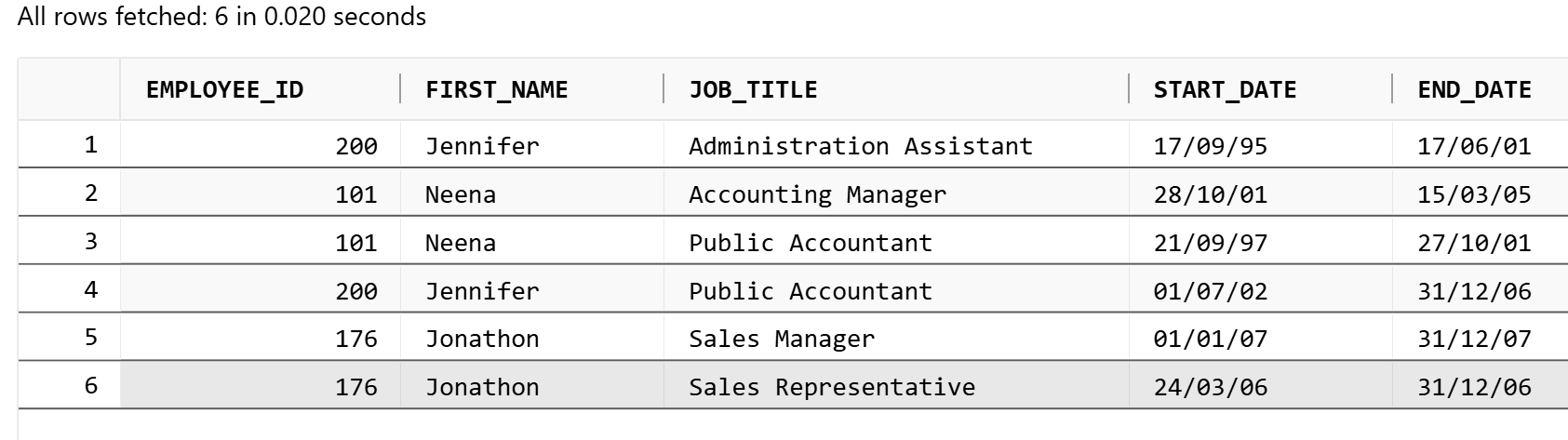
OUTPUT



Using EMPLOYEES & JOB\_HISTORY list the EMPLOYEES job\_history which will have employee\_id, first\_name, job\_title, start\_date, end\_date, filter the join that matches to the list of employee\_id: 200, 101, 176



Output



LEFT OUTER JOIN

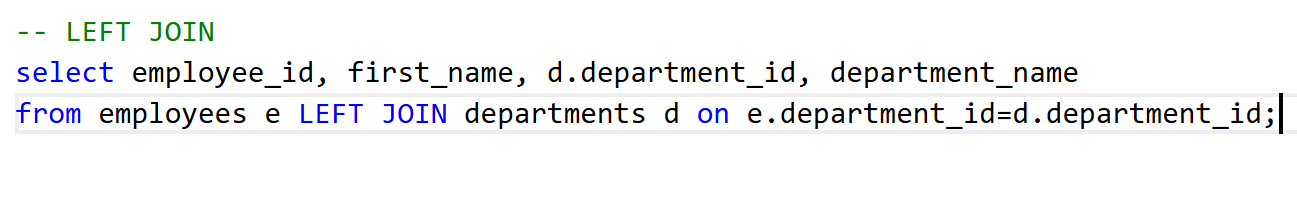
It is used to list all the rows of the LEFT table and only the matching rows of the RIGHT table.

Sometimes you want to list the employees who are not assigned to any department

syntax:

select column(s) from table1 LEFT JOIN table2 on condition

List employees & departments with LEFT JOIN based on the department\_id



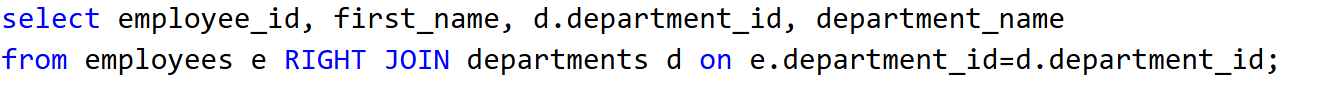
RIGHT OUTER JOIN

It is used to list all the rows of the RIGHT table and only the matching rows of the LEFT table.

Sometimes you want to list all the employees who are assigned to the department and orphan departments

syntax:

select column(s) from table1 RIGHT JOIN table2 on condition



Try the following JOINS and understand its working

1. FULL OUTER JOIN: Combines the results of left & right table
2. CROSS JOIN: Gives the cartesian product of left & right table
3. SELF JOIN: Joins the same table, this is used to list the relationship present in the same table
4. NATURAL JOIN: It uses the column names having the same name to JOIN

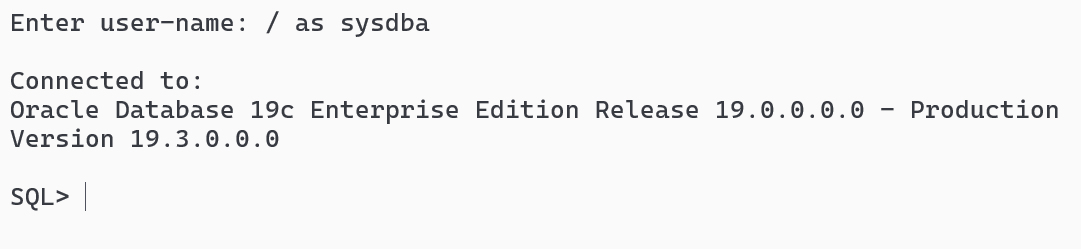
Types of SQL

1. DRL: Data Retrieval Language - SELECT, DESC
2. DDL: Data Definition Language - CREATE, ALTER, DROP, TRUNCATE
3. DML: Data Manipulation Language - INSERT, UPDATE, DELETE
4. TCL: Transaction Control Language - COMMIT, ROLLBACK
5. DCL: Data Control Language - GRANT, REVOKE

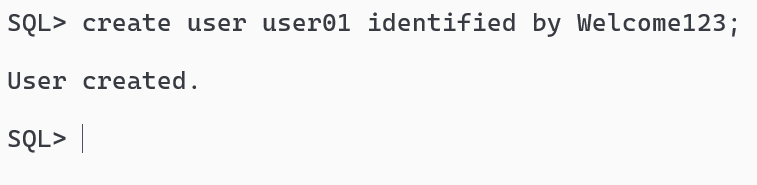
Creating user & giving connect, resource privilege

Steps to create users & give connect privilege

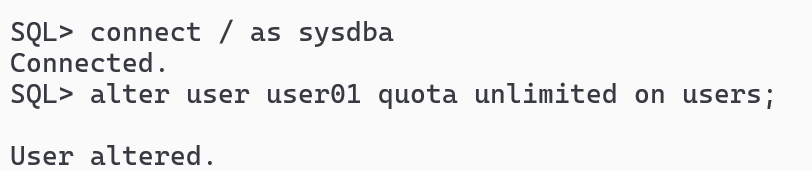
Connect / as sysdba



Create a user identified by password



Set the quota else you can’t store records

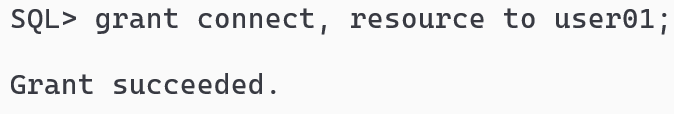


If you are using PDB then you need to first set the session

alter session set container=orclpdb;

Then you must able to create the user

Grant connect, resource permission to the user

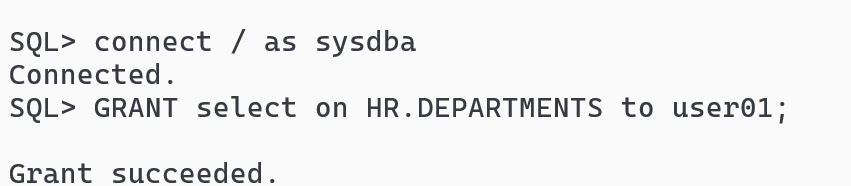


Now you must able to login as user01

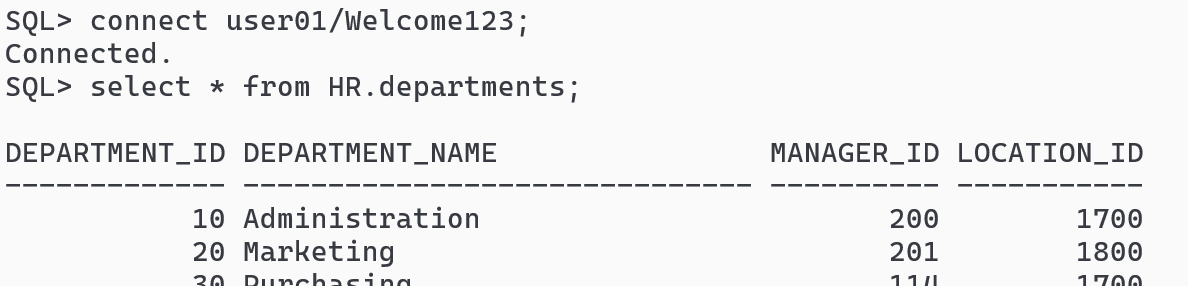


Note: The user having orclpdb must connect using user01/Welcome123@orclpdb

Can user01 access tables present in another schema



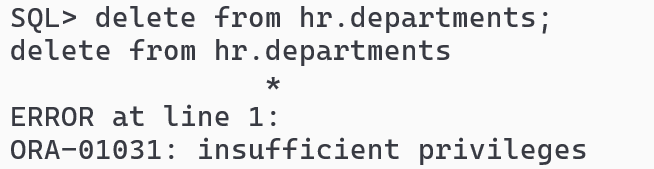
Now login as user01 and enter select command on HR.DEPARTMENTS



NOTE: If you give SELECT privilege, you can use DESC command also.

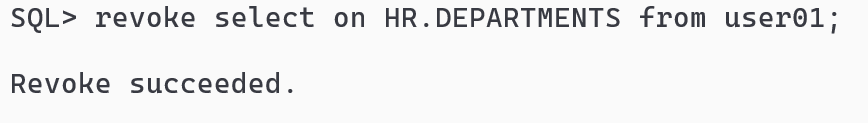
Try to access EMPLOYEES table of HR from user01, if you can’t access, give permission to user01 to read the EMPLOYEES table data.

delete from HR.departments;



How to revoke the permissions from the user

Syntax: REVOKE privilege(s) on table\_name from user\_name;



DDL

It is used to manipulate the structure of the table like create, alter, truncate & drop, you can’t rollback DDL statements i.e., you can’t undo

CREATE syntax

create table table\_name (column\_name type [constraint], ….)

Constraint: These are the rules which you can apply on a table or a column, we have the following constraints

1. PRIMARY KEY
2. UNIQUE
3. NOT NULL
4. CHECK
5. FOREIGN KEY

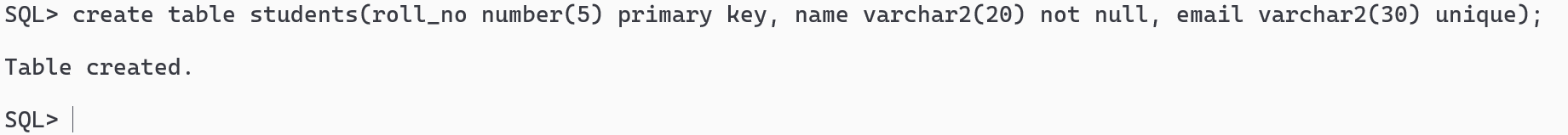
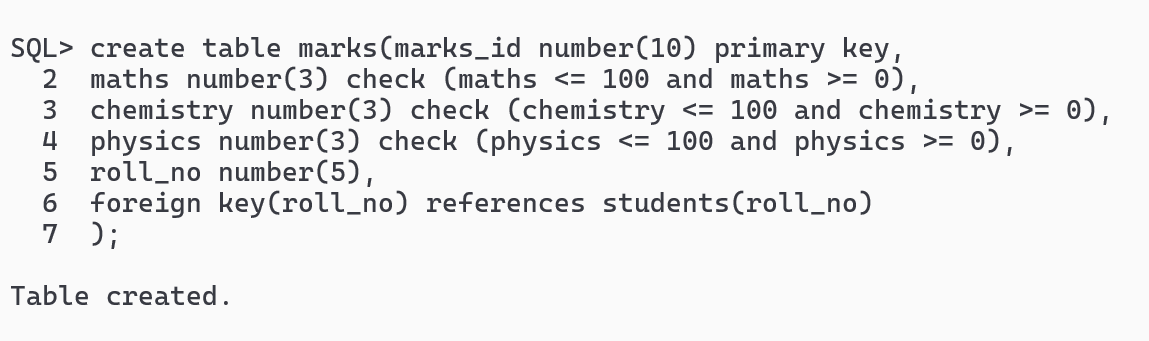


Table with CHECK & FOREIGN KEY



DML Statements

INSERT command: Store the records

Syntax:

1. insert into table\_name (col1, col2,..) values(….);
2. insert into table\_name values(….);

Display the student and marks table using join and show the total marks and their average that will have 2 fractional points (ex: 70.54)

PL/SQL

Procedural Language is a program you can store in the database and execute it whenever you need, you store SQL statements in PL/SQL, these are useful when admins want to run SQL commands everyday without rewriting the queries

There are three types of PLSQL block you can create

1. anonymous block: Doesn’t have a name, but useful to run procedures and functions
2. procedural block: Has a name
3. functional block: Similar to procedural block, but it has to mandatorily return a value

Syntax of Anonymous block

DECLARE [optional]  
 <<declaration-statement>>  
BEGIN  
 <<pl-sql statements>>  
END;

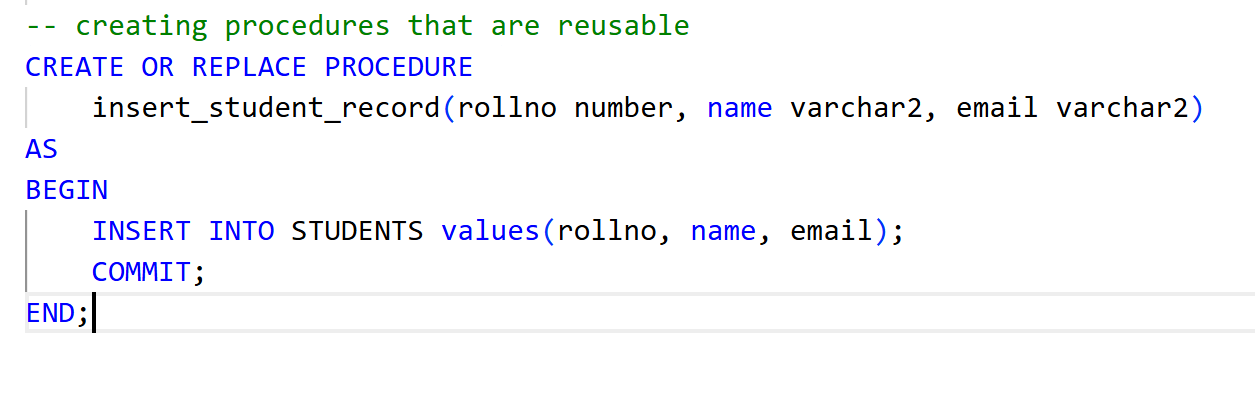
Syntax of Procedural block

CREATE OR REPLACE PROCEDURE procedure\_name(arguments)  
AS  
 <<declaration-statement>>  
BEGIN  
 <<sql statements>>  
END;

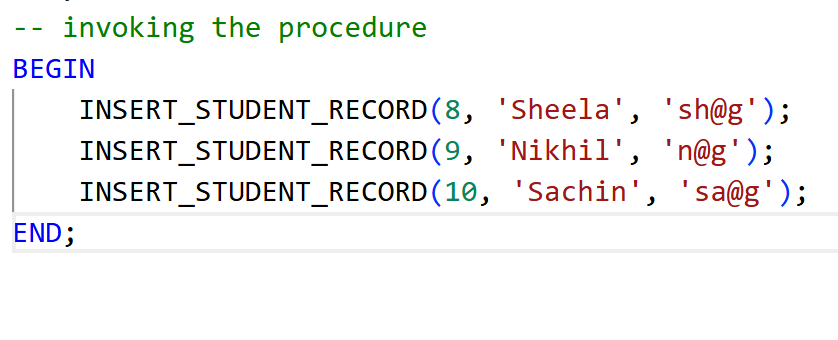
Syntax of Functional block

CREATE ORE REPLACE FUNCTION function\_name(arguments) return type  
AS  
 <<declaration-statement>>  
BEGIN  
 <<some-statements>>  
END;

Procedural block



If you must get compiled successfully to ensure procedure is created



Writing SELECT statement in PLSQL is a bit different, because you must assign the values of the columns to the PLSQL variables

SELECT column1, column2 INTO variable1, variable2 from table\_name where condition

Note: At a time you can access 1 record, if you want to access one or more records you must use FOR LOOP or CURSOR

For Self-Learning

Complete all the chapters in the TOC (Till Chapter 20)

In PLSQL write FOR LOOP to get all the student records

Java 17

Java is platform independent & object oriented programming language.

Java Fundamentals

1. Datatypes
2. Operators
3. Conditional Blocks
4. Looping Constructs
5. Arrays
6. Classes & Objects

Datatypes

They represent what kind of data a variable can store, they are of two types

1. Primitive datatypes - byte(1), short(2), int(4), long(8), float(4), double(8), char(2) & boolean(U)
2. Derived datatypes - arrays, classes, interfaces, enums

Package: It is a folder structure for your classes, its recommended to have atleast 2 level packages like com.oracle, org.examples and so on

Taking input from the keyboard

Scanner class is one of the widely class used in Java program to take input from the keyboard in console based program.

Scanner scan = new Scanner(System.in);

You can use various methods in Scanner like  
next(): to read a single word  
nextLine(): to read multiple words or Enter key  
nextInt(): to read an int  
nextDouble(): to read a double  
Note: There’s no method for character, you need to enter a String with next() method & use charAt(0) to extract the 1st character of a String.

Classes & Objects

What you can write inside the class

1. fields / variables / properties - it’s purpose is to store the values
2. constructors - same as the class name, shouldn’t have the return type, it’s purpose is to initialize the properties
3. methods - can have any name, but should have the return type, it’s purpose is to have logics that are reusable
4. nested classes - a class inside another class - use it only if its required, it’s purpose is to allow classes to be available only through outer class
5. Initializer blocks - instance block (IIB) & static block (SIB)

Principles of OOPs

1. Encapsulation - hiding the data
2. Inheritance - acquiring the properties & behaviors of an object form another object
3. Polymorphism
4. Abstraction

Encapsulation: It hides the data using private keyword and allows you to access the data using getters (reading) & setters (updating)

Inheritance: It is used to acquire properties & behaviors of a class from another class, it is achieved using extends keyword, it avoids redundant code in your application

Polymorphism: A method with many forms, there are two type of polymorphism

1. Method Overloading / Compile time Polymorphism
2. Method Overriding / Runtime Polymorphism / Dynamic Dispatch

Method Overloading

System.out.println(1);  
System.out.println(“HELLO”);  
System.out.println(500.0);  
System.out.println(true);

Method Overriding

Same method in super & sub class but different implementations

Self-Learning

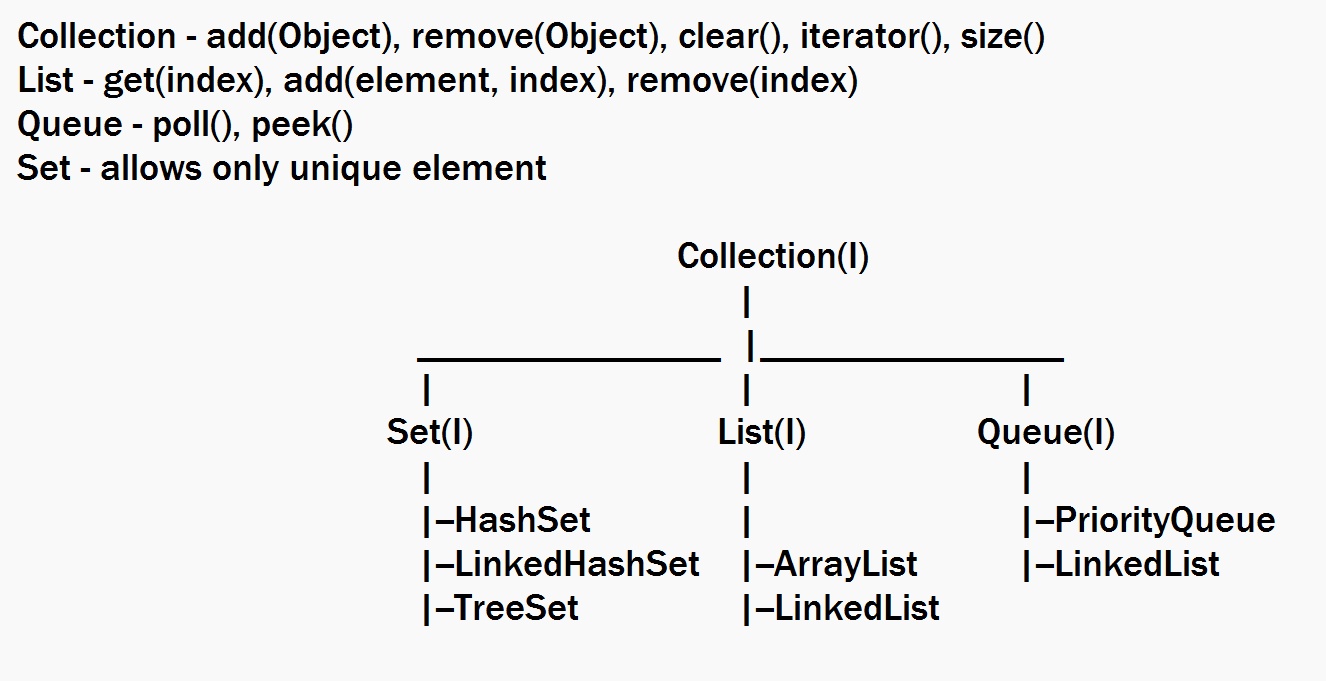
1. Understand interface & abstract class
2. Exception Handling
3. Factory Pattern - Creating factory methods
4. Complete till Day 4 topics

Note: Write code for all the topics with comments

Note: Collection framework try few interfaces & classes - I will also explain it tomorrow

Collection Framework

It provides set of interfaces & classes to maintain the data



HashSet, LinkedHashSet & TreeSet allow only unique elements.

HashSet - Maintains random order, it is faster while searching the elements

LinkedHashSet - Insertion order - FCFS - Online Ticket Reservations

TreeSet - Sorted order - Student results based on their rollNo

List - Maintains elements based on the index, it allows duplicate

ArrayList - Contiguous memory address - Retrieval is faster - Adding & Removing is slower

LinkedList - Non Contiguous memory address - Adding & Removing is faster, but slower retrieval

Queue - Mainly used to process the element by removing from the queue, it has poll() method to remove the elements in certain order

LinkedList - FIFO - poll() removes the element in FIFO

PriorityQueue - Sorted Order - poll() removes the element in Sorted Order

Collection framework is dynamic in nature, means it increases/shrinks the memory at runtime based on the data you add/remove, all the APIs are part of java.util package

Collection support any type of data in a single container which is not recommended

List list2 = new ArrayList(); // non-generic type  
list2.add(20);  
list2.add(45.4);  
list2.add(true);  
list2.add(LocalDate.parse(“2025-08-06”));  
list2.add(new Student(…..));

// Generic type

List<Student> list3 = new ArrayList<Student>(); or new ArrayList<>();  
list3.add(new Student(….));  
list3.add(20); // compilation error

Generic supports collection to have same type of data, which is type safe

Collection<T> : add(T)  
Collection<String> : add(String)  
Collection<Student> : add(Student)

Set when it maintains complex type

Set invokes hashCode() of the object to store them in the HashSet [ hash buckets ], if the hashCode() is same, then the object is added to the same hash bucket & it invokes equals() method to compare the objects, if they are same then they are identified as duplicate else considered as unique

Comparable<T>: This interface has comparTo(T) method that compares two elements and return -1, 0 or +1, using these int values TreeSet sorts the element, it is used as a default sorting technique.

Comparator<T>: This interface has compare(T t1, T t2) method that can customize the sorting, you must use this when you don’t want the default sorting mechanism.

Note: Comparator is implemented outside the class with anonymous class or lambda expression

If Comparator has one abstract method i.e., int compare(T t1, T t2), then the lambda expression looks like  
Comparator<T> c = (t1, t2) -> expressionReturnsInt;

TreeSet(Comparator<T>) takes Comparator as the parameter, hence we can write the TreeSet constructor as below:-  
Set<Employee> employeeSet = new TreeSet( (t1, t2) -> Double.compare(t1.salary, t2.salary))); // this sorts based on the salary in ascending order

Set<Employee> employeeSet2 = new TreeSet(); // this sorts using Comparable<Employee>

Case Study should have the following approach

1. MVC layer architecture - View, Service, Controller & DAO layer
2. Use factory design pattern to instantiate Service & DAO classes
3. Use interfaces in Service and DAO to create a loosely coupled application
4. Create one simple example that follows the layered architecture that can perform create, read, update and delete employee object in a List<Employee>
5. Understand how to use layered architecture in console based application with Core Java & implement the same for the above example.
6. Service layer will use the interface reference of DAO and get the DAO instance with the factory pattern, Controller will use the interface reference of Service and get the Service instance with the factory pattern.

Note: In console based application main() method acts like View & Controller both, however in real time you will have separate files to maintain View & Controller logic.

In the case study:

1. Scanner object & System.out.println() you will write only in main method, it should not be used in the Service/DAO
2. Package structure should be created as:

com.oracle.dao -> Dao interfaces & implementations

com.oracle.business -> Service interfaces & implementations

com.oracle.beans -> Java beans / models / Encapsulated classes

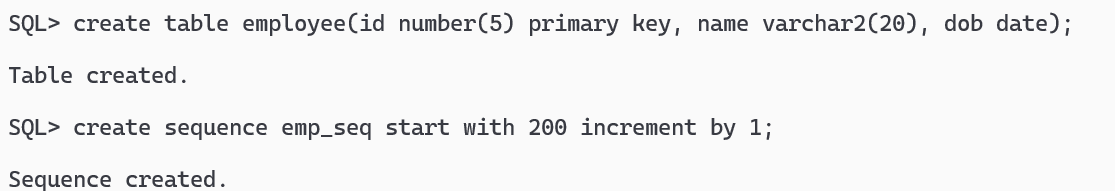
com.oracle.dao.utility -> Factory patterns & Utility classes for DAO layer  
com.oracle.business.utility -> Factory patterns

com.oracle.controller -> Controller classes

Simple example following layered architecture.

1. We will create a DAO that uses Collection first
2. Then we will create a DAO that uses JDBC
3. Performing CRUD operations on employee
4. Things to create
   1. ViewController -> main method
   2. Employee class
   3. EmployeeService & EmployeeServiceImpl
   4. EmployeeDao & EmployeeDaoTempImp
   5. EmployeeDaoFactory
   6. EmployeeServiceFactory

Create table & a sequence



JDBC is used to interact with any RDBMS, it provides set of APIs which helps you to perform DB operations, below are the API’s

1. DriverManager
2. Connection
3. PreparedStatement
4. ResultSet

Every Database gives us a jar file which are JDBC driver jar files, we need to add this jar file in our project to connect to the database.

ex:

1. Oracle : ojdbc.jar
2. MySQL: mysql-connector.jar

Steps to connect to DB

1. Loading the JDBC Driver

Class.forName(“oracle.jdbc.driver.OracleDriver”); // optional from Java 8 onwards

1. Establish connection

Connection con = DriverManager.getConnection(url, user, pass);

1. Create Statements - convert java to sql statements

PreparedStatement pstmt = con.prepareStatement(sqlQuery);

pstmt.setType(index, value); // do only if query as ?

1. Execute the statement

int count = pstmt.executeUpdate(); // for insert, update, delete queries

ResultSet rs = pstmt.executeQuery(); // for select queries

1. Close the resources

rs.close();  
pstmt.close();  
con.close();

Eclipse Link with JPA

JPA stands for Java Persistence API, it is a ORM framework that can directly map java object to the table, it is built on top of JDBC to avoid lot of repeating tasks, if you use ORM you don’t have to perform following tasks

1. Exception handling
2. Data conversion from SQL to Java and vice versa
3. Writing long queries to JOIN tables or insert, select, update, delete queries

JPA does this using Entity class, which is a Java bean with table informations.

Eclipse Link is a library that implements JPA feature

Entity class

@Table(name = “employee”) // optional if table name & class name are same  
@Entity  
class Employee {   
 @Id // to mark the primary key column i.e., id column in employee table  
 @Column(name = “id”) // to map the column, optional if column & property have same  
 private int empId;  
 @Column(name = “name”)  
 private String name;  
}

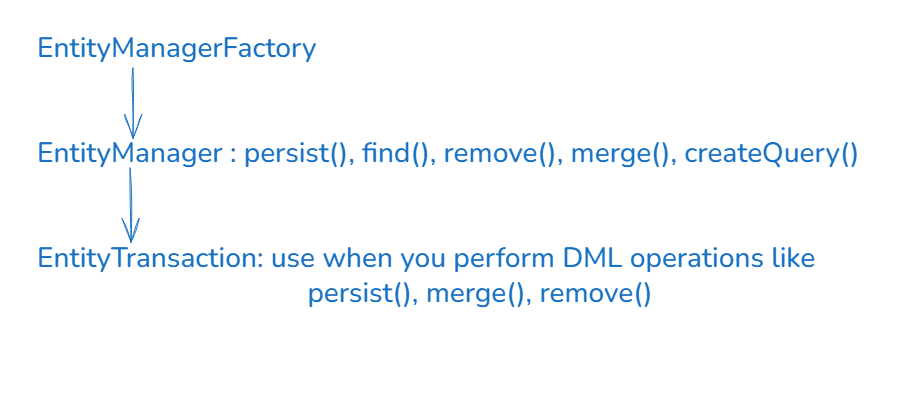
JPA

It gives you built-in methods to perform CRUD operations & also built-in annotations to JOIN the tables like @OneToOne, @OneToMany and so on

JPA takes care of creating the connection by itself using the persistence.xml file, these connections will be stored in a pool called as connection pool, we need to just reuse this connection and close once we have completed our operation, once you close the connection, the connection goes back to the pool so that it will be reusable.

EntityManagerFactory factory = Persistence.createEntityManagerFactory(“unit-name”);

The above code creates a connection pool factory



EntityManagerFactory factory = Persistence.createEntityManagerFactory(“unit-name”);  
EntityManager manager = factory.createEntityManager(); // get the connection to perform CRDU operations  
EntityTransaction tx = manager.getTransaction();  
tx.begin();  
manager.persist(emp); // stores employee to employee table  
manager.merge(emp); // updates employee table  
manager.remove(emp); // removes employee record from employee table  
tx.commit(); // save the changes

Employee e = manager.find(Employee.class, 1); // select \* from employee where id = 1;

TypedQuery<Employee> query = manager.createQuery("select e from Employee e", Employee.class);

List<Employee> list = query.getResultList();

In your case-study

1. Follow the layered-architecture
2. Learn about JPA association mapping first (like one-to-one, one-to-many, many-to-many etc.,) and then start your case-study
3. Try to complete by Tuesday 4PM

Weblogic Administrator roles

1. Installing
2. Configuring the domains
3. Starting the server
4. Deploying war/ear files
5. Creating Datasource using which any program can connect with the help of JNDI name
6. Creating JMS resources which you can use to send and receive messages

Weblogic console: This is a web page an admin use to login.

JMS stands for Java Messaging Services, which enables you to send and receive different types of messages.

It needs two things to make any program to connect to the JMS resources

1. Connection Factory -> connects to the JMS server which is part of the server
2. Destination Objects -> resources that maintain messages which are Queue, Topic and etc.

Webservices

It is a technology that helps to share the data in a common format like XML/JSON so that other technologies can easily convert to the structure they understand

ex: Phone Pay can share data to various Banking Services

ex: Uber or Ola and use the Google Map Service

There are two types of webservices

1. SOAP - Simple Object Access Protocol (XML to share the data & its quite older) - it works only with Enterprise applications, you can’t use this to integrate mobile or web apps
2. REST - Representational State Transfer - It works with all types of applications, you can integrate any types of applications, it uses data-structure’s like XML, JSON, TEXT, CSV, HTML)

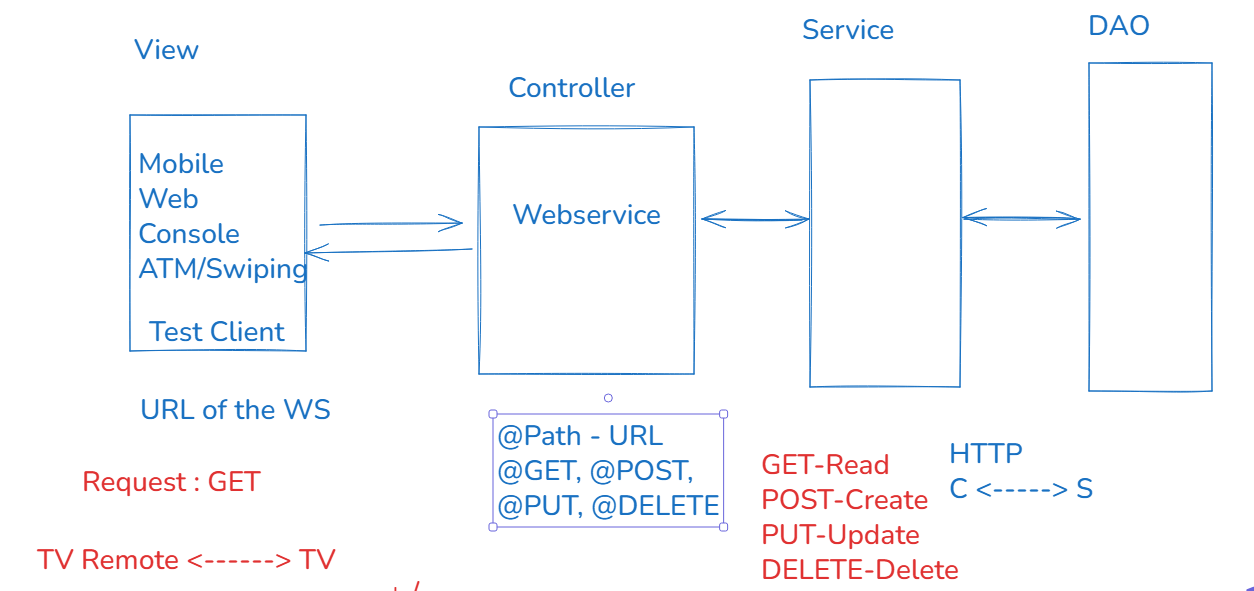
Java has JAX-RS API that helps you to develop RESTful webservices, JAX-RS stands for Java API for XML Rest Service, it has many implementations like Jersey library

API’s are not implemented, it must be implemented by some vendors like Jersey, Apache CFX.

Software’s required

1. JDK 17 or later
2. Eclipse IDE
3. Apache Tomcat 10 or later
4. VS Code with Thunder client plugin - Test the webservice

Where does the webservice come in the MVC layer



You must use Web Project with Maven configuration

Maven configuration: It is used to mainly mention the library names, which can download from the internet.

Configuring the project Webservice + JPA + MVC architecture

1. Create a Dynamic Project
2. Convert to JPA -> Download EclipseLink library
3. Add ojdbc8.jar
4. Add the libraries to the deployment assembly
5. Convert the project to Maven -> Add jersey library -> change web.xml
6. Make sure all the libraries are present in the deployment assembly

Topics to go through

* All the topics mentioned in the Content
* Make sure you to go through @PathParam, @RequestParam which you will be using in your case study.

Spring Framework

It is a light weight framework that helps you to create complex enterprise applications in a better way that you can easily test & maintain.

70% of the tasks spring framework does & the rest 30% developers has to take care, spring framework takes care of following tasks

1. Type conversion
2. Database connections
3. Exception Handling
4. Design pattern
5. Transaction Management
6. Object creation & Dependency Injection

At the core Spring framework has IOC (Inversion of Control) which takes care of Dependency Injection

IOC: It is a container that manages the objects

DI: It is a process where an object is supplied to another object

Spring Framework introduced a module called Spring Boot which simplifies the spring project configuration

Without Spring Boot you may need to still configure

1. Servers
2. XML configurations -> To configure the beans

With Spring Boot you don’t need to configure anything

1. You don’t need server, because spring boot project will have default servers
2. You don’t need XML configurations, spring boot uses a simple text file i.e., properties file to maintain the application configuration

Without Spring -> You must do all the tasks -> design patterns, configuring servers, object creations

With Spring -> You will get the platform to perform all the repeating tasks -> design patterns, object creations, exception handling -> you only need to configure server, front controller & some other things in the XML file

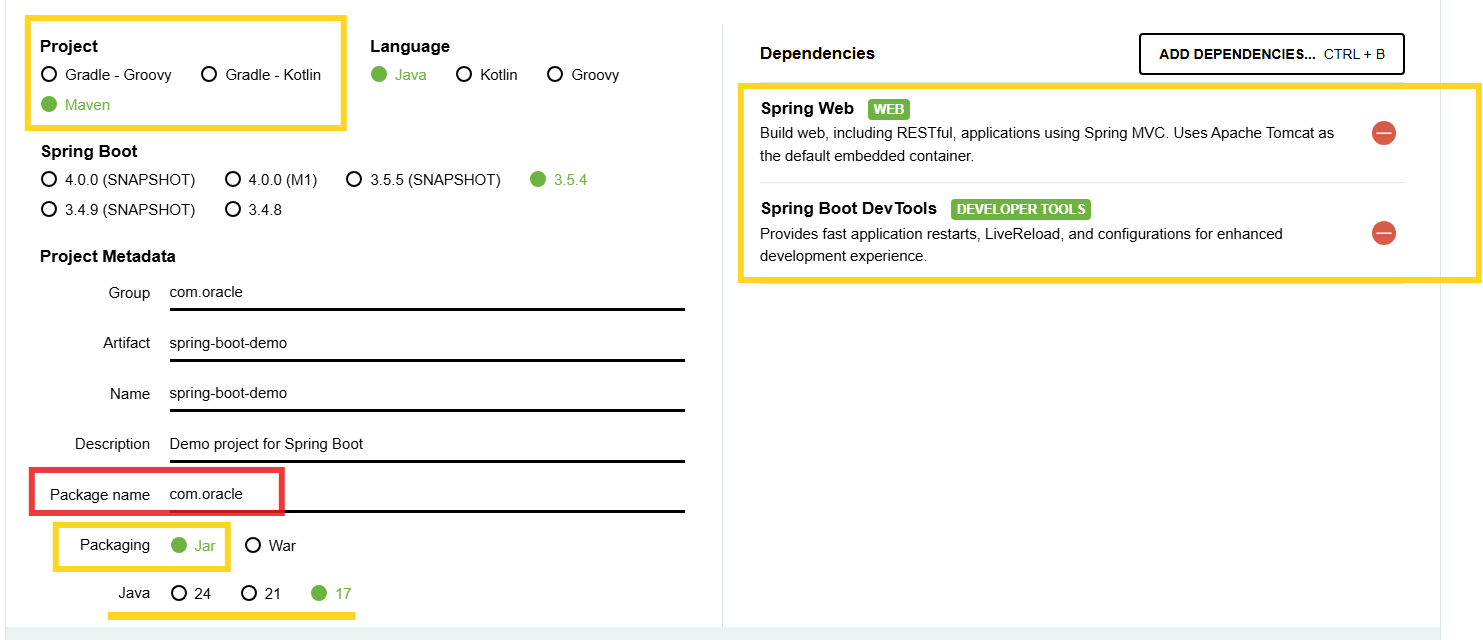
With Spring Boot -> You will get the ready to run project -> you must write only business logics

JaxRS vs Spring Boot annotations for Restful webservice

|  |  |  |
| --- | --- | --- |
| Description | JAX-RS | Spring / Spring Boot |
| Specifying a class as a controller | @Path | @RestController |
| Specifying the main path to the controller | @Path | @RequestMapping |
| Specifying HTTP methods | @GET, @POST, @PUT, @DELETE | @GetMapping, @PostMapping, @PutMapping, @DeleteMapping |
| Response | Response | ResponseEntity<T> |
| Path Parameters | @PathParam | @PathVariable |
| Dependency Injection | No Annotation | @Autowired |
| Server | Need to Configure | Not required - Default servers |
| DB Connection | You need to write code | Not required |

Note: When you have an old project using Core Java or Java EE or JAX-RS you need to migrate the project to spring by making lot of changes in the class

Spring community has a website called spring initializr that gives you ready run maven / gradle projects, you can also get all the maven dependencies in that website



Without Spring

interface UserService { }   
class UserServiceImpl implements UserService { }   
class ServiceFactory { create object of UserService implementations }  
client code -> UserService service = ServiceFactory.getServiceInstance();

With Spring

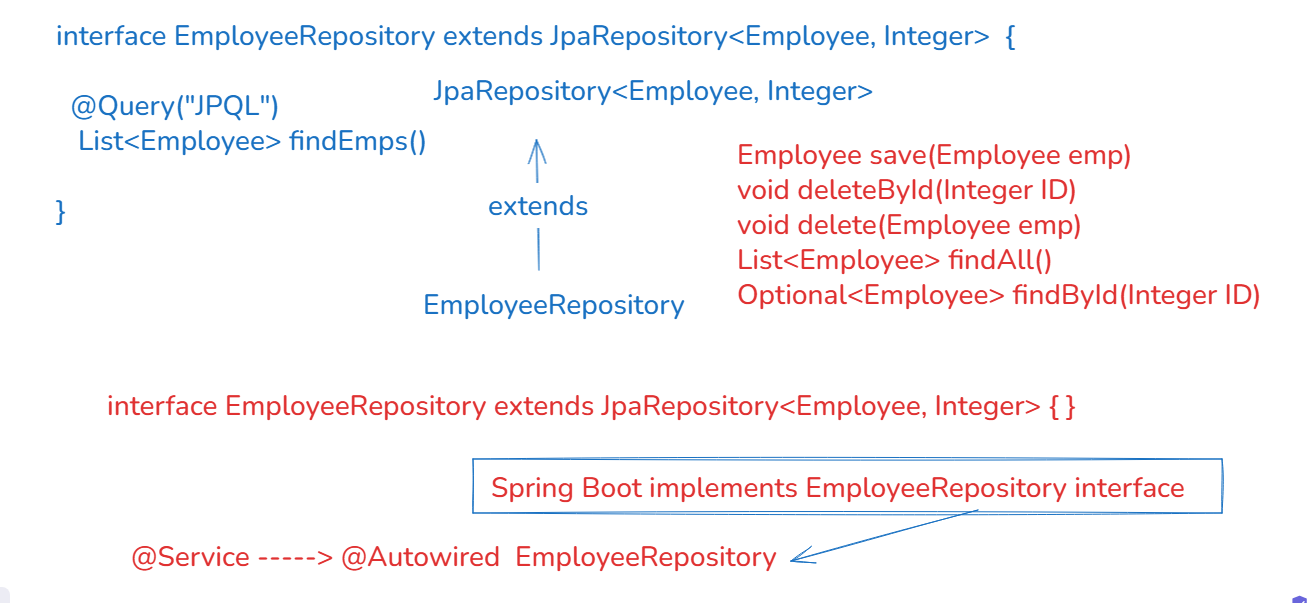
interface UserService { }   
@Service  
class UserServiceImpl implements UserService { }   
Client code -> @Autowired UserService service;

Spring Data JPA

This library auto-implements the DAO layer, you just need to extend the interface the library provides i.e., JpaRepository<T, ID>, this interface has all the CRUD methods to perform DB operations

JpaRepository<T, ID>: It is an interface having methods to perform CRUD operations on the database, T stands for entity class, ID stands for primary key type, it has methods like

1. T save(T)
2. void delete(T)
3. void deleteById(ID)
4. List<T> findAll()
5. Optional<T> findById(ID)
6. long count()



If you use Spring Boot + Data JPA then no need to create DAO layer, because spring boot auto-implements DAO layer & its object is available to use, you can still write custom methods inside the interface which must use @Query() annotation that takes JPQL.

Libraries required

1. Spring Web
2. Spring Data JPA
3. ojdbc8.jar
4. Dev tools

application.properties: It will have the datasource information, spring boot establishes the connection based on the datasource details

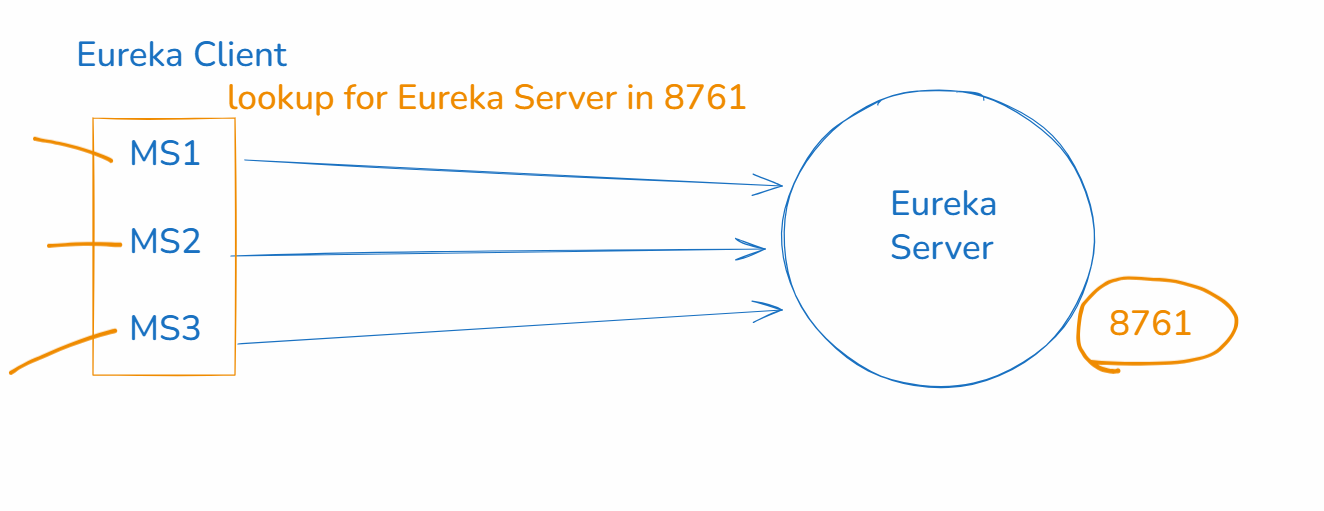
Microservices

These are small services that you can develop, test and deploy independently.

Steps to follow while creating microservices

1. Discovery Server: You need to create a program that registers all the micro-services which are called as Discovery clients, in spring boot you have a library Eureka Server that helps your program to act like Discovery Server
2. Discovery Client: You need to create a program that acts like microservice which can call other microservices by using an unique-id of the microservice, in spring boot you have a library Eureka Client that automatically registers the microservice in the service discovery.

Note: Eureka Client expects Service discovery to run in 8761, that is where Microservice searches the service discovery to register its instance.



@EnableEurekaServer: This annotation configures the application to behave like a service discovery

What is Kafka

What is Kafka Topic

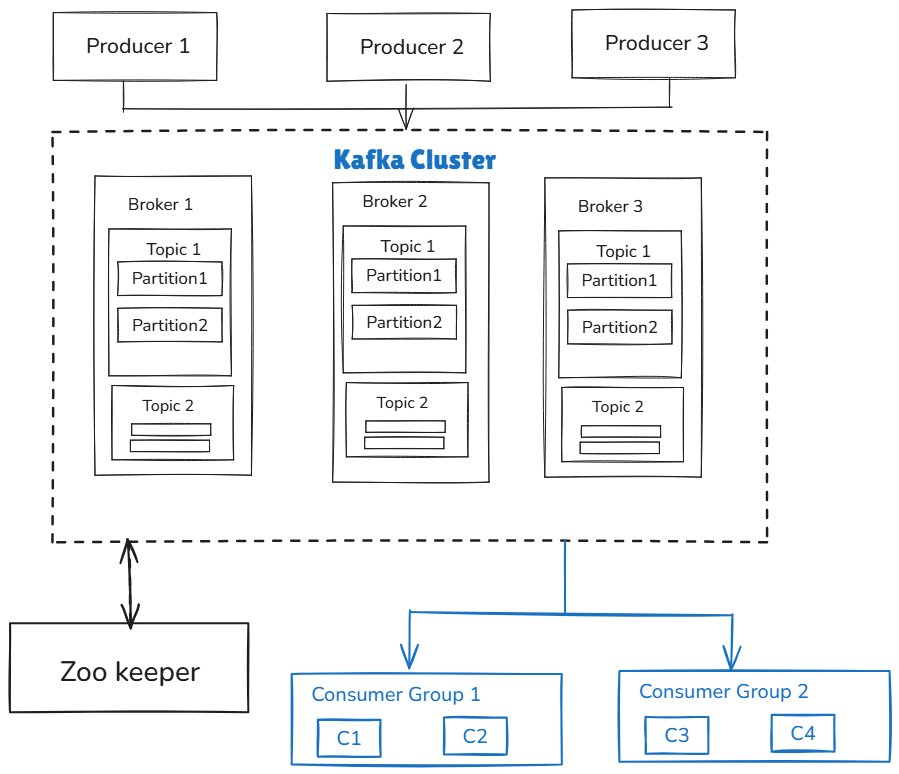
Why we use Kafka

What are Consumers & Producers in Kafka

Apache Kafka

Kafka is an event streaming platform that can transport high amount of real time data between the systems.

Kafka Architecture



Kafka Producers: These are the client programs which uses Kafka Producer API’s to send messages/events to the Kafka topics

Events: These are the data which will have message or action that needs to be performed

Kafka Brokers: It is a computations node or server in the Kafka cluster, it contains topic partitions that store messages in an immutable sequence. When producers produce the message they are stored across multiple partitions, if you have a replicated broker then the same message will be stored in the particular partition of the topic

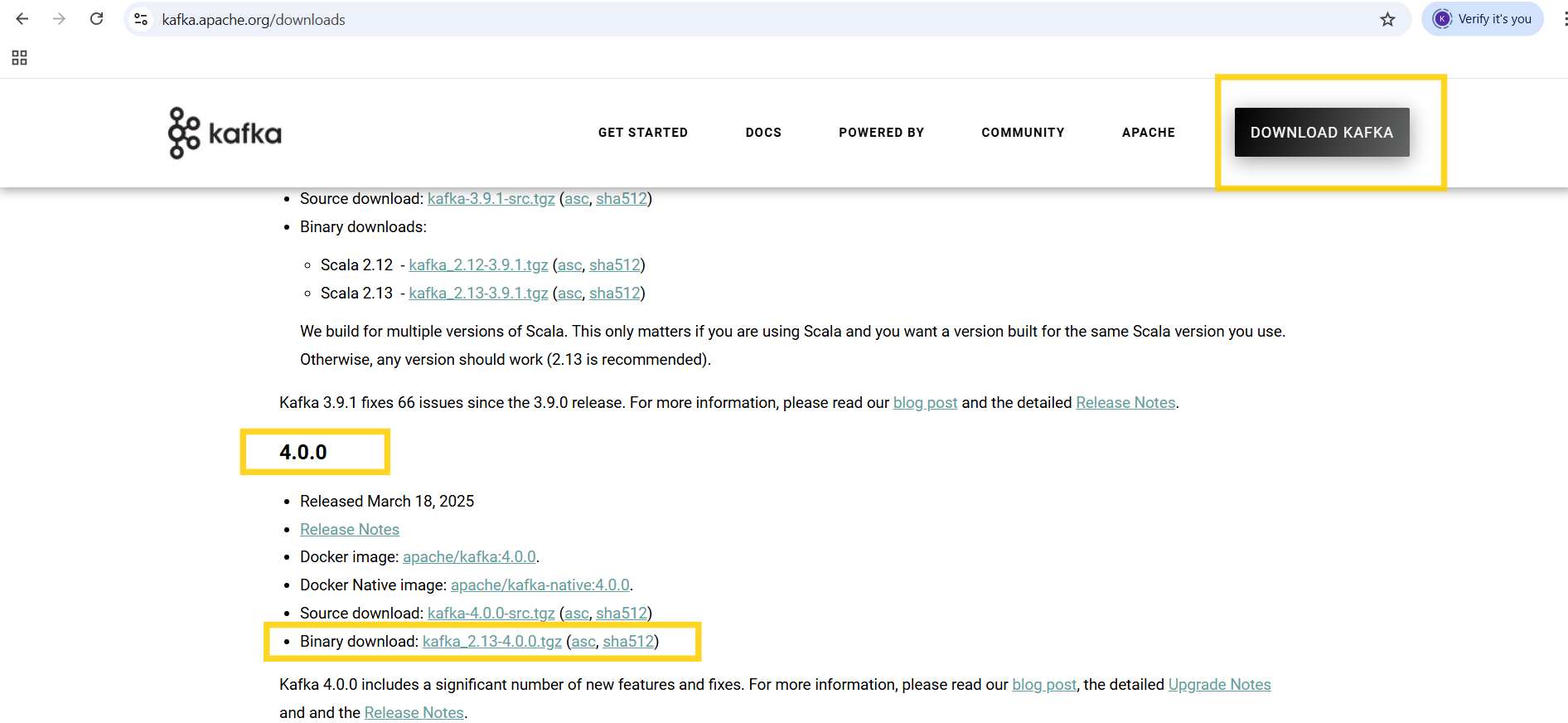
Zoo Keeper: It is a distributed coordinator that helps broker to co-ordinate with each other, it takes care of broker availability, status and determine the main broker

Note: zookeeper is no longer required from kafka 3.5 onwards

Quick Start

Let's set up Kafka to start its server, then we will produce and consume messages from topics.

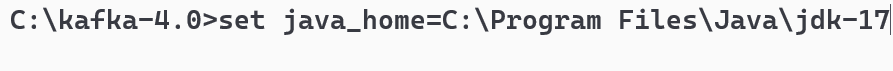
Download Kafka from the official site



Extract the zip and keep it in the C drive, then change the folder name to kafka-4.0

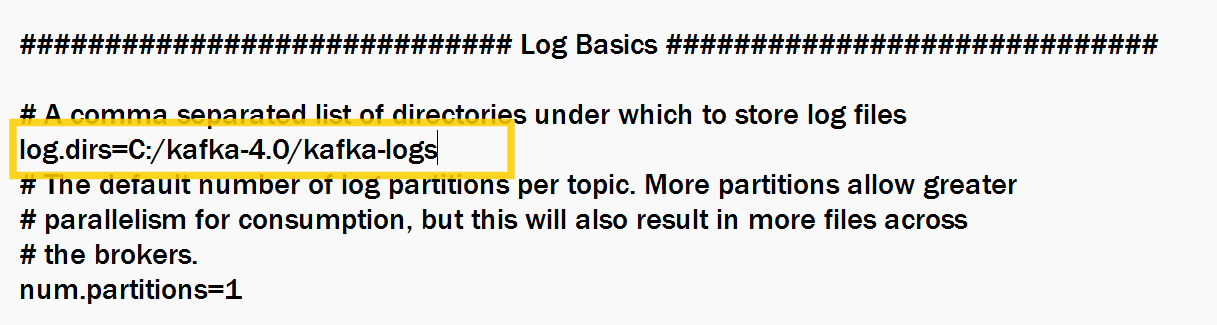


Set the JAVA\_HOME to point to JDK 17

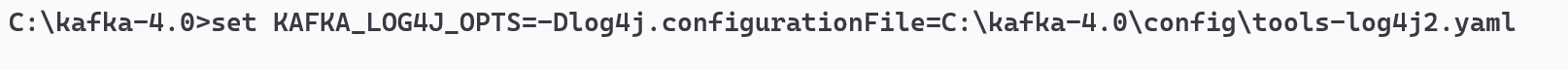


Note: Ensure from now on every terminal you use is a clone of this JAVA\_HOME

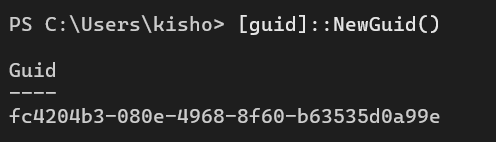
Open server.properties file present in kafka/config and change the log location



In Kafka 4 we get errors like reconfiguration failed when we try to start the server, hence we need to set location of LOG4J\_OPTS



Open powershell to generate a UUID

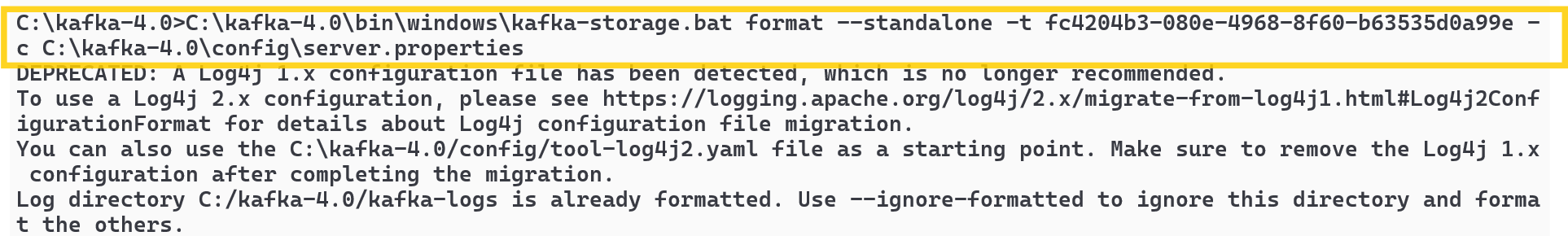


Note: We will use this ID to start the server

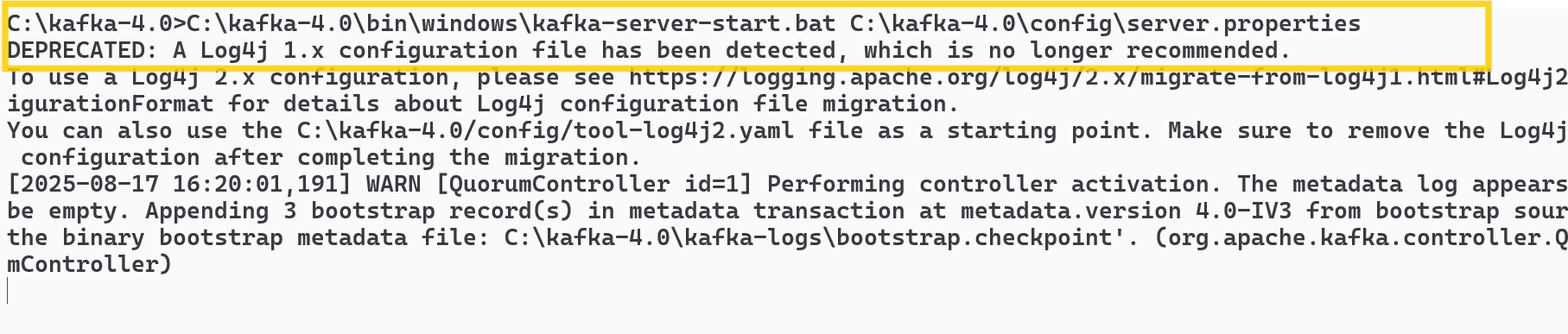
Format Log Directories

Syntax:

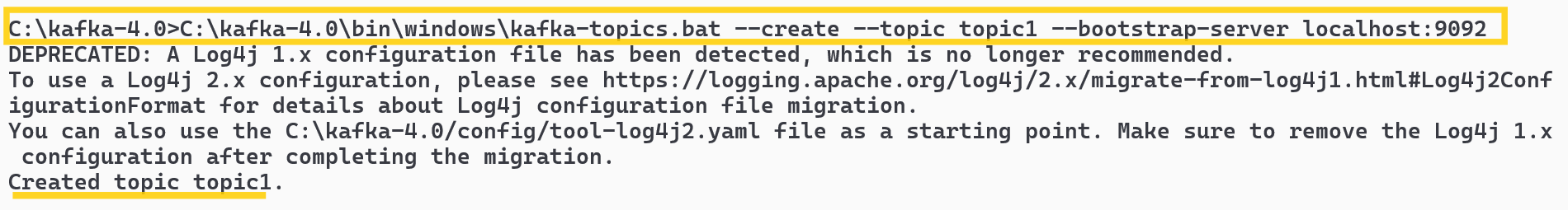
bin/kafka-storage.sh format --standalone -t $KAFKA\_CLUSTER\_ID -c config/server.properties



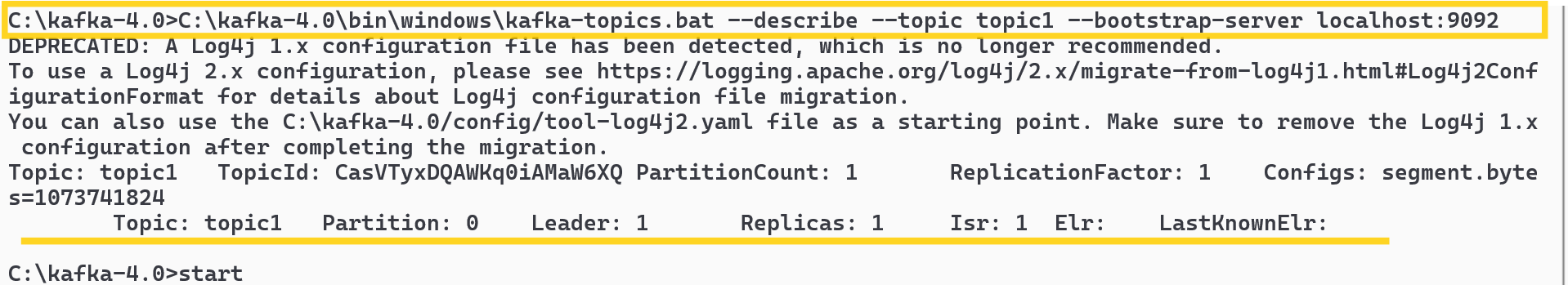
Start the Kafka Server



Create a topic to store your events

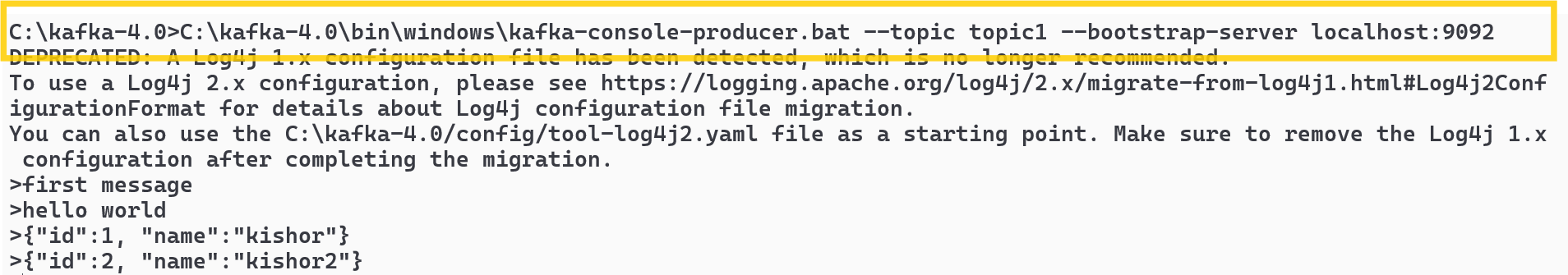


You see the details of the new topic using kafka-topics.bat

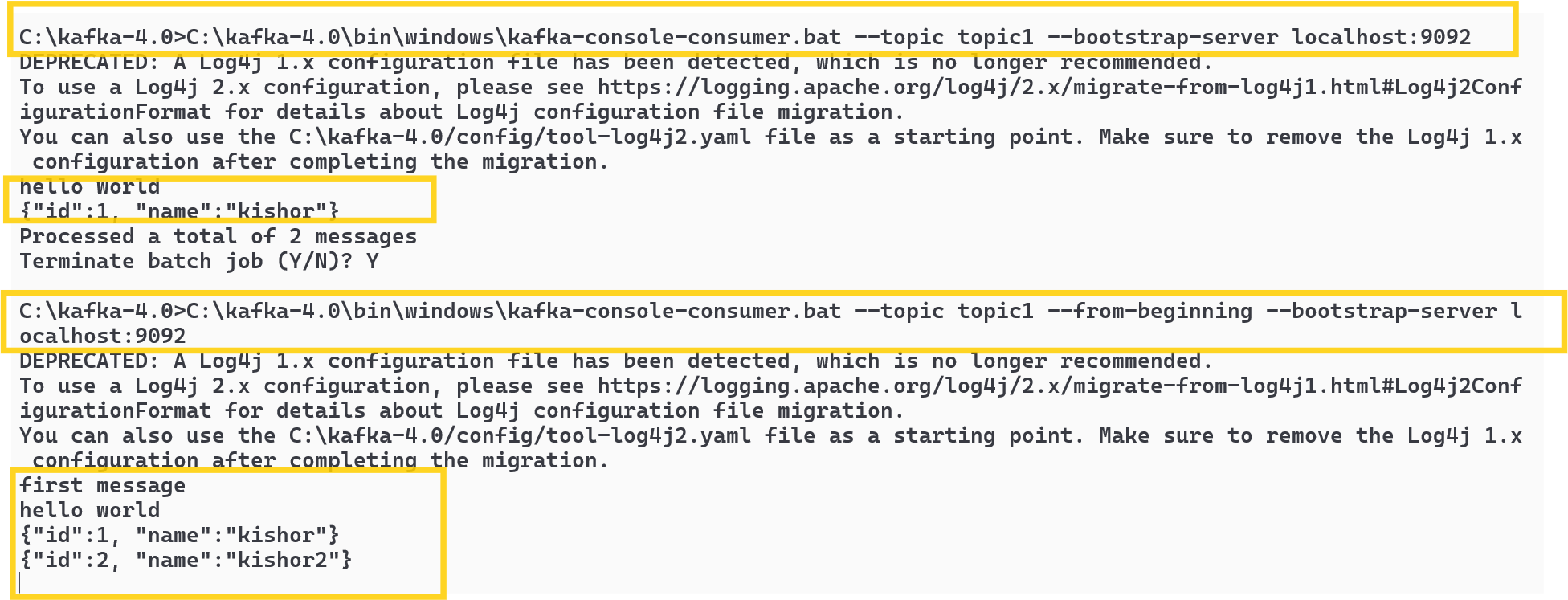


Write & Read events using producer & consumer

Producer program



Consumer program



Summary of kafka setup

1. Download & Install Kafka -> Copy to the C drive and rename the folder to kafka-4.0
2. Set the path of JAVA\_HOME to JDK 17, because Kafka 4.0 is compatible with JDK 17
3. Set the kafka-logs path in server.properties which stores the logs
4. Optional set KAFKA LOG4J OPTS to refer the tools-log4j.yml because internally the logs use this yml file
5. Kafka storage using an unique UUID (generate the UUID using powershell)
6. Start the kafka server
7. Create topic
8. Produce & Consume messages.

How to access the backend-services implemented using Jax-RS/Spring Boot from the front-end applications using Javascript or any other technologies.

Backend services blocks the requests coming from different technologies, the error you get is CORS error, CORS stands for Cross Origin Resource Sharing.

fetch(URL, {..}): It is a Javascript API used to send HTTP requests, by default it makes HTTP GET request, however you can change that using the 2nd parameter which is a Javascript object with properties like method, headers, body

HTTP GET with fetch

fetch(URL): no need to use 2nd parameter

HTTP POST with fetch

fetch(URL, { method : “POST”, headers : {“prop” : “value”}, body : jsonContent })

Javascript Recap

1. Modern Javascript Features
2. Object & Array Destructuring
3. Callback & Arrow functions
4. Promises
5. Async/Await

Typescript: It is a super set of javascript, it makes javascript type-safe, so that the result will be more predictable

Javascript

let x = 50;  
x = “hello world”;

x = {id: 200, name : “Ajay”, age : 35 }

Typescript

let x : number = 50;

x = “test”; // tsc gives error

require.js

RequireJS is a JavaScript file and module loader. It is optimized for in-browser use, but it can be used in other JavaScript environments, like Rhino and [Node](https://requirejs.org/docs/node.html). Using a modular script loader like RequireJS will improve the speed and quality of your code.

Knockout.js

It is a Javascript library used to develop rich UI, it follows MVVM architecture (Model-View-ViewModel) to separate Data, UI and Logic that interacts with Data & UI.

Model: It represent the data

View: It represents the UI

ViewModel: It is a function that interacts with the View & Model to update the changes to the View or to the Model.

OJET stands for Oracle Javascript Extension Toolkit, it is built on top of Knockout.js to create rich UI, it follows same MVVM model

Software requirements

1. node.js
2. ojet-cli: It can create & manage ojet project

Installing ojet-cli

npm install -g @oracle/ojet-cli@16.0.0

How to verify the ojet installation

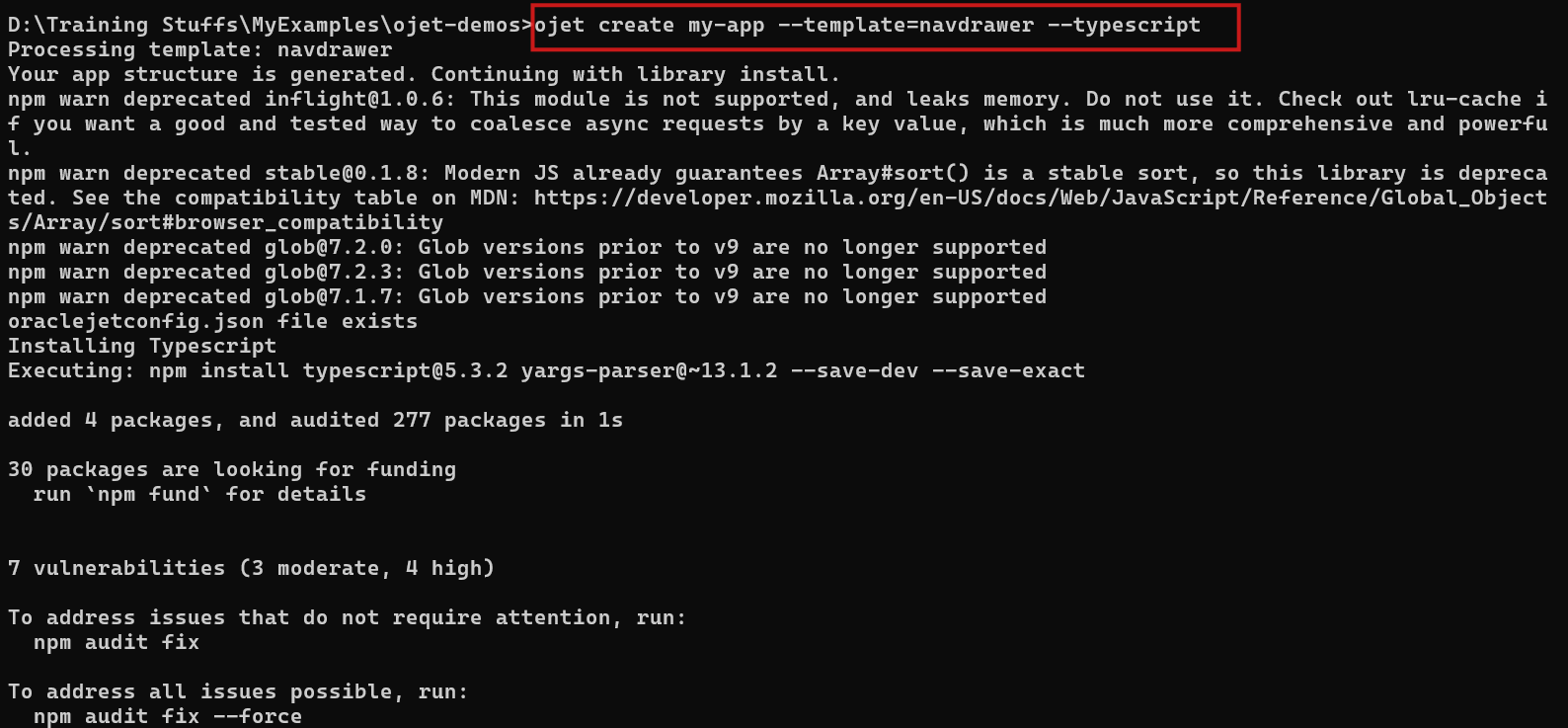
ojet -v

Ojet gives you many oracle website templates like blank, navdrawer, these templates will have built in oracle website templates

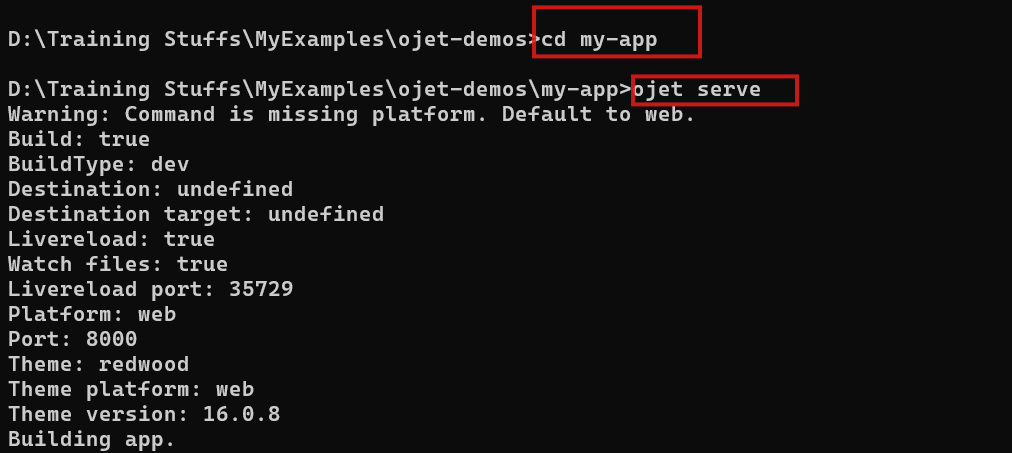
How to create ojet project

ojet create app-name --template=navdrawer --typescript

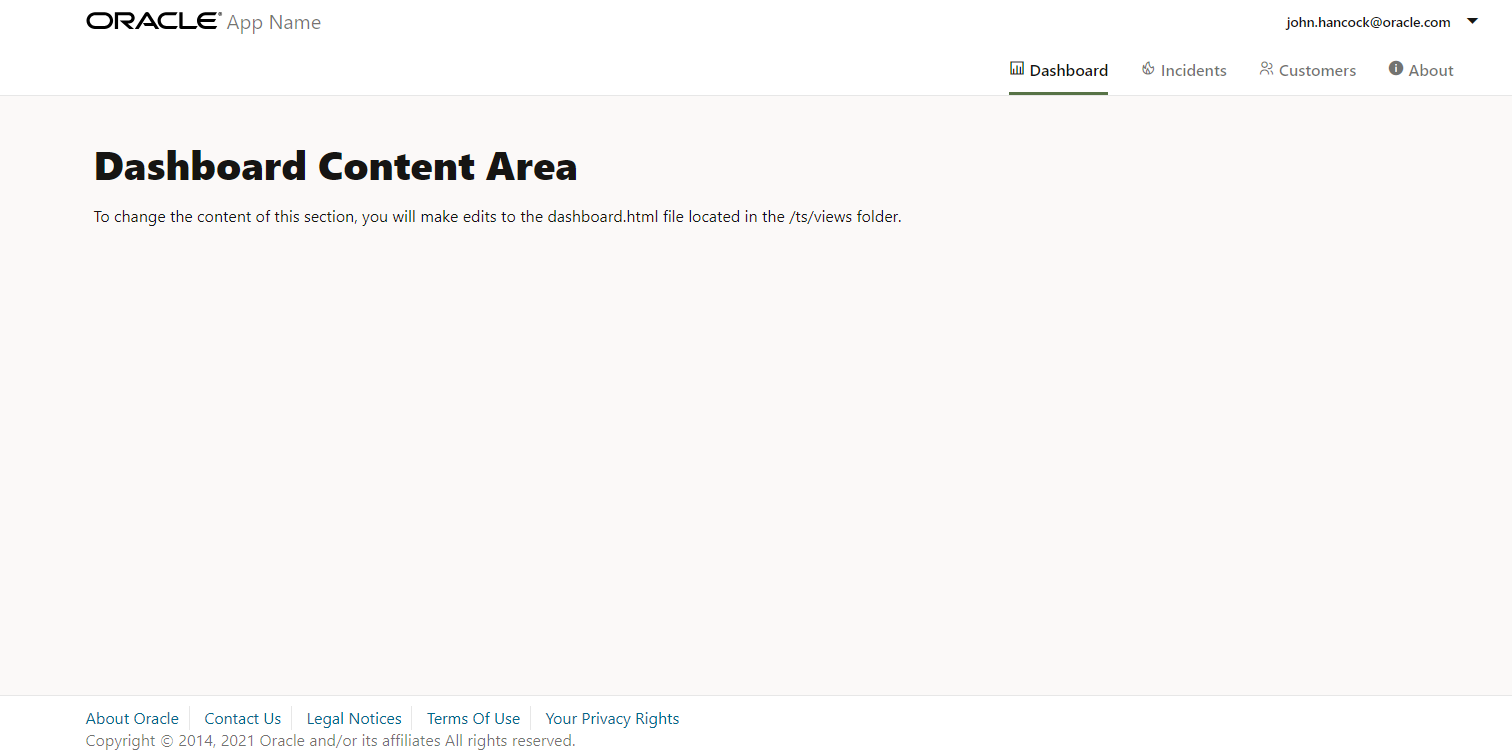
Creating the project with the starter template



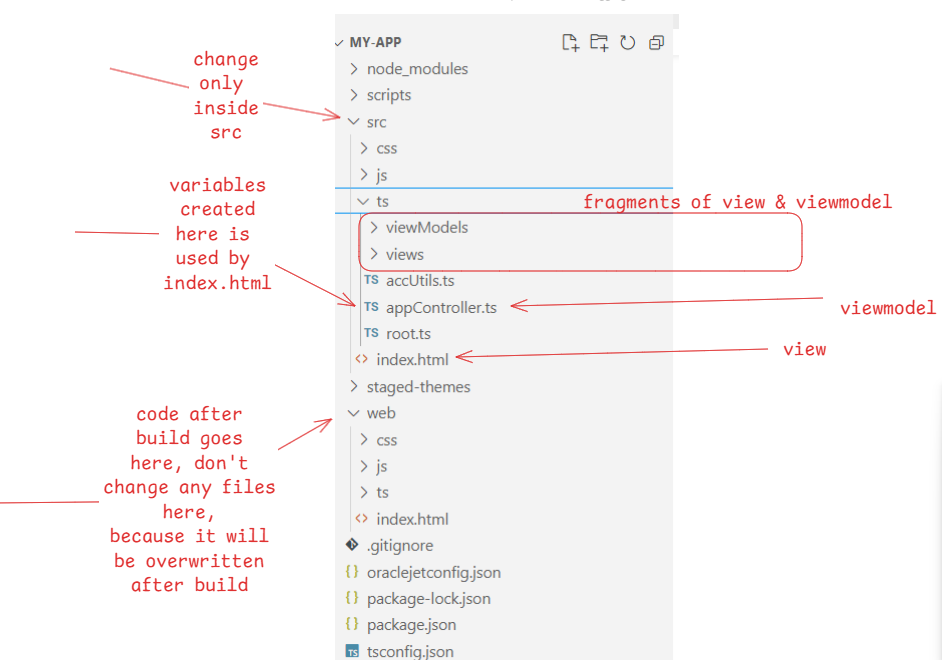
Navigate to the project & run using ojet serve



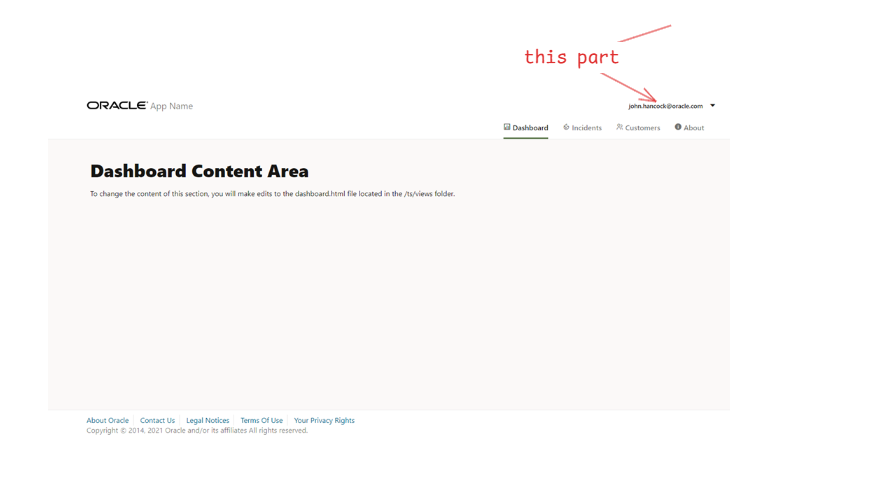
Output: will have a navigate template



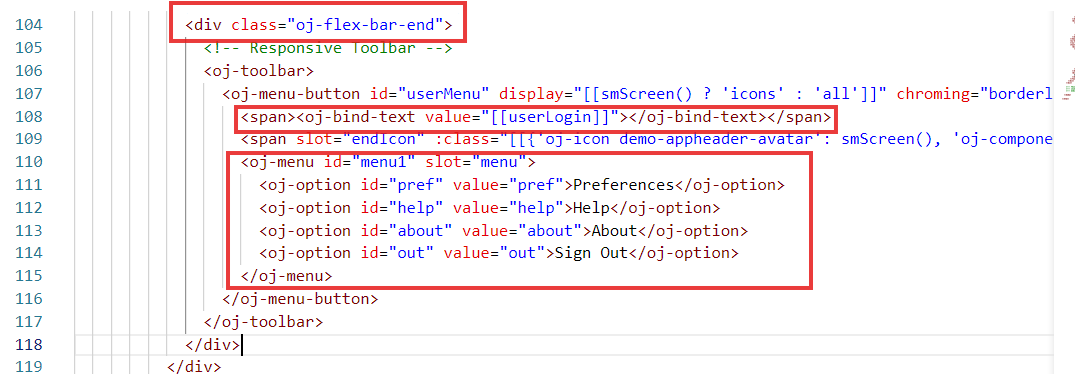
open the project folder in the vs-code



Removing the header content having the drop-down



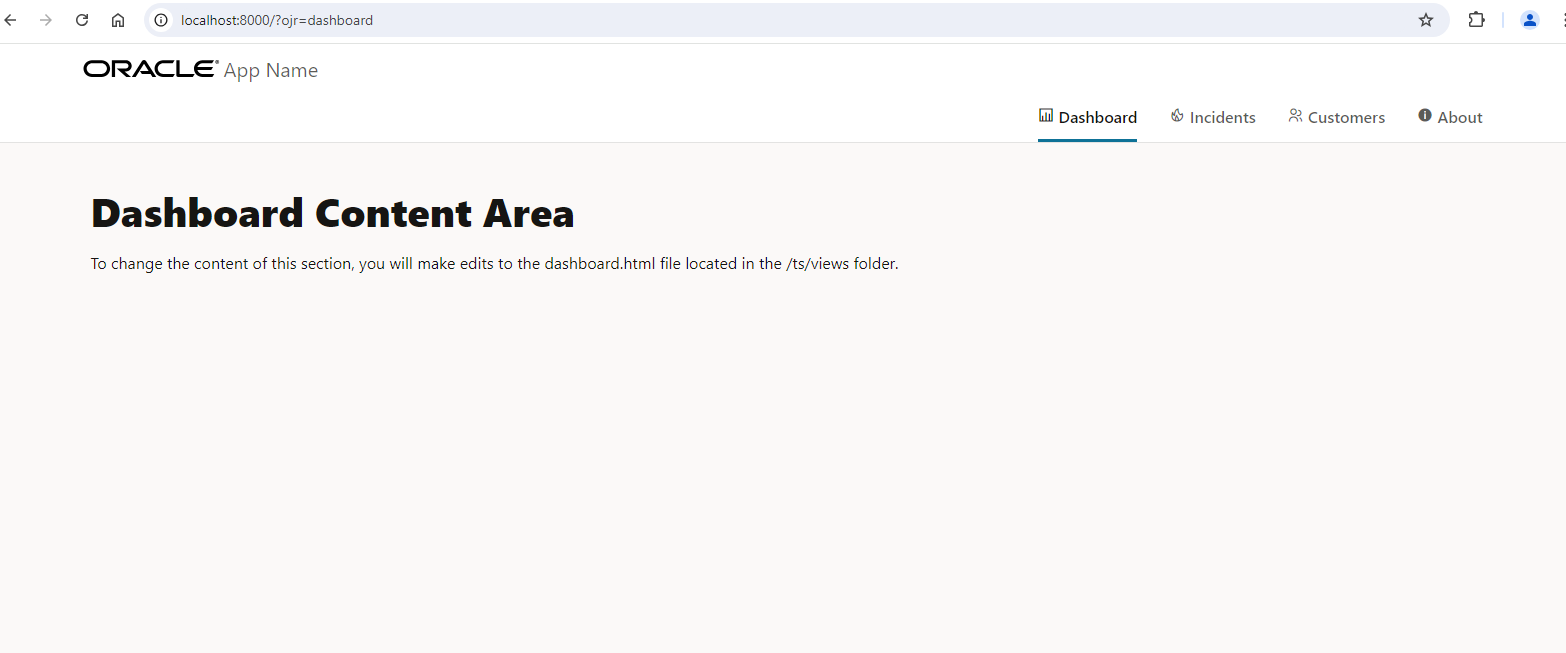
You can change the App to Banking App and also userLogin can be removed in the index.html



Now it looks like this



You can see the output

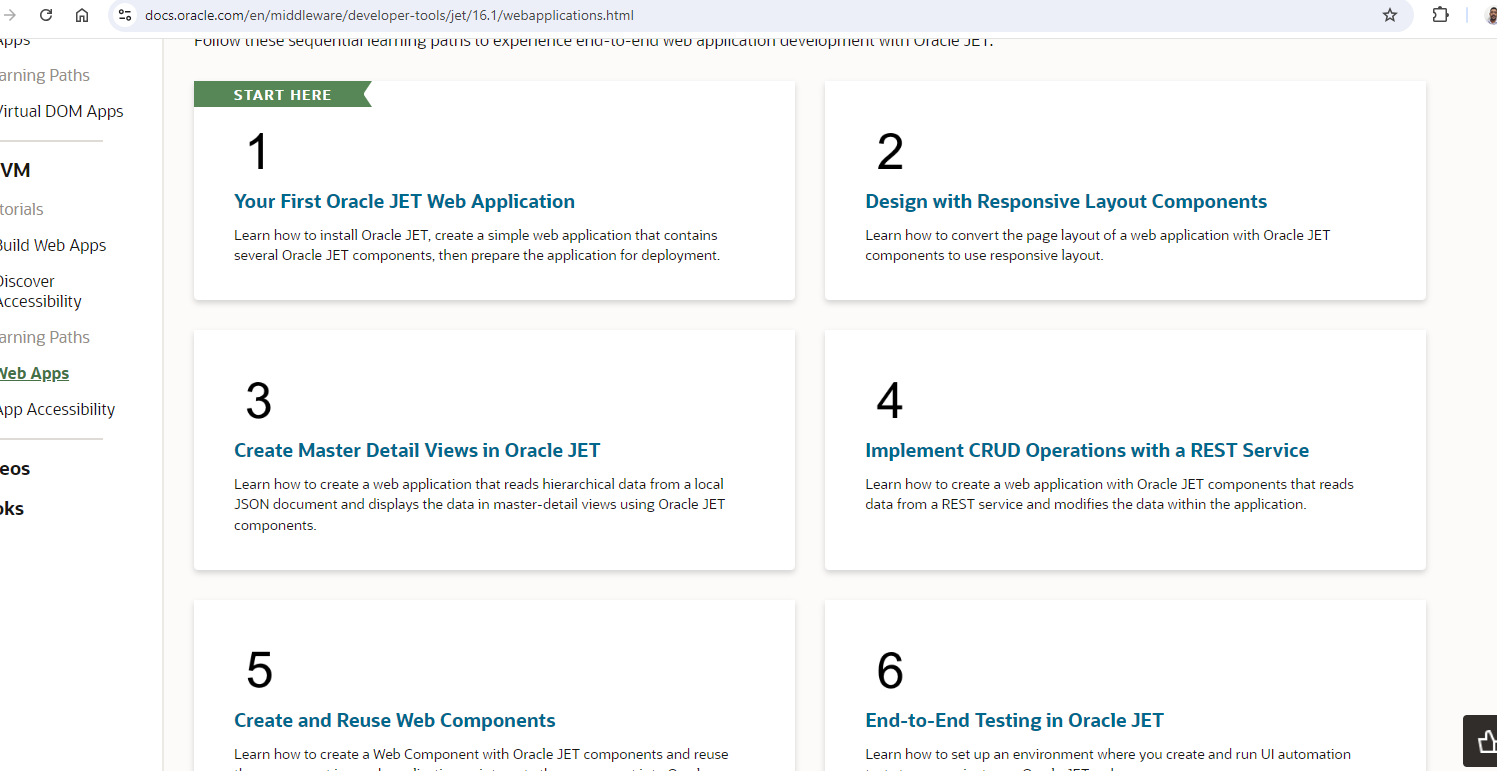


OJET basically offers rich UI components developed with knockout.js its documentation has cookbook, developer guide to learn about how to create components, how to use routers, how to make REST calls and so on.

Ex:

Use the below URL to see the step by step paths to develop an end to end OJET application from creating the first project to enhance it to perform REST calls

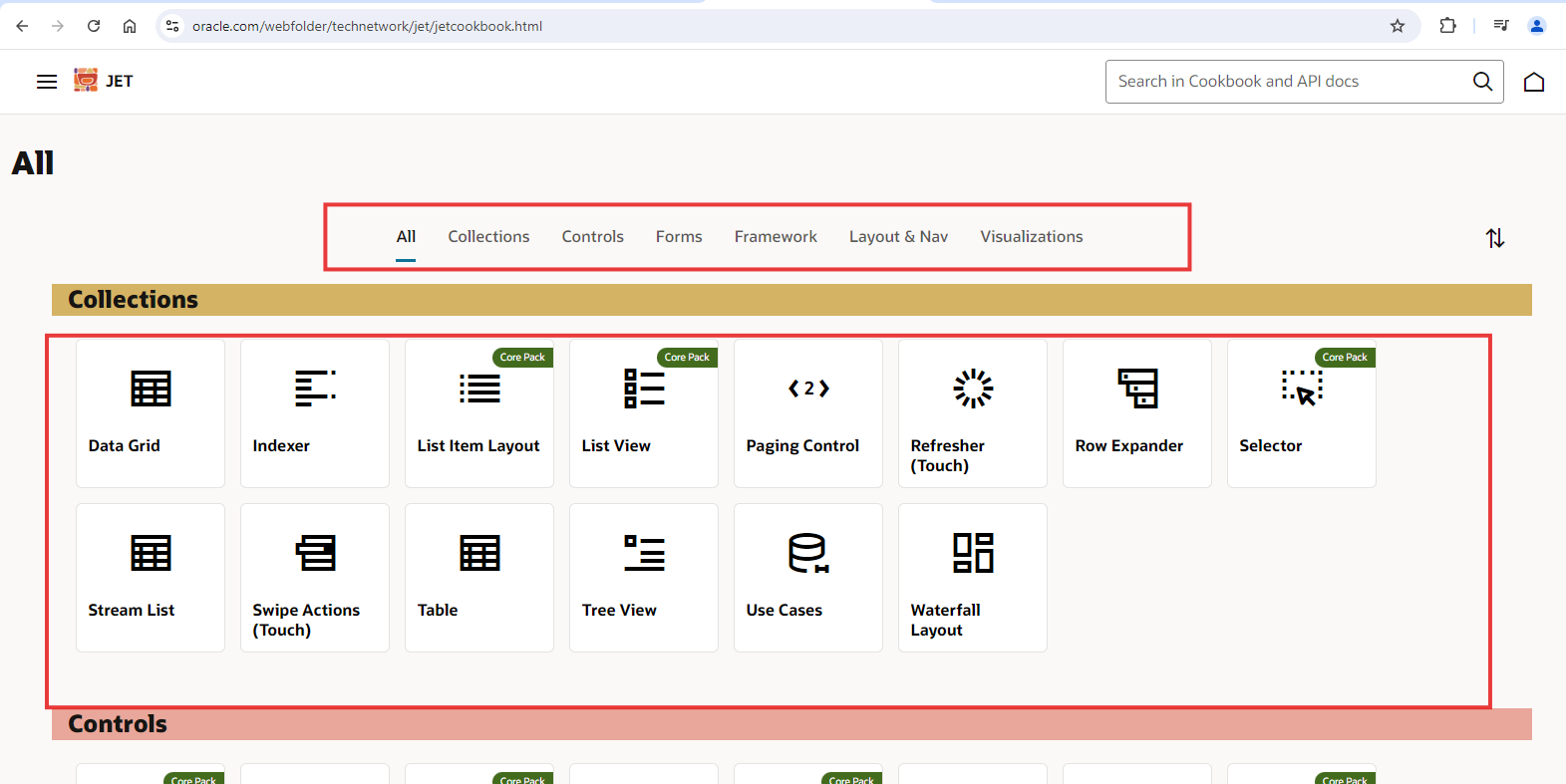
<https://docs.oracle.com/en/middleware/developer-tools/jet/16.1/webapplications.html>



Notice the 4th lesson which teaches you how to access backend with a REST service, which is what most of the UI technologies do, however for the beginning we will use the basic components with the help of cookbook

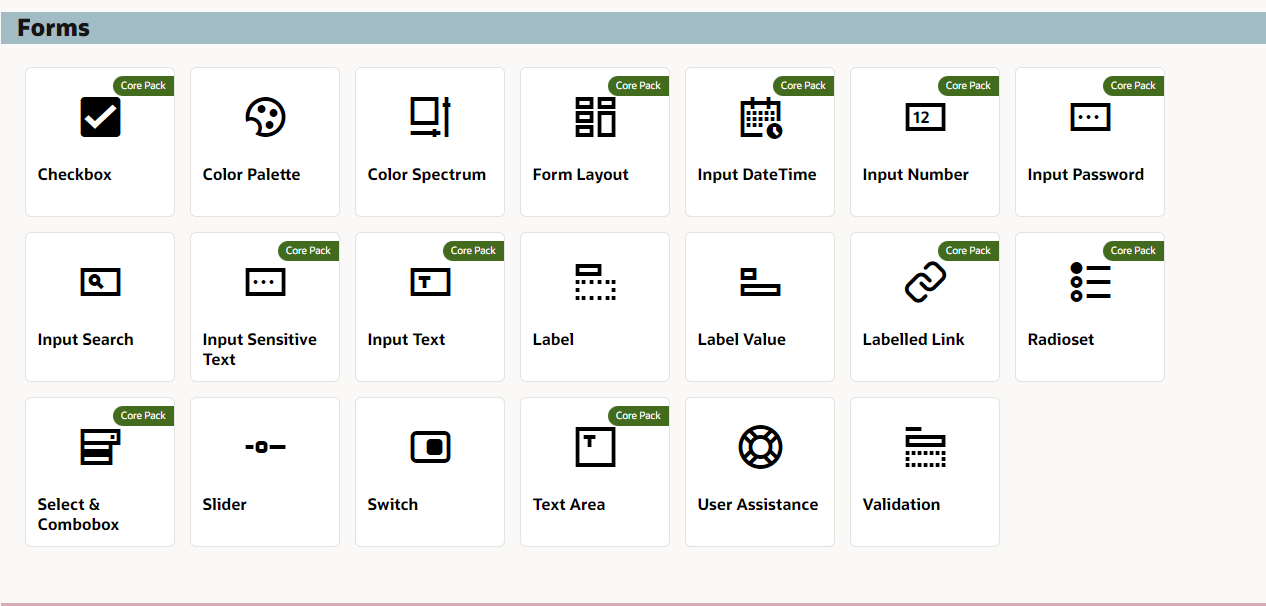
Below is the URL, you can type OJET cookbook in google to get this link

<https://www.oracle.com/webfolder/technetwork/jet/jetcookbook.html>

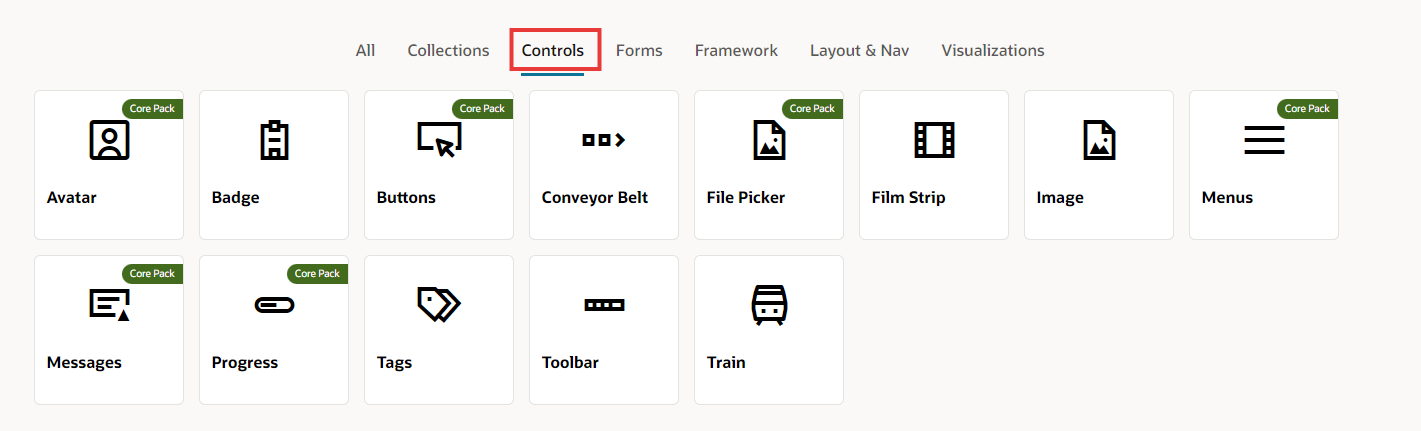
****

You have different types of components like Forms, Controls, Collections

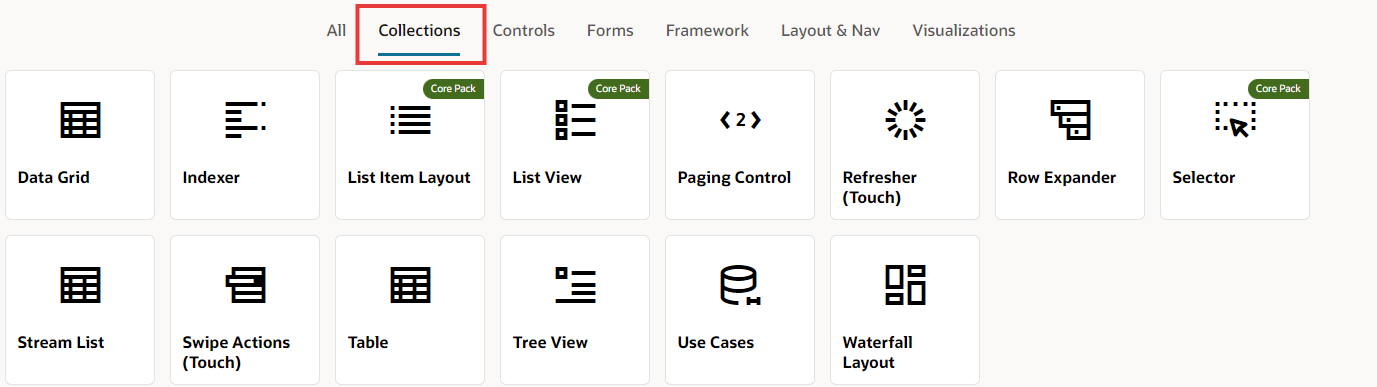
Forms: It has Input Text, Input Number, Input Password, Checkbox and so on.



Controls: It has buttons, Menus, Toolbar, Progress



Collections



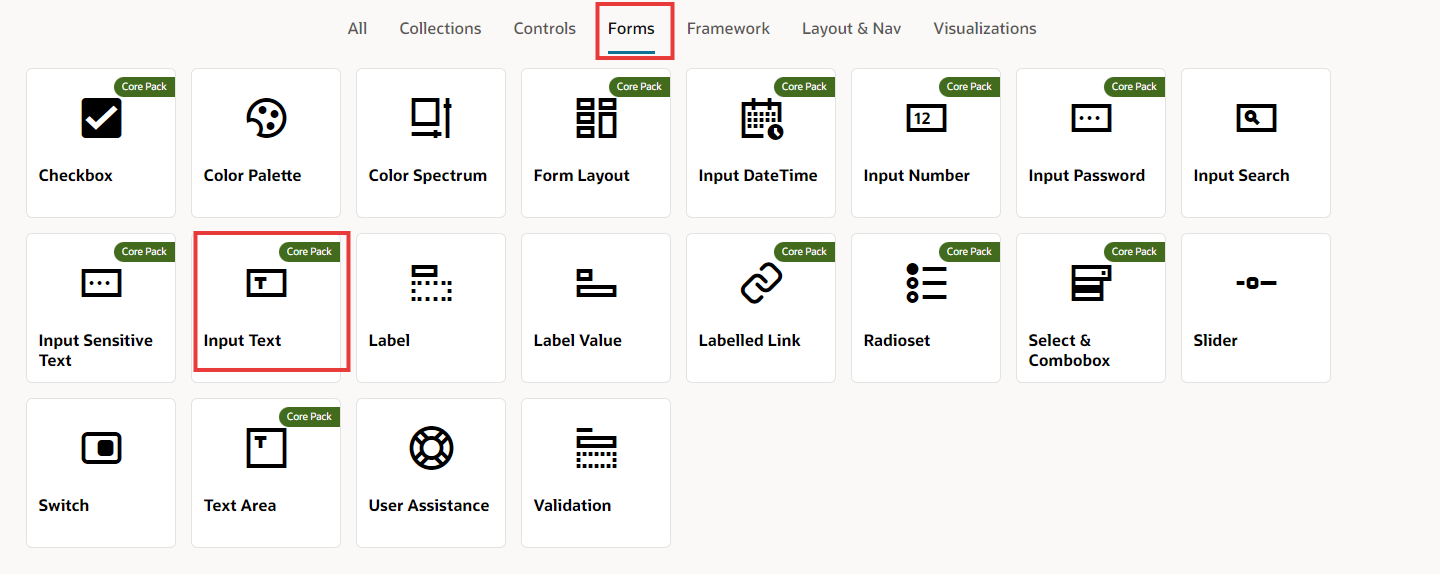
Lets begin with adding simple text box

Whenever any components you want to add in the application, you must follow below steps

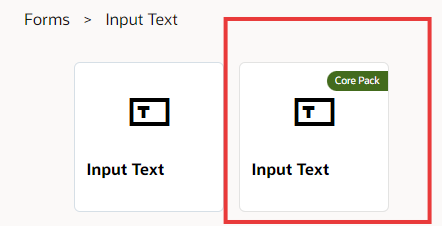
1. import the component in the TS/JS file
2. initialize the variables using knockout observables which you want to use in the HTML
3. Use the component in the html
4. If you are creating fragments of your own, you must create an html & TS/JS file with same names, in the viewModels & views folder
5. The new views wouldn’t be displayed unless you mention in the navData of appController.js file

Adding the input text

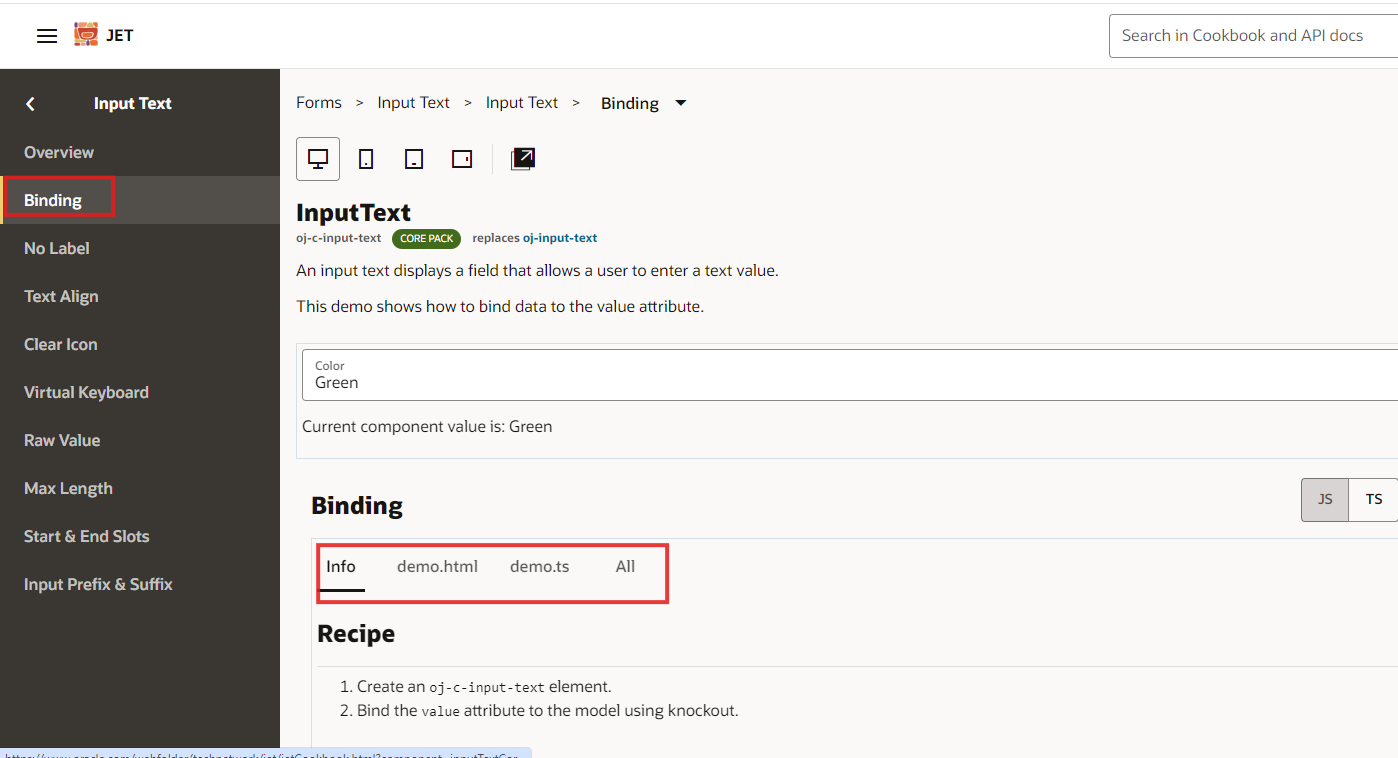
Go to the cookbook and search for input text in the Forms section



Click on the input text



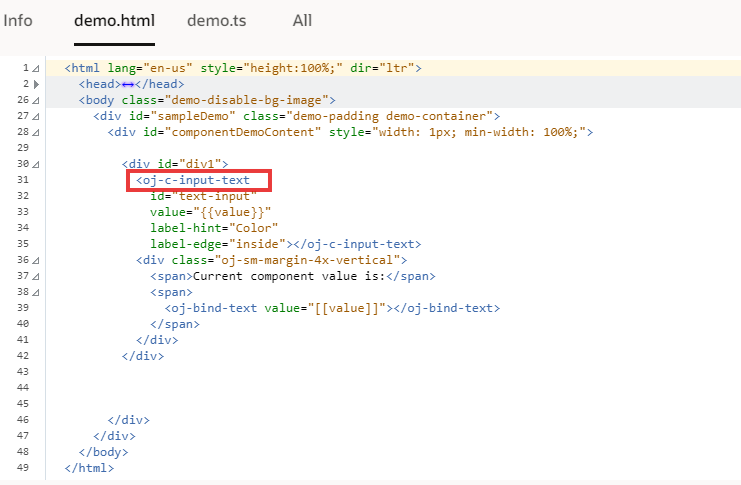
Select the Input Text having Core Pack, you can select the other one, but its in maintenance, since we are using OJET 16 and installed the project using OJET 16 we can use Input Text of Core pack



You will see description about the Input Text, what is the tag and what library you must import, there’s a section at the left side which might hide sometimes, just click on the 3 small horizontal lines beside the JET



Click on the demo.html and locate the input-text tag and paste in dashboard.html, but this tag must be imported in the dashboard.ts

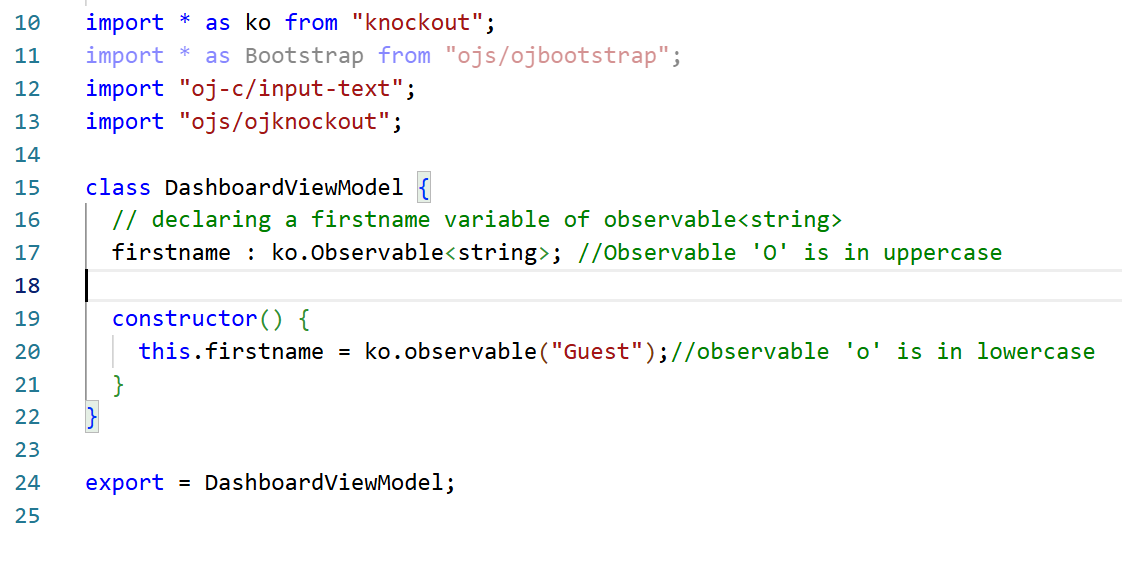


The code in demo.ts looks like below

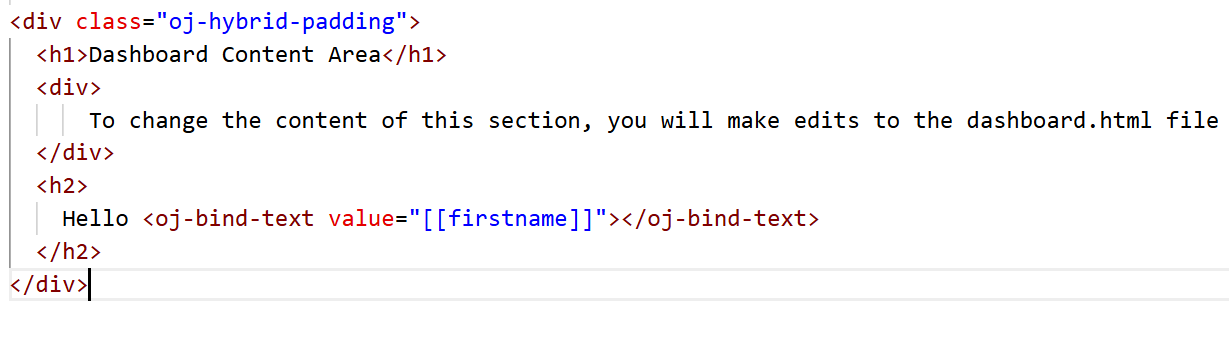


Let us copy these code and paste in dashboard.ts

dashboard.ts

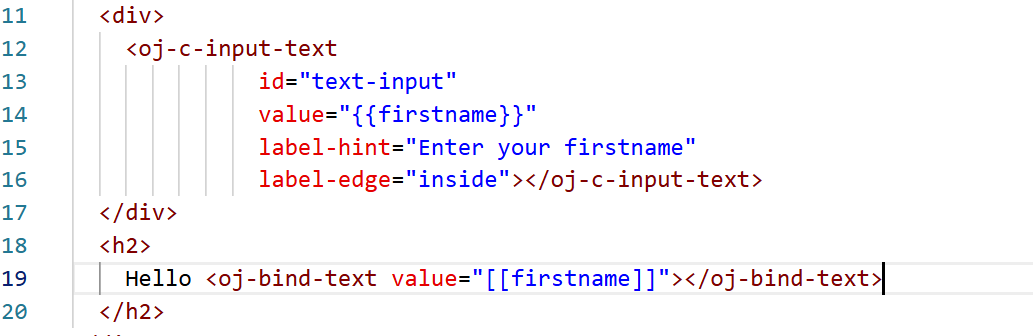


dashboard.html

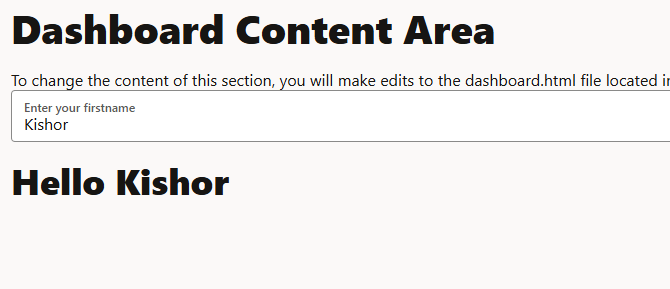


oj-c-input-text

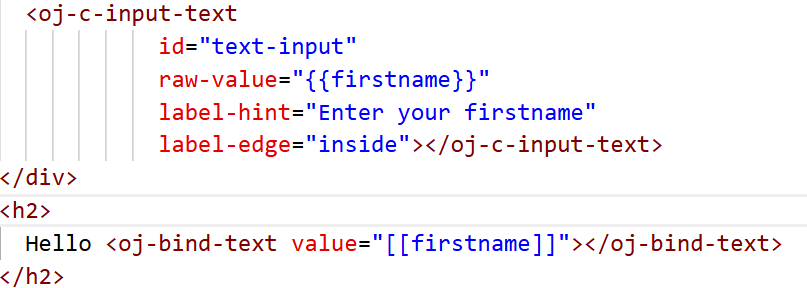
It is used to accept the input



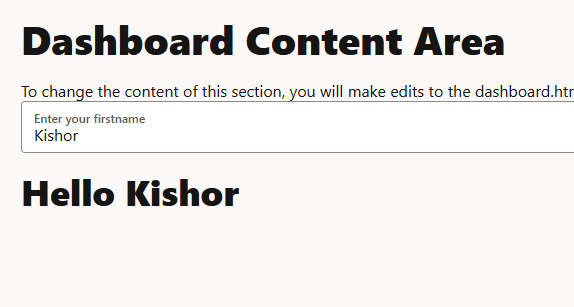
Output:



Note: You can use raw-value in case you want observable to update the changes while you type

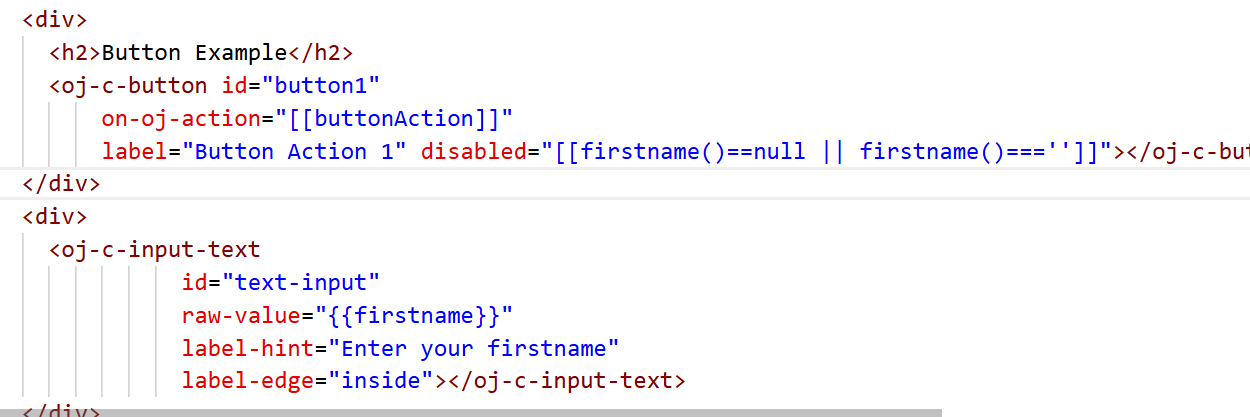


Output:

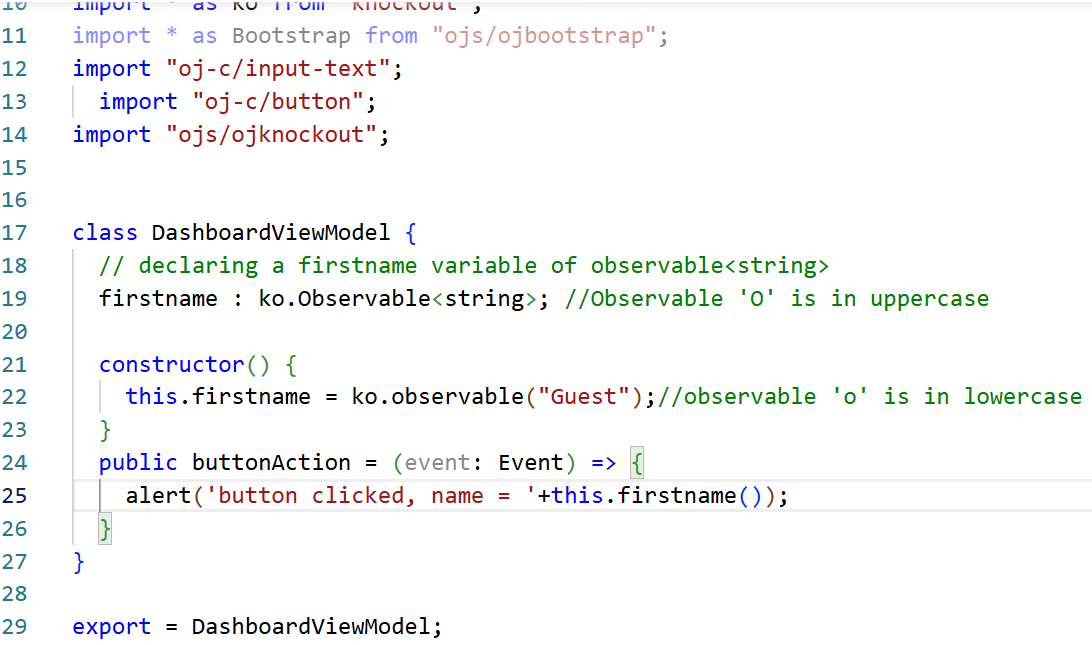


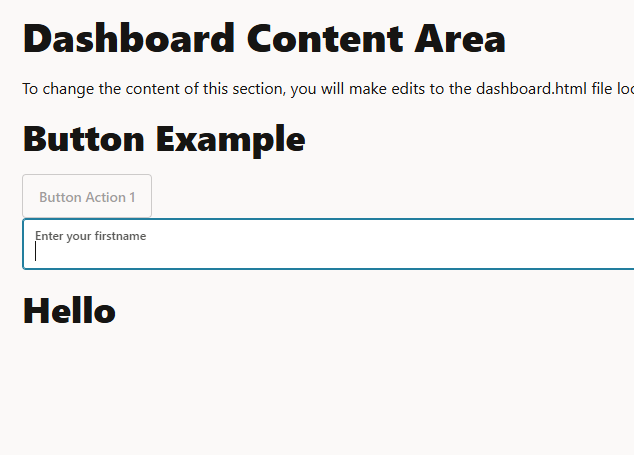
Buttons & Event handling

HTML



TS

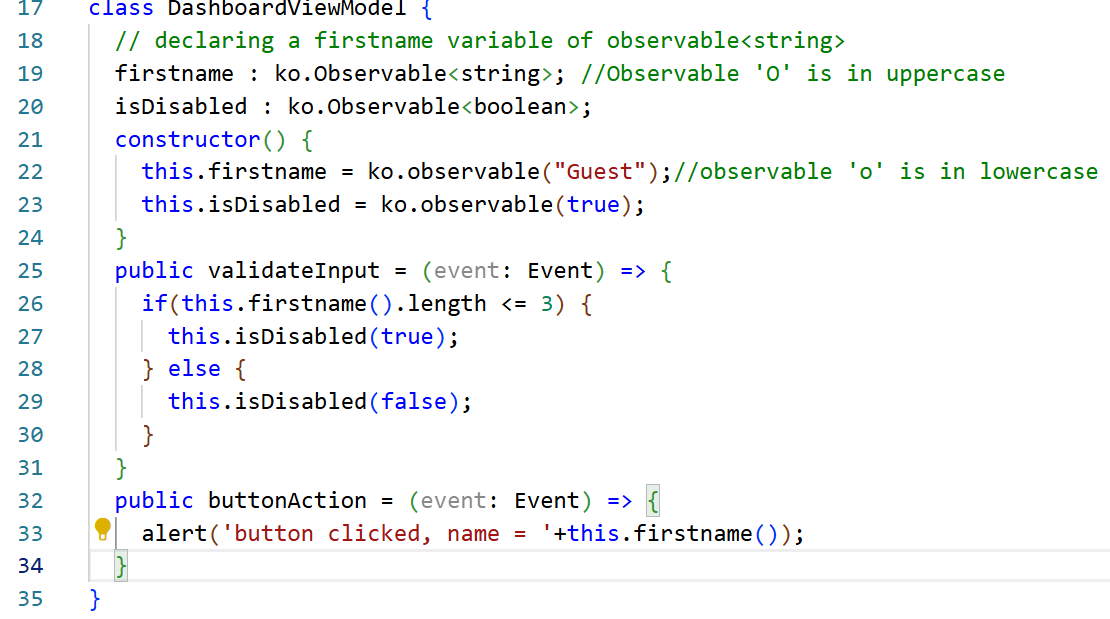


Output  


on-input event & validation

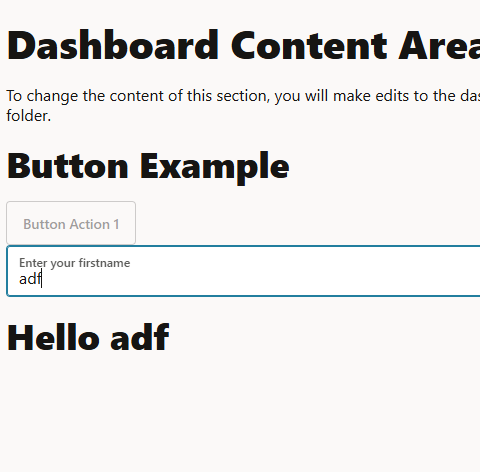
It occurs when you enter the input in the text box

TS

HTML



Output:

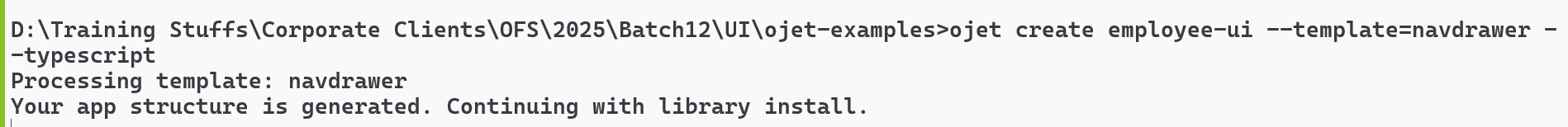


Self-learning

* In the TOC Complete Day2, Day3 & Day4 topics
* From the Cook book go through all the sections atleast 4 to 5 components
* In the Framework complete Core Router & Data Provider mandatorily
* Work on your case-study

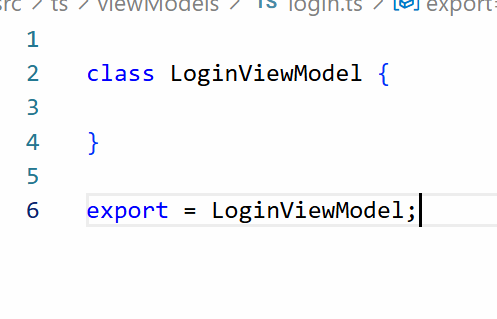
Using routers & access the backend rest api from OJET

Create a new project

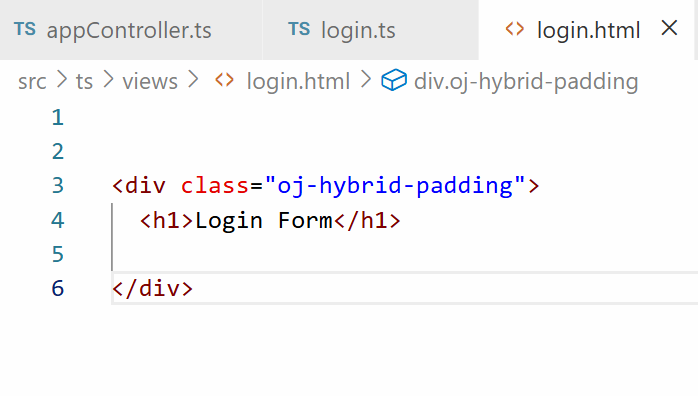


Creating login.ts & login.html

login.ts

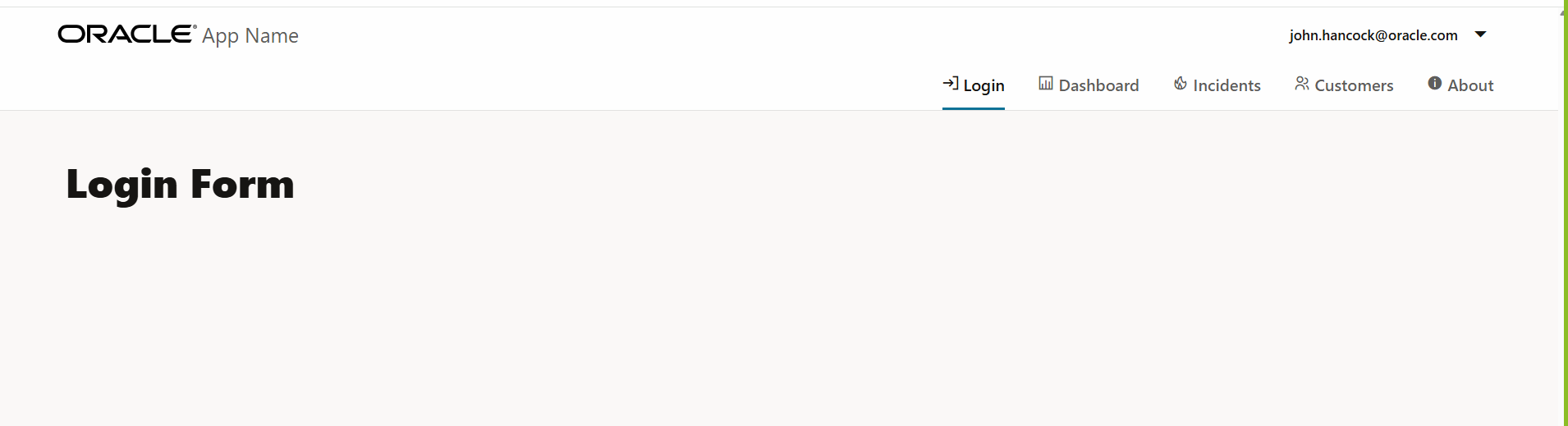


login.html

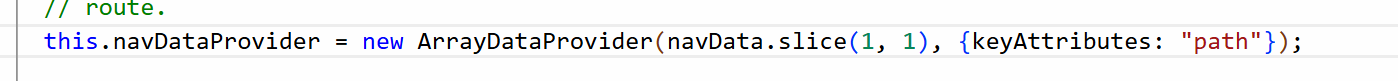


Mention your component in the appController.ts

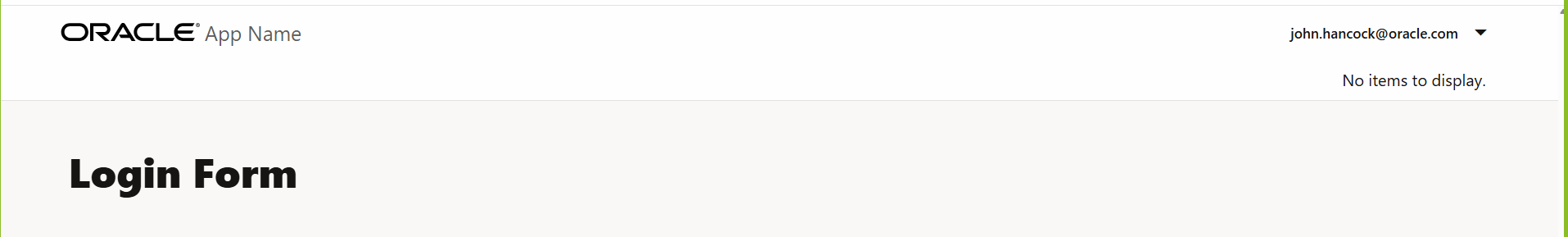


Output  


Let us only show the Login page and go to other pages only when we click on some link or button.

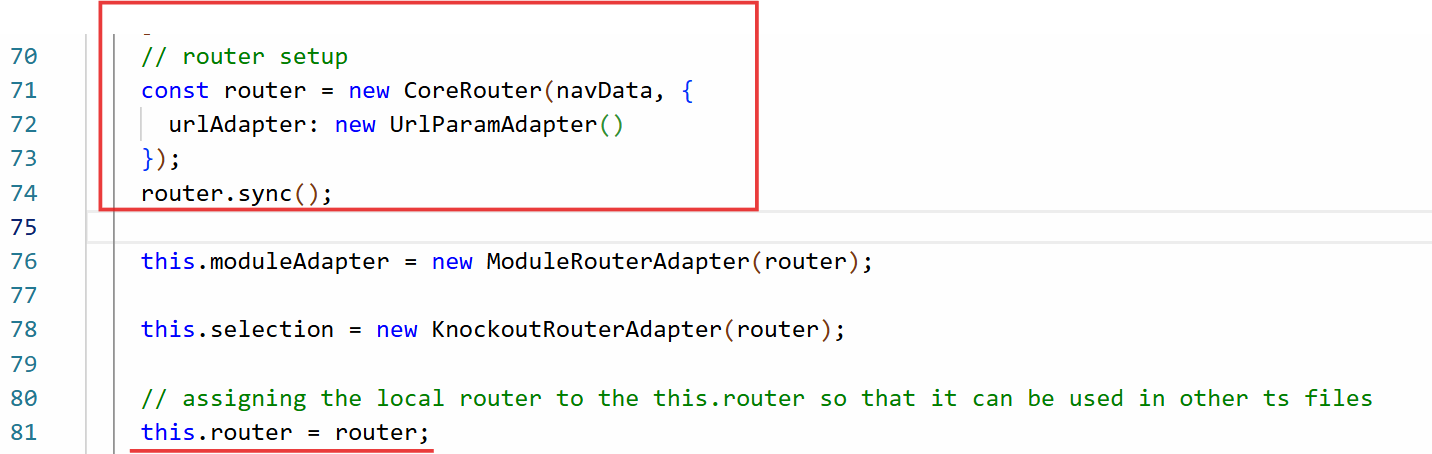


Modify the navDataProvider to take only one route, so that you will see only the very first route in the output

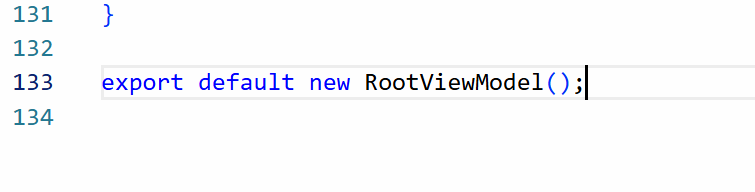


Now we can allow user to enter some credentials and go to success page i.e., dashboard for that we need a router instance that is present in the appController.

Assign the const router to the this.router in the appController.ts



Note: This controller class already has export statement which means you can import it in other ts files, if you navigate till the end you will see below code



Now you can import this RootViewModel and access the router then call the go function to navigate from one page to another

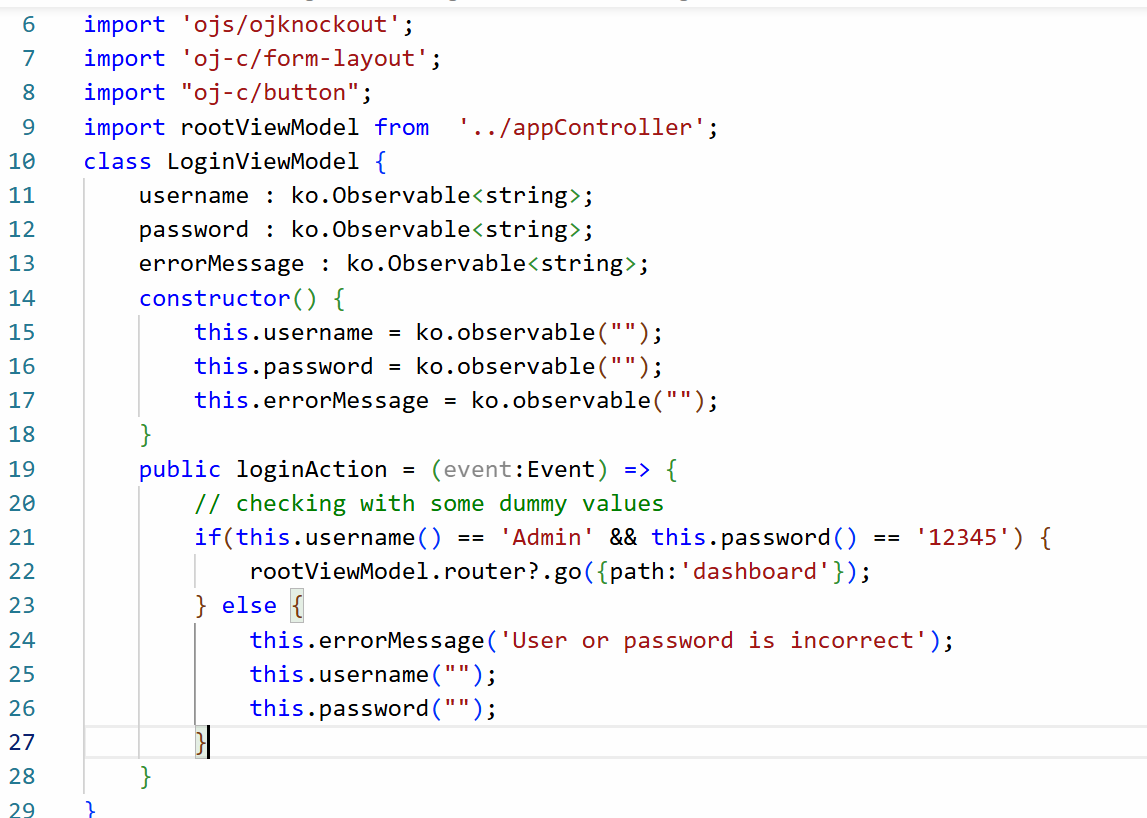
i.e.,

router.go({path:”dashboard”})

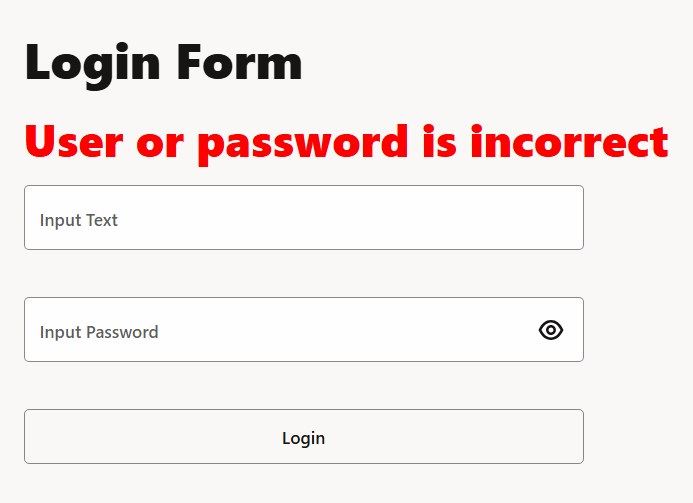
login.html



login.ts



Output:



Rest Data Provider

RestDataProvider is an implementation to DataProvider for fetching JSON data from the REST endpoints, it helps you to directly map the response data to the views like tables

Navigate to cookbook as below

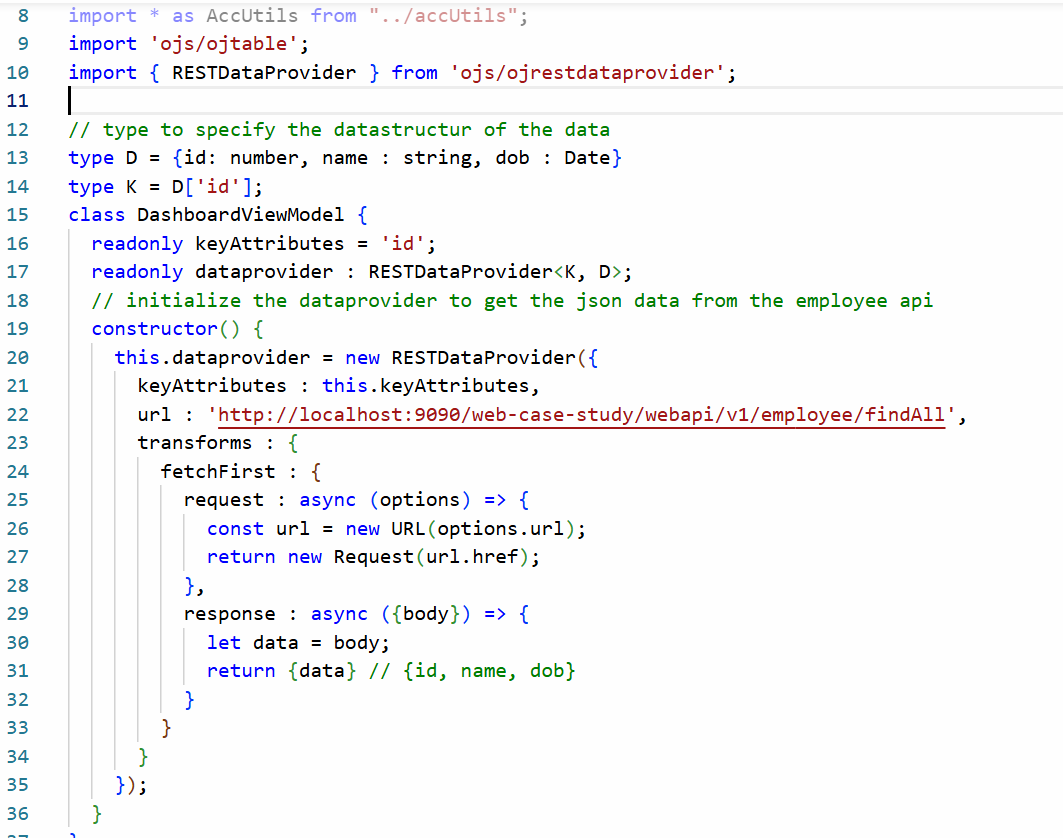


You need to create an object of RESTDataProvider with parameter which is an object of following properties keyAttributes, url, transforms

Note: RestDataProvider must have a data as a property in the response which can be mapped to the data attribute of oj-table

We will access JSON placeholder & display id & name

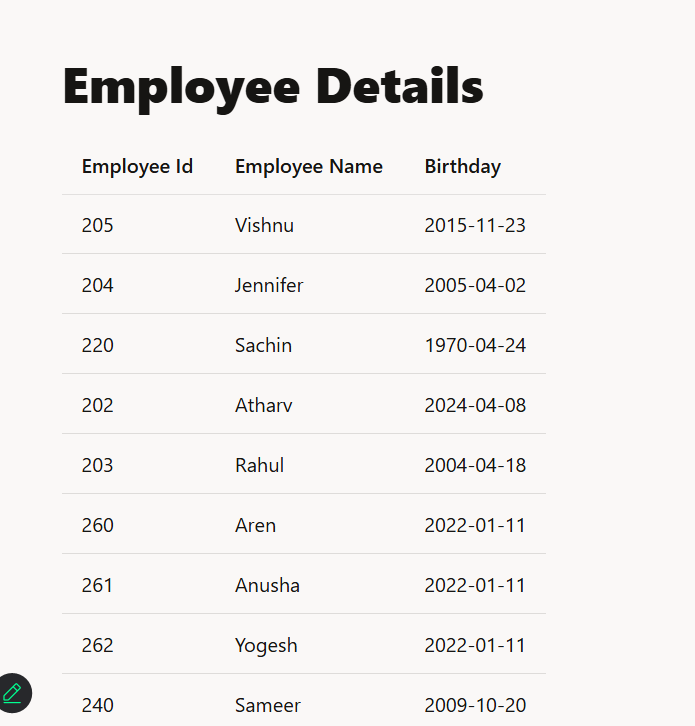
dashboard.ts



dashboard.html



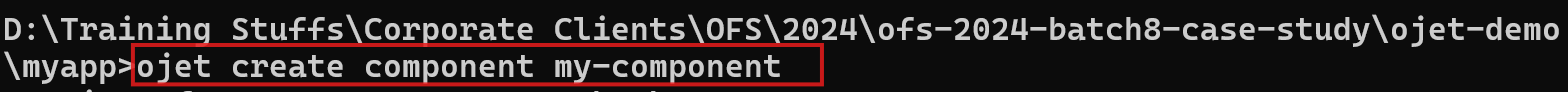
Output:



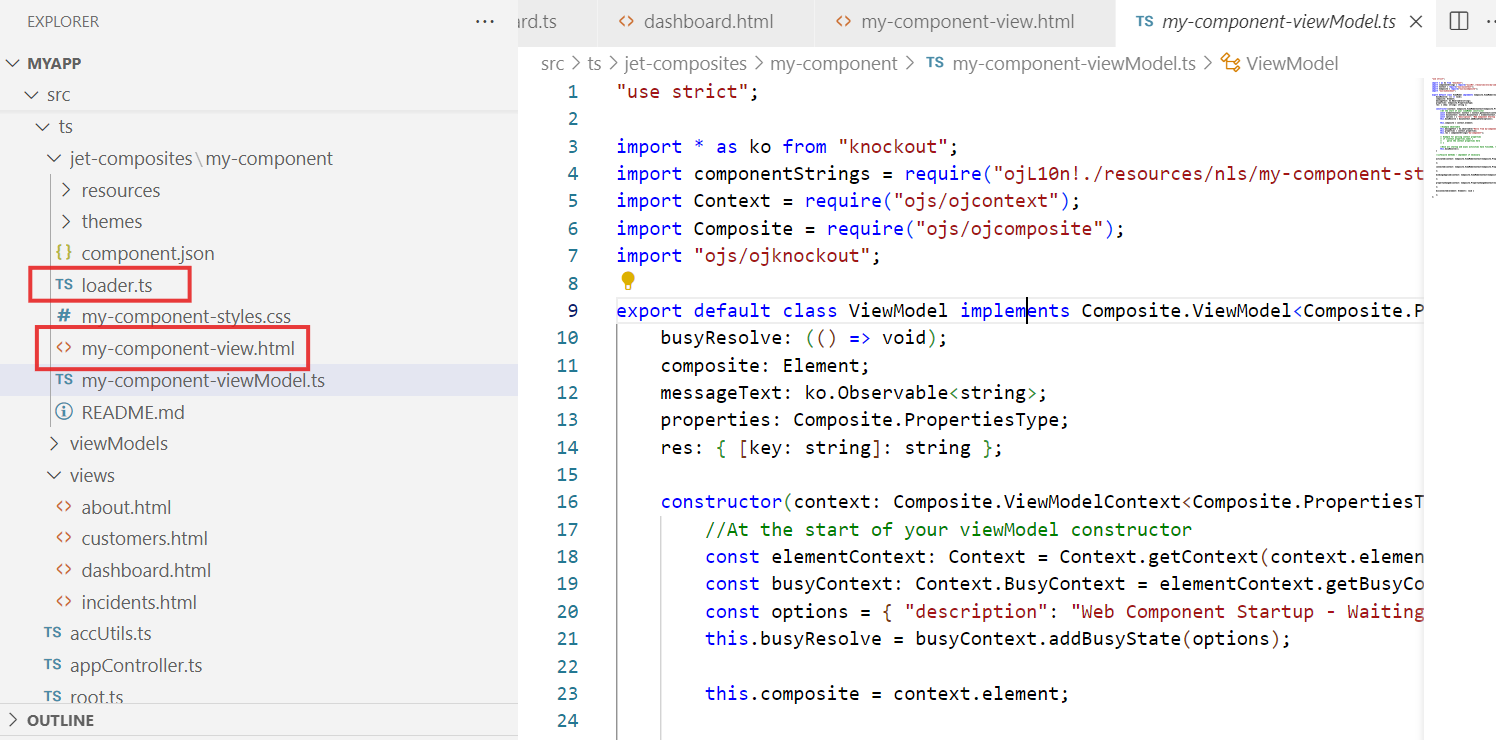
Creating a reusable components

You can create custom components & use them in the view

Note: If you add the custom component to any view and if in case it doesn’t show the output, you need to re-run the application with ojet serve

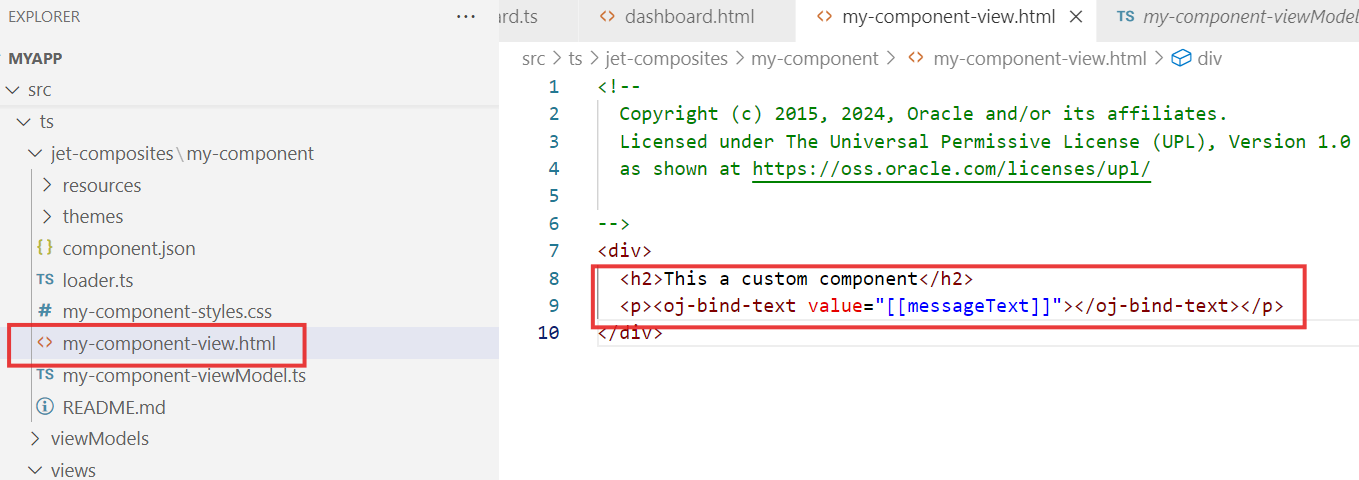


It creates some files, you must locate the views & viewModels and then loader.ts

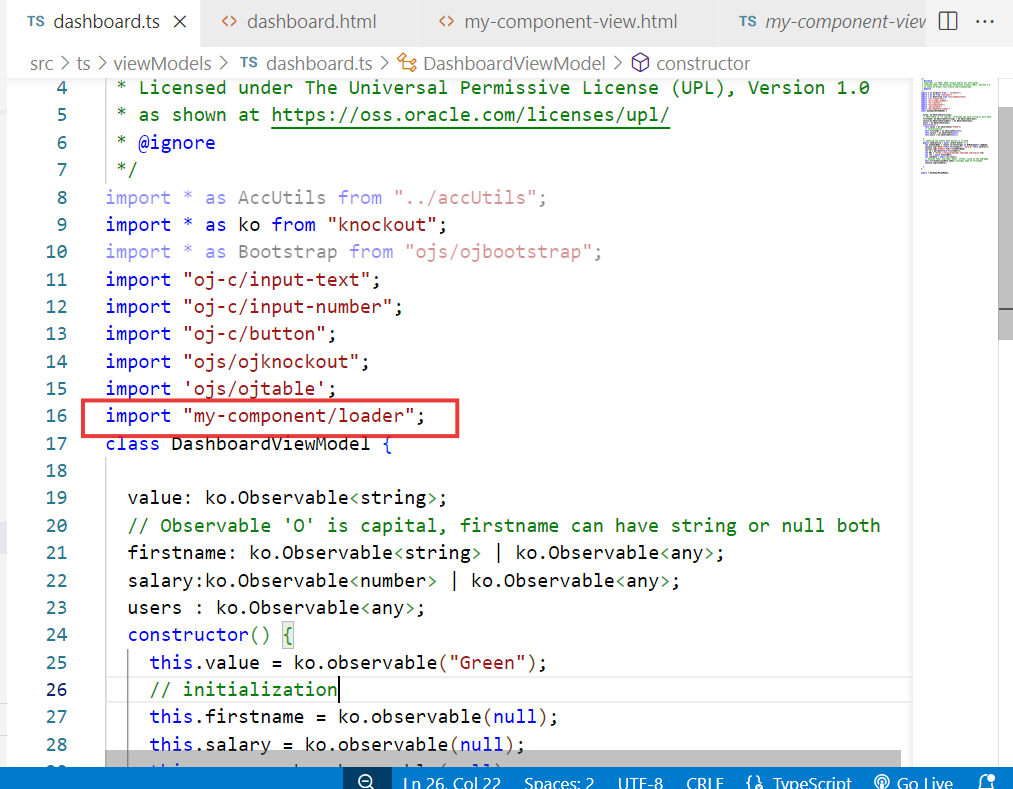


The loader.ts has the component registered code, hence you must import the loader file in case you want to reuse the my-component

Edit some code in the my-component-view.html



dashboard.ts



dashboard.html

Re-run the application

