

# 7BDINoo6W.1 BIG DATA THEORY AND PRACTICE

## Week 3 Seminar Tasks

### Introduction to the Oracle APEX. SQL Basics

#### ORACLE APEX OVERVIEW

Oracle Application Express, often known as Oracle APEX, stands out as a potent *low-code development platform*, enabling the efficient creation of scalable and secure enterprise applications, all within the bounds of a simple web browser. Rooted deeply in its interaction with an Oracle Database, APEX simplifies the application development journey by offering an intuitive graphical interface that is accessible to developers of all experience levels. The platform champions *rapid application development* and ensures that applications are inherently responsive and mobile-friendly, thereby guaranteeing usability across various devices, from desktops to smartphones.

Navigating through the world of data management and integration, APEX demonstrates a remarkable capacity for *direct interaction with Oracle Databases*, leveraging *SQL and PL/SQL* to provide powerful mechanisms for data management and manipulation. It also excels in *data visualisation*, supplying developers with the tools to create insightful visual data representations through various charts, reports, and grids. From a security standpoint, APEX is well-equipped, providing a suite of *built-in security features* that handle authentication, authorisation, and secure session management, as well as tools for the safeguarding of data, ensuring its privacy and integrity.

In the deployment sphere, Oracle APEX offers noteworthy flexibility, *supporting deployment across various environments*, such as on-premises Oracle Databases and Oracle Cloud, while maintaining a high degree of scalability and aptly supporting a wide range of user numbers and data volumes. The platform is further enriched by a vibrant community and expansive ecosystem, which provides developers with access to a variety of plugins, templates, and support via forums and online platforms. Furthermore, APEX enables integration by allowing the *incorporation of various APIs and web services*, thereby ensuring its operability within a larger enterprise IT ecosystem.

With its applicability stretching across various domains, such as finance and healthcare, Oracle APEX finds its utility in developing a wide variety of applications, from simple data trackers to complex Enterprise Resource Planning (ERP) systems, showcasing its versatility and utility in the toolkit of developers and organisations. Thus, Oracle APEX stands prominently as a tool that unifies the capacity to *create secure, scalable, and visually appealing web-based applications* with the simplicity and efficiency that define low-code development.

Oracle APEX finds particular resonance in the realm of a *Big Data Theory and Practice module*, establishing itself as a central platform that navigates the complexities and voluminosity of big data with streamlined data management, analysis, and visualisation capabilities. Notably, APEX's low-code environment minimises the technical barrier, helping students focus on exploring and managing massive datasets without being ensnared in the complexities of traditional coding approaches.

A pivotal addition to the learning trajectory within this context is the introduction of *SQL basics*. Students, while immersed in the APEX environment, embark on a journey through the foundational

principles of SQL, gaining insights into data querying, manipulation, and management at a professional level. This foundational knowledge in SQL not only amplifies their capacity to interact with databases within the APEX platform but also augments their analytical capabilities, enabling them to construct, query, and manipulate databases to glean actionable insights from large data sets.

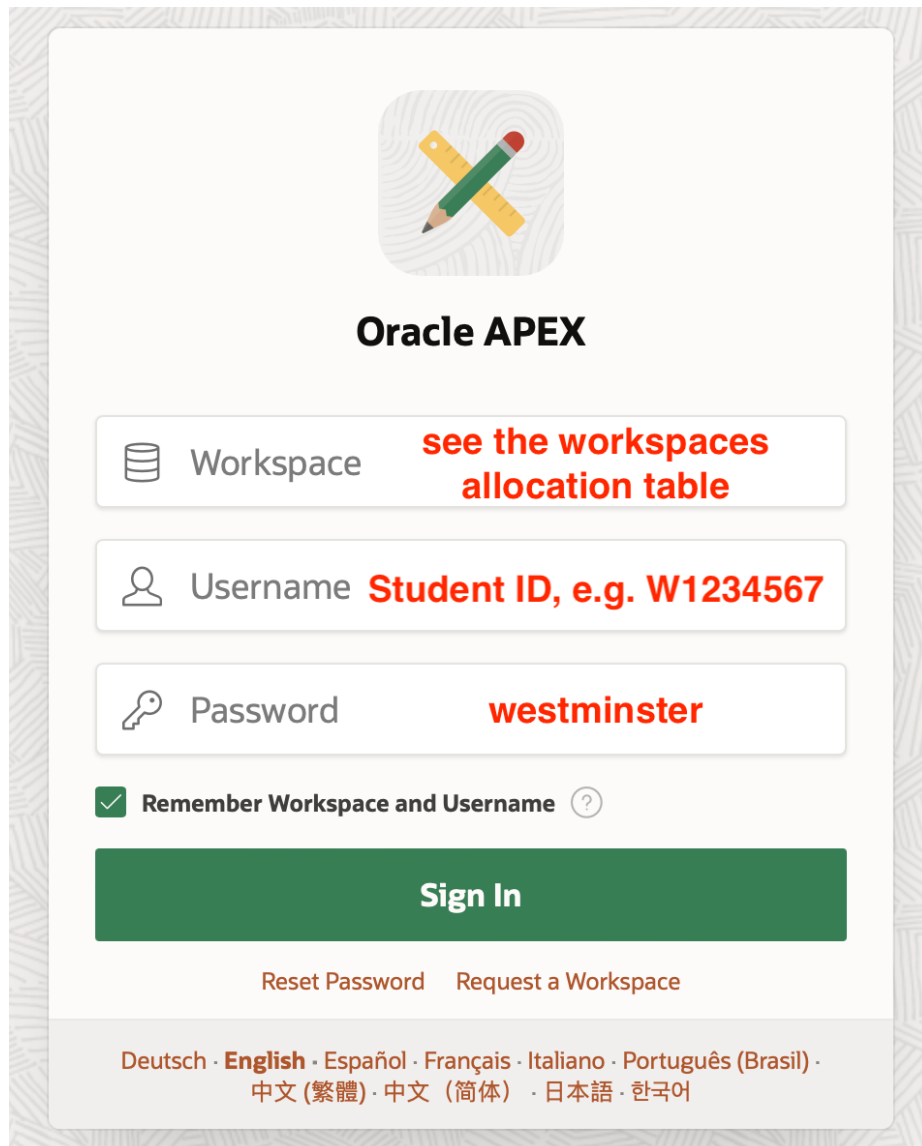
The platform further facilitates students in translating their theoretical knowledge into tangible skills by employing APEX's robust data visualisation tools to articulate complex data into comprehensible visual formats. Additionally, APEX provides a substantive exploration into data integration, offering students practical experiences and insights into amalgamating data from diverse sources, thereby nurturing a holistic understanding of integrative processes within big data contexts.


In its entirety, Oracle APEX not only serves as a significant tool but also as an educational conduit, bridging theoretical knowledge with practical application, particularly in the expansive and intricate domain of big data. Through mastering SQL basics and utilising APEX's vast capabilities in data management, visualisation, and analysis, students are afforded a thorough, practical, and enriched learning journey, navigating through the multifaceted world of big data theory and practice.

## TASK 1. SYSTEM LOGIN


Go to the Oracle APEX login page: [\[Oracle APEX Sign In\]](#)


Fill in your login info. We will use **one workspace for each group**. For your group workspace title, see the table below. The username is **your student ID**, starting with “W” (capital letter). The initial password is “**westminster**”. Please be prepared to reset your password upon your first login.


The image shows the Oracle APEX login page. At the top is the Oracle APEX logo, which consists of a stylized 'X' made of a yellow ruler and a green pencil. Below the logo is the text 'Oracle APEX'. There are three input fields: 'Workspace' with a database icon, 'Username' with a person icon, and 'Password' with a key icon. Each field has a red instruction text to its right: 'see the workspaces allocation table' for Workspace, 'Student ID, e.g. W1234567' for Username, and 'westminster' for Password. Below these fields is a checkbox labeled 'Remember Workspace and Username' with a question mark icon. A large green 'Sign In' button is below the checkbox. At the bottom of the form are two links: 'Reset Password' and 'Request a Workspace'. At the very bottom is a language selection bar with options: Deutsch, English (highlighted), Español, Français, Italiano, Português (Brasil), 中文 (繁體), 中文 (简体), 日本語, and 한국어.




**Oracle APEX**

 Workspace **see the workspaces allocation table**

 Username **Student ID, e.g. W1234567**

 Password **westminster**

☒ Remember Workspace and Username 

**Sign In**

[Reset Password](#) [Request a Workspace](#)

Deutsch · **English** · Español · Français · Italiano · Português (Brasil) · 中文 (繁體) · 中文 (简体) · 日本語 · 한국어

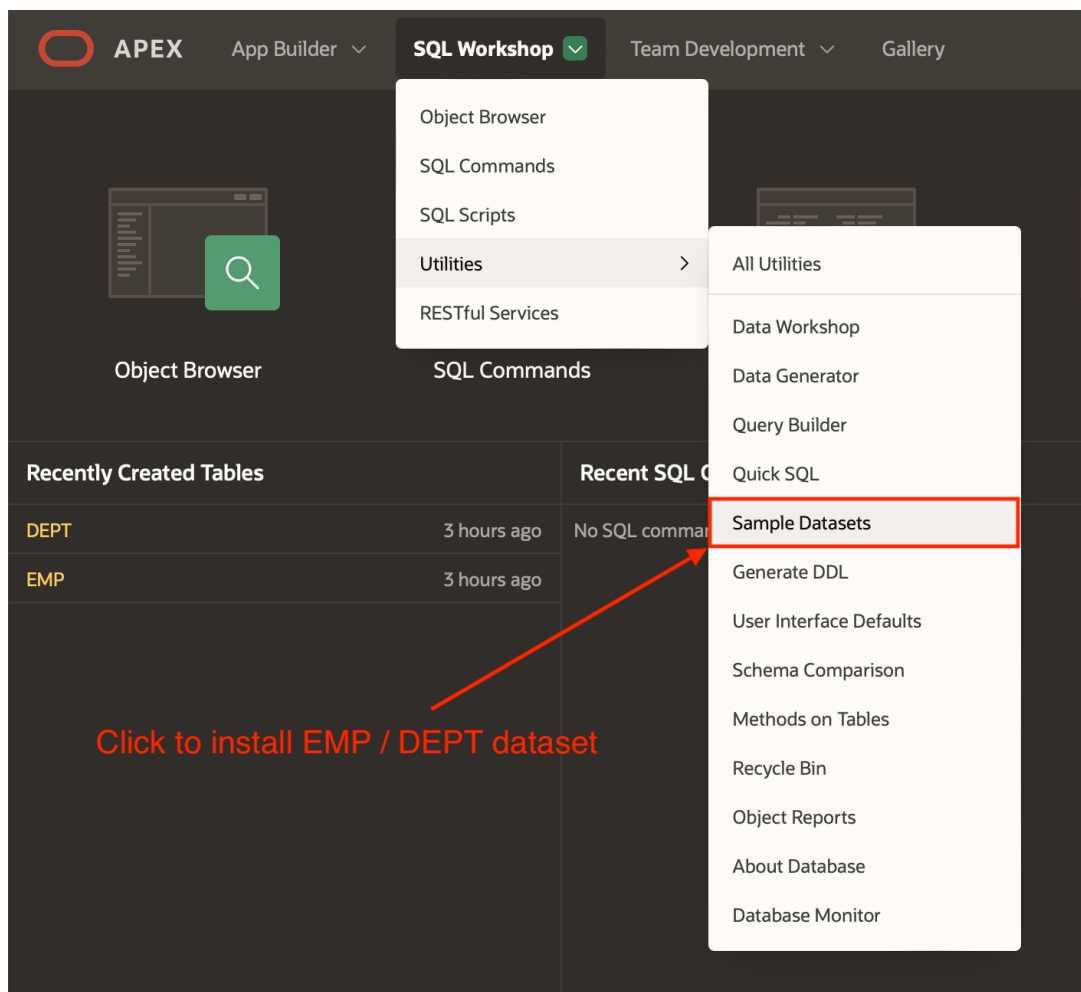
**NOTE:** if you have trouble accessing your account, please let your tutor know tutor (in class or by email) so they can help you unlock your account. The University does NOT support the system, so you **MUST NOT** contact the Service Desk for support.

The Workspaces Allocation Table:

Workspace	Tutor's Name	Group's Title
UOW2023-BDTP-GP4	Stelios	GP4
UOW2023-BDTP-GP1	Karim	GP1
UOW2023-BDTP-GP2	Hafiz	GP2
UOW2023-BDTP-GP1FINTEC-GP13	Hafiz	GP1 Fintech, GP13
UOW2023-BDTP-GP12	Karim	GP12
UOW2023-BDTP-GP3	Karim	GP3
UOW2023-BDTP-GP5	Edmund	GP5
UOW2023-BDTP-GP11	Stelios	GP11
UOW2023-BDTP-GP7	Edmund	GP7
UOW2023-BDTP-GP9	Natalia	GP9
UOW2023-BDTP-GP6	Natalia	GP6
UOW2023-BDTP-GP8	Edmund	GP8
UOW2023-BDTP-GP10	Edmund	GP10
UOW2023-BDTP-GP3B	Vitaly	GP3B

Once you are in Oracle APEX Workspace, let's add the sample tables and create your first sample application:

- ☐ Click on "SQL Workshop"
- ☐ Pick "Utilities", then "Sample Datasets"

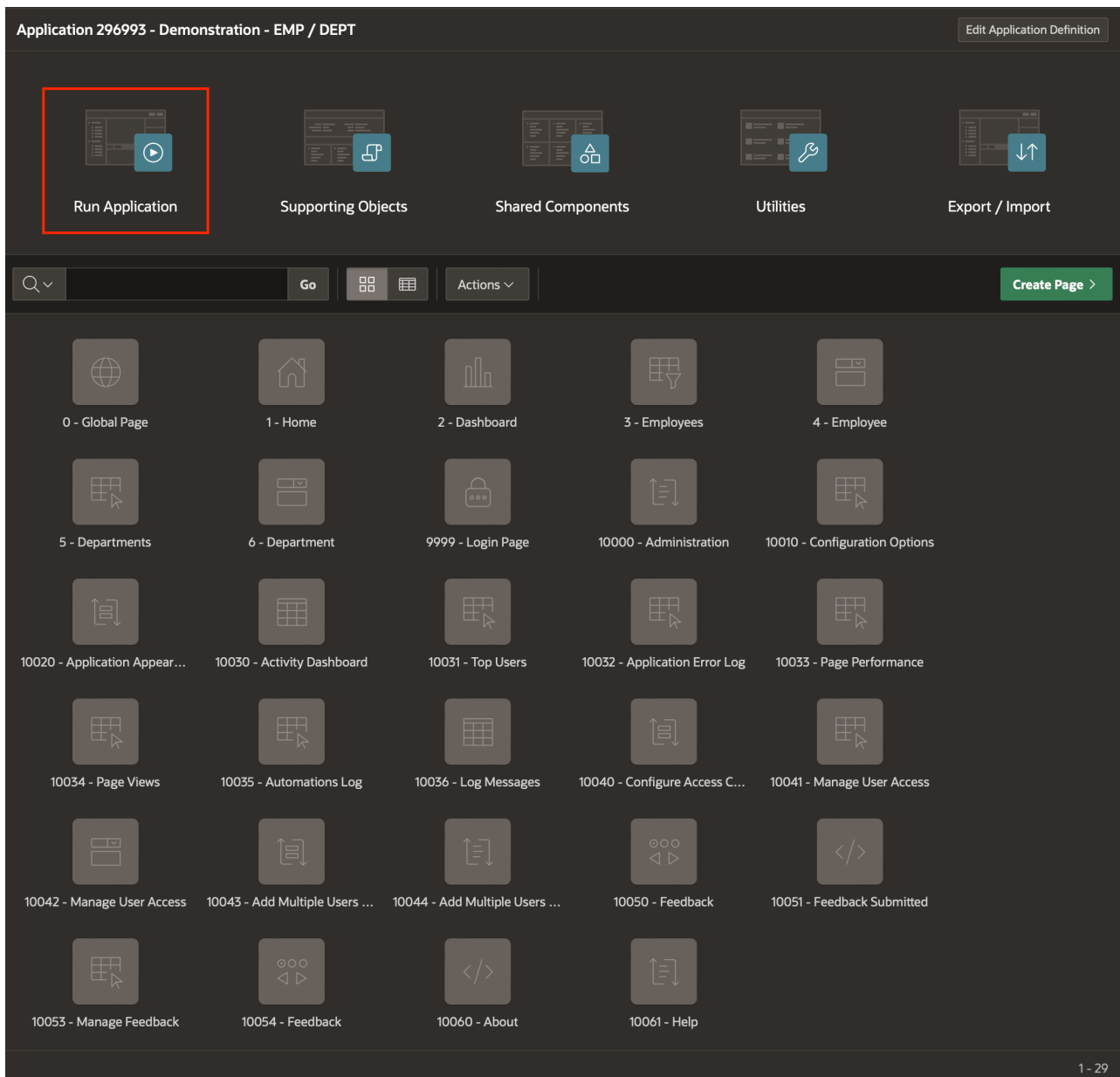


- ☐ Click "Install" on EMP / DEPT dataset and follow the instructions. In the end, click on "Create Application" and follow the instructions.

Now, your first web application on Oracle APEX, contains two tables. “Employees” and “Departments” is created. Well Done!

## TASK 2. UPLOADING THE DATASETS. APPLICATION CREATION.

To run the app, click on “Run Application”.



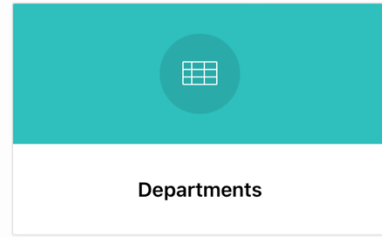
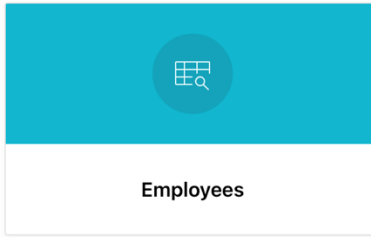
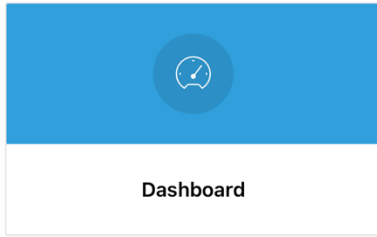
You need to log in again to access the app's starter page.

The *starter page* should look like this, and this is a fully functioning web application:



## Demonstration - EMP / DEPT

Generated based on a Sample Dataset!



The “Employees” and “Departments” sections display the tables with some basic filtering options. Research the tables' data before proceeding with the following tasks.

Click on the “Employees” table.

### Employees

**Department**

☐ SALES (6)

☐ RESEARCH (5)

☐ ACCOUNTING (3)

**Manager**

☐ BLAKE (5)

☐ KING (3)

☐ JONES (2)

☐ CLARK (1)

☐ FORD (1)

☐ SCOTT (1)

**Job**

☐ CLERK (4)

☐ SALESMAN (4)

☐ MANAGER (3)

☐ ANALYST (2)

☐ PRESIDENT (1)

**Salary**

☐ <900 (1)

☐ 900 - 1,300 (4)

☐ 1,300 - 2,000 (3)

☐ 2,000 - 2,500 (1)

☐ >=2,500 (5)

**Total Row Count 14**

**Employees**

	Employee Name	Job	Manager	Hired	Salary	Commission	Department
	CLARK	MANAGER	KING	6/9/1981	2,450		ACCOUNTING
	JONES	MANAGER	KING	4/2/1981	2,975		RESEARCH
	BLAKE	MANAGER	KING	5/1/1981	2,850		SALES
	ALLEN	SALESMAN	BLAKE	2/20/1981	1,600	300	SALES
	WARD	SALESMAN	BLAKE	2/22/1981	1,250	500	SALES
	MARTIN	SALESMAN	BLAKE	9/28/1981	1,250	1400	SALES
	TURNER	SALESMAN	BLAKE	9/8/1981	1,500	0	SALES
	JAMES	CLERK	BLAKE	12/3/1981	950		SALES
	MILLER	CLERK	CLARK	1/23/1982	1,300		ACCOUNTING
	SCOTT	ANALYST	JONES	12/9/1982	3,000		RESEARCH
	FORD	ANALYST	JONES	12/3/1981	3,000		RESEARCH
	ADAMS	CLERK	SCOTT	1/12/1983	1,100		RESEARCH
	SMITH	CLERK	FORD	12/17/1980	800		RESEARCH
	KING	PRESIDENT		11/17/1981	5,000		ACCOUNTING

Create

Using the left-hand side filtering panel, Let’s display all the employees with King as their manager.

The result should be the following:

Employees							Create
	Employee Name	Job	Manager	Hired	Salary	Commission	Department
	BLAKE	MANAGER	KING	5/1/1981	2,850		SALES
	CLARK	MANAGER	KING	6/9/1981	2,450		ACCOUNTING
	JONES	MANAGER	KING	4/2/1981	2,975		RESEARCH

Return to the initial Employees table and create the new tuple by clicking the “Create” button.

Craft a new tuple, ensuring the employee is *distinctive and unique*. Embrace creativity; if a duplicate of an existing employee is used, the system will generate an error. An example showcasing the creation of an employee named “POTTER” is depicted in the screenshot attached below:

Employee
×

Employee

Employee Name  
POTTER

Job  
CLERK

Manager  
KING

Hired  
10/01/1981

Salary  
700

Commission

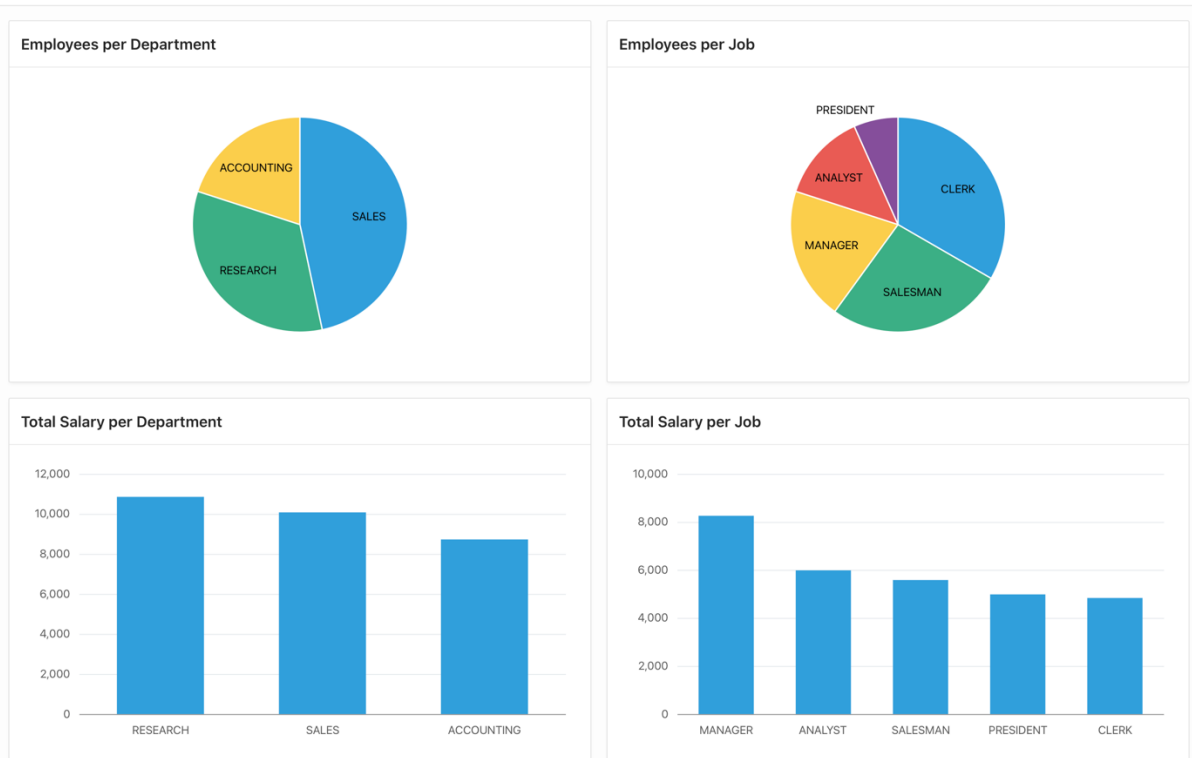
Department  
SALES

Go back to the app’s starting page.

Click on “Dashboard”. It contains the basic visualisations of the two sample tables.

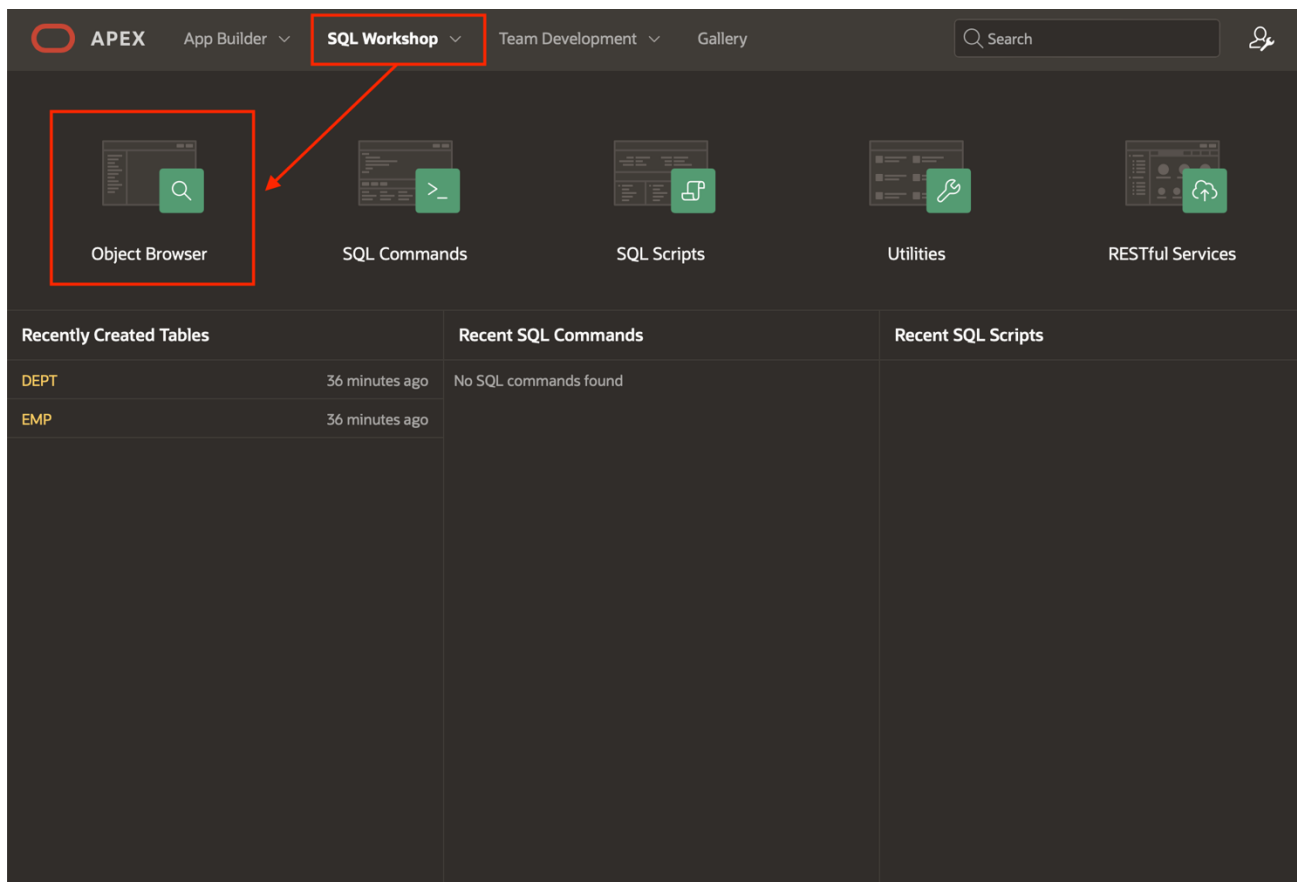


## Dashboard



Now, let's return to the Oracle APEX environment and explore the app from the inside.

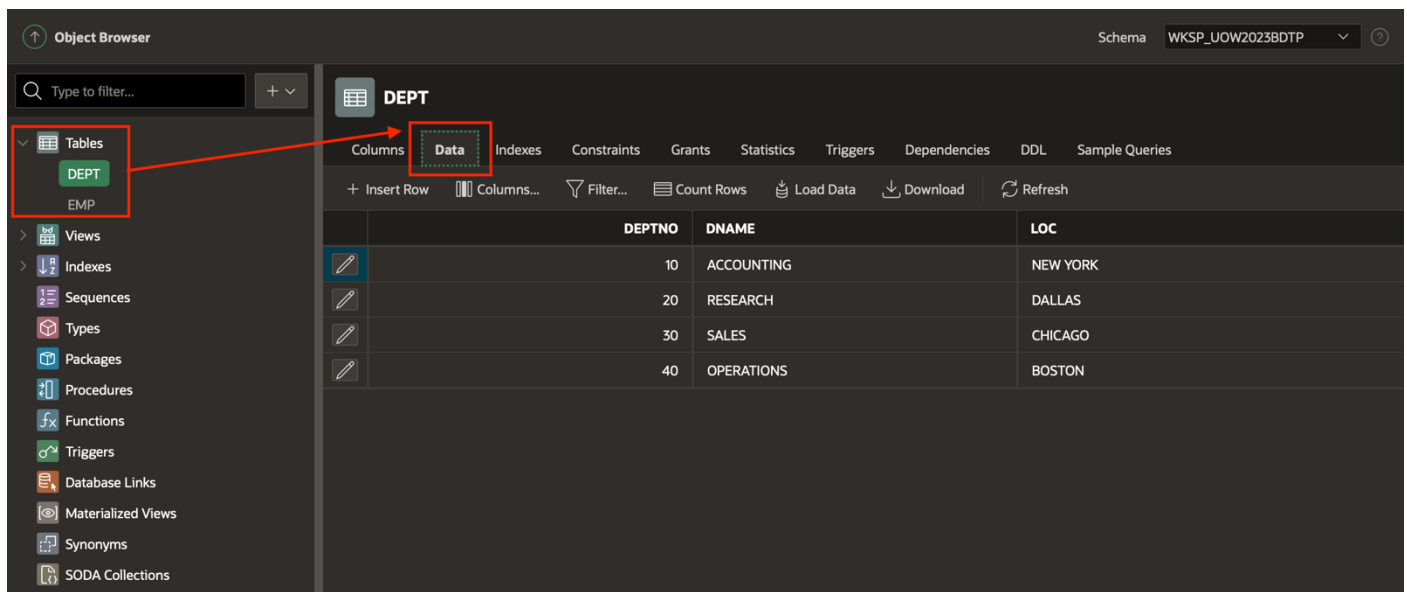
Click on "SQL Workshop" on the upper panel, then on "Object Browser".



The Object Browser in Oracle Application Express (APEX) is a pivotal tool that facilitates the intuitive management and navigation of database objects within a user's workspace. It offers

developers a comprehensive and user-friendly interface to interact with, manipulate, and manage various database objects directly within the APEX environment, such as tables, views, packages, etc. Developers can explore object attributes, execute scripts, and alter object structures (such as modifying table columns), all without needing to engage directly with SQL scripts or external database management tools. The Object Browser not only simplifies the visual management of database objects, enhancing developers' ability to maintain data consistency and integrity across applications, but also lowers the technical barrier for those who may not possess an in-depth understanding of SQL or database management, making APEX a more accessible development platform.

Click on the particular table in the left-hand side panel of the Object Browser. You can see the table's characteristics, like "Data", which shows the full content of the table and provides capabilities of its management (e.g., insert/delete rows):



The screenshot shows the Oracle APEX Object Browser interface. On the left, the 'Object Browser' panel is open, showing a tree view of database objects. The 'Tables' folder is expanded, and the 'DEPT' table is selected. An orange arrow points from the 'DEPT' table in the left panel to the 'Data' tab in the right panel. The right panel shows the 'DEPT' table with the 'Data' tab selected. The table data is displayed in a grid with columns DEPTNO, DNAME, and LOC. The table has four rows of data.

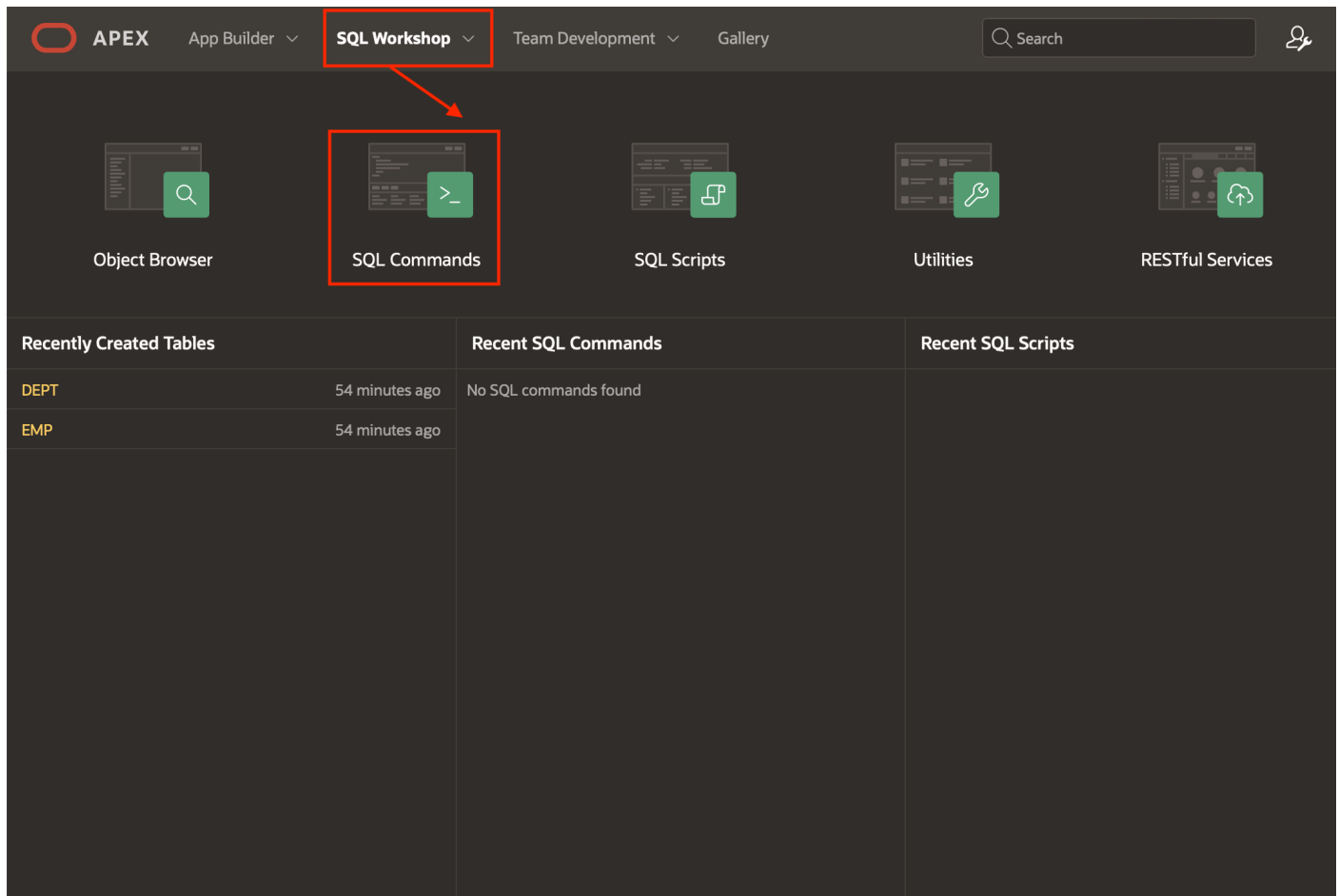
DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Examine the two tables to identify what constitutes the *primary key* and any *foreign keys* within them.

After researching the table content, we can continue with our first SQL queries!

## TASK 3. SIMPLE SQL QUERIES

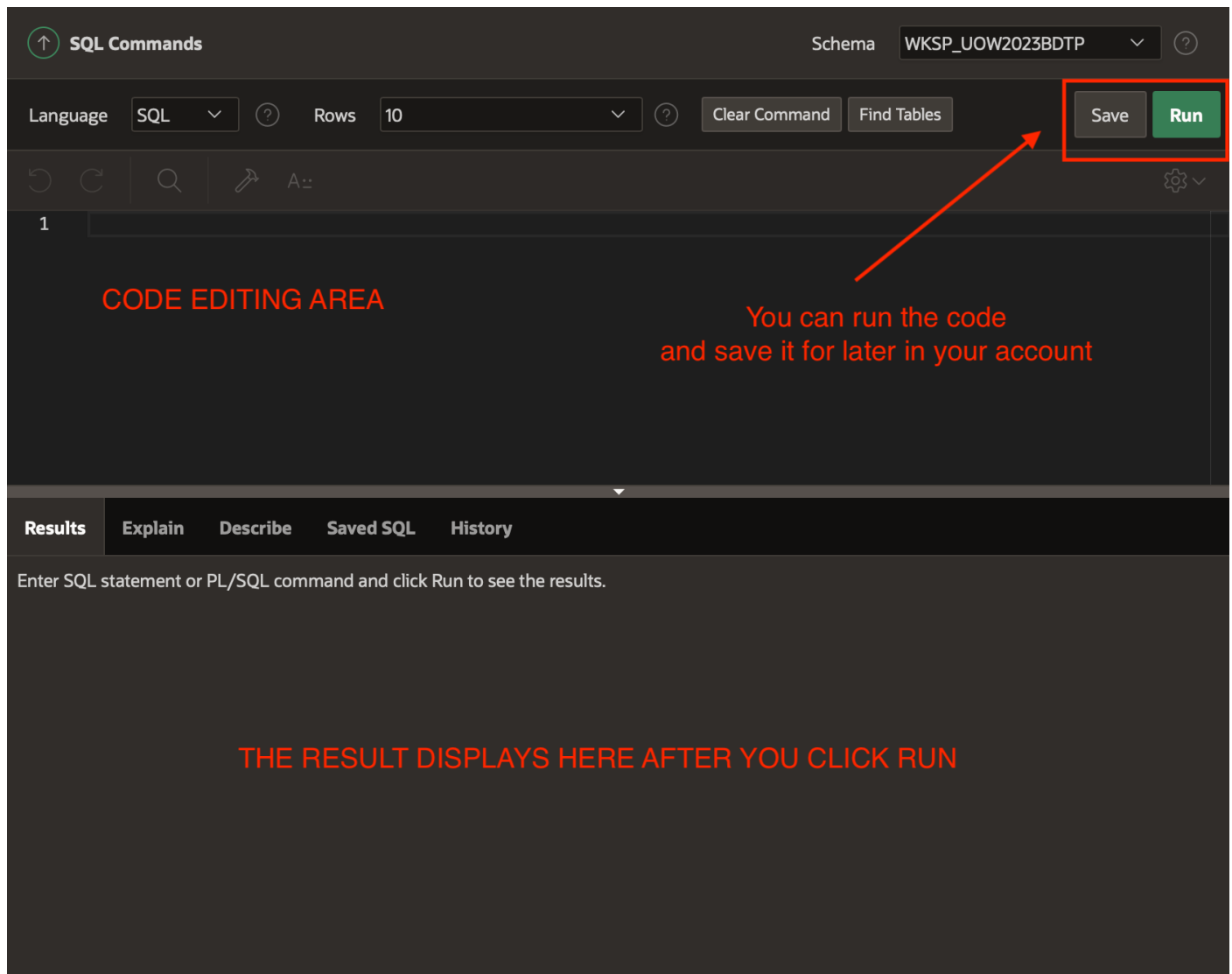
Click on “SQL Workshop” again and choose “SQL Commands.”



The screenshot shows the APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop' (highlighted with a red box and an arrow pointing to the 'SQL Commands' icon), 'Team Development', and 'Gallery'. Below the navigation bar are five icons: 'Object Browser', 'SQL Commands' (highlighted with a red box), 'SQL Scripts', 'Utilities', and 'RESTful Services'. Below the icons are three panels: 'Recently Created Tables' showing 'DEPT' and 'EMP' tables, 'Recent SQL Commands' showing 'No SQL commands found', and 'Recent SQL Scripts'.

Recently Created Tables	Recent SQL Commands	Recent SQL Scripts
<b>DEPT</b> 54 minutes ago	No SQL commands found	
<b>EMP</b> 54 minutes ago		

This is the environment to write the SQL commands:



Now, you will explore the basic SQL querying techniques using the classic `EMP` and `DEPT` tables. Ensure to examine the table structures before attempting the queries to familiarise yourself with the available columns.

Write the SQL expressions that will display the following. Some of the SQL commands will be new for you, so read the hints and use the SQL manuals on BB.

The answers are provided below. Please try to do the tasks by yourself, then check the answers.

### Task 1: SELECT and Project Operations

1. Retrieve all columns for all employees from the `EMP` table.
2. Select only the `ENAME` and `SAL` columns for all employees in the `EMP` table.
3. Fetch all department names from the `DEPT` table.

### Task 2: Basic Filtering using WHERE Clause

4. Retrieve details of employees with a salary above 3000.
5. Extract the names and locations of departments located in 'Dallas'.

#### HINT:

In the context of a SELECT statement, the WHERE clause plays a pivotal role in filtering the retrieved data from a database by imposing specified conditions that the data must adhere to. When a SELECT statement is executed with a WHERE clause, only the rows for which the WHERE

condition is true are returned in the query result. This allows users to narrow down the results from potentially vast datasets to a manageable subset that satisfies particular criteria. Hence, the WHERE clause provides an efficient means to extract precisely the data needed for analytical, reporting, or operational purposes, ensuring that queries can be designed to furnish specific, relevant information rather than unduly large and unwieldy datasets.

### **Task 3: JOIN Operations**

6. Utilize an INNER JOIN to retrieve employee names (`ENAME`) alongside their respective department names (`DNAME`).
7. Perform a LEFT JOIN to fetch all employee names along with department names, ensuring that employees with no department assignments are also included.

#### **HINT:**

In SQL, a JOIN operation is used to combine rows from two or more tables based on a related column between them. Different types of JOIN operations allow data retrieval in various ways, depending on the relationship and data requirement.

An INNER JOIN retrieves rows from both tables that satisfy the specified condition, omitting rows that don't meet the condition.

A LEFT JOIN (or LEFT OUTER JOIN) returns all rows from the left table and the matched rows from the right table. If there's no match, the result from the right side will contain NULL.

A RIGHT JOIN (or RIGHT OUTER JOIN) provides all the rows from the right table and the matched rows from the left table. If there's no match, the result from the left side will contain NULL.

A FULL OUTER JOIN returns all rows when there is a match in one of the left or right table records. It returns NULL on the side where there is no match.

A SELF JOIN is a technique to join a table with itself using either INNER JOIN or LEFT JOIN, usually involving aliasing to prevent confusion between the table's instances.

A CROSS JOIN produces the Cartesian product of two tables, meaning it returns every possible combination of rows from the two joined tables.

A NATURAL JOIN automatically performs a JOIN based on all columns in the tables that have the same name. It selects rows from the two tables that have equal values in all matched columns.

### **Task 4: UNION Operation**

8. Construct a query using UNION to create a consolidated list of all unique job titles from both the `EMP` and `DEPT` tables (using `JOB` and `DNAME` respectively, assuming roles might be similar to department names for the context of this exercise).

Note: Ensure to use columns that are logically compatible with UNION operations.

### **Task 5: Aggregate Functions**

9. Determine the highest salary in the `EMP` table.
10. Calculate the average salary for all employees.

#### **HINT:**

Aggregate functions in SQL provide a means to compute a single value from a set of values, often to yield summary or statistical information about datasets. These functions operate on a group of rows, either divided by the GROUP BY clause or considering the entire result set as a single group when used without GROUP BY. Common aggregate functions include SUM, AVG, COUNT, MIN, and MAX, each providing distinct kinds of summary information, such as total, average, count of rows, minimum, and maximum, respectively. They are typically used in SELECT statements, offering insights into data distributions, quantities, and variations directly from the database, thereby facilitating a deeper understanding and analysis of the stored data. Consequently, aggregate functions enhance data analysis by efficiently providing summarised data directly through query operations.

### **Task 6: GROUP BY Clause**

11. Find the total number of employees in each department.

12. Calculate the average salary per department, grouping by the department number.

**HINT:**

The "GROUP BY" clause in SQL serves to arrange identical data into summary rows and is especially pertinent when aggregate functions are used in SELECT statements. This clause groups the result set (derived from a SELECT statement) into summary rows by the values of one or more columns. Each row in the resulting group typically represents a unique combination of values in the grouped columns.

**Task 7: CROSS PRODUCT (CROSS JOIN)**

13. Perform a CROSS JOIN between the `EMP` and `DEPT` tables and observe the output.

Note: Use caution with cross joins on larger tables, as they can produce significantly large result sets.

**Task 8: Sorting Results with ORDER BY**

14. Retrieve all employees' names and salaries, ordered by salary in descending order.

15. Extract department names, ordered alphabetically.

**Task 9: Working with Dates**

16. Extract all employee names who were hired in the year 1981.

17. Identify the employee who has the earliest hire date.

Please ensure to validate your queries by examining the output and verifying that the returned data meets the requirements of each task. These exercises should provide you with a foundational understanding of basic SQL operations and prepare you for more advanced querying techniques in future tutorials. Happy querying!

Ask your tutor if there are any questions when interpreting query results.

## **TASK 3 SOLUTIONS.**

### **Task 1: SELECT and Project Operations**

1. -- Retrieve all columns and rows from the EMP table.  
`SELECT * FROM EMP;`
2. -- Select only employee names and salaries from the EMP table.  
`SELECT ENAME, SAL FROM EMP;`
3. -- Extract all department names from the DEPT table.  
`SELECT DNAME FROM DEPT;`

### **Task 2: Basic Filtering using WHERE Clause**

4. -- Retrieve employee details with a salary greater than 3000.  
`SELECT * FROM EMP WHERE SAL > 3000;`
5. -- Get department names and locations for departments located in 'Dallas'.  
`SELECT DNAME, LOC FROM DEPT WHERE LOC = 'DALLAS';`

### **Task 3: JOIN Operations**

6. -- Join EMP and DEPT tables to show employee names and their respective department names.  
`SELECT E.ENAME, D.DNAME FROM EMP E INNER JOIN DEPT D ON E.DEPTNO = D.DEPTNO;`
7. -- Use a left join to display all employee names with their department names, including those without departments.  
`SELECT E.ENAME, D.DNAME FROM EMP E LEFT JOIN DEPT D ON E.DEPTNO = D.DEPTNO;`

### **Task 4: UNION Operation**

8. -- Combine and display unique job titles from EMP and department names from DEPT.  
`SELECT JOB FROM EMP  
UNION  
SELECT DNAME FROM DEPT;`

### **Task 5: Aggregate Functions**

9. -- Determine the highest salary in the EMP table.  
`SELECT MAX(SAL) FROM EMP;`
10. -- Calculate the average salary of employees in the EMP table.  
`SELECT AVG(SAL) FROM EMP;`

### **Task 6: GROUP BY Clause**

11. -- Count employees in each department and group the results by department number.  
`SELECT DEPTNO, COUNT(*) FROM EMP GROUP BY DEPTNO;`
12. -- Calculate average salary and group the results by department number.  
`SELECT DEPTNO, AVG(SAL) FROM EMP GROUP BY DEPTNO;`

### **Task 7: CROSS PRODUCT (CROSS JOIN)**

13. -- Produce a Cartesian product of EMP and DEPT tables.  
`SELECT * FROM EMP CROSS JOIN DEPT;`

## **Task 8: Sorting Results with ORDER BY**

-14. - Select employee names and salaries, ordered by salary in descending order.  
SELECT ENAME, SAL FROM EMP ORDER BY SAL DESC;

15. -- Select and order department names alphabetically.  
SELECT DNAME FROM DEPT ORDER BY DNAME;

## **Task 9: Working with Dates**

16. -- Select employee names who were hired in the year 1981.  
SELECT ENAME FROM EMP WHERE EXTRACT(YEAR FROM HIREDATE) = 1981;

17. -- Identify the employee with the earliest hire date.  
SELECT ENAME, HIREDATE FROM EMP ORDER BY HIREDATE ASC FETCH FIRST ROW ONLY;

These answers, along with the accompanying comments, are aimed to provide clarity regarding the intent and functionality of each SQL query in relation to the tasks provided. It's crucial that the students understand not just the syntax but also the logical underpinning of each query's construction.