



Faculty of Technology and Engineering

Chandubhai S. Patel Institute of Technology (CSPIT)

Department of Computer Science & Engineering

Date: / /

Laboratory Manual

Academic Year	:	2024-25	Semester	:	4
Course code	:	CSE206	Course name	:	DATABASE MANAGEMENT SYSTEM

Practical - 6

- **Aim:** - You are a database administrator for a manufacturing and consulting company. The company maintains two primary tables: Product and Employee Company (emp_company). You are tasked with solving business queries related to order quantities, employee salaries, and company analysis using SQL grouping and aggregate functions. To manipulate and retrieve meaningful insights using grouping and aggregate functions in SQL while adhering to database constraints and integrity rules.

Constraints –

- **Not Null Constraints:** Critical fields such as product numbers, employee names, and salaries must not contain null values.
- **Unique Constraints:** Ensure the integrity of unique fields like Product_no and ENAME.
- **Check Constraints:** Validate that quantities and salaries have valid positive values.

1. Product Table: Tracks order details for various products.

- Detorder_no (Primary Key)
- Product_no (Not Null, Unique)
- Qty_order (Not Null, Check: Greater than zero)

2. emp_company Table: Tracks employees, their companies, and salaries.

- ENAME (Not Null, Unique)
- CNAME (Not Null)
- SALARY (Not Null, Check: Greater than zero)

Tasks:-

The logistics department has provided the following product order details to be inserted into the Product table:

1. Insert Values:

- Insert the above data into the Product table.

The screenshot displays the Oracle SQL Developer interface. On the left, the 'Connections' pane shows the 'C23CS32' database selected. The main window is titled 'C23CS32_PRACTICAL_6.sql' and contains an SQL Worksheet. The script includes several INSERT statements for the 'Product' table, followed by a SELECT statement to verify the data. The 'Query Result' pane at the bottom shows the output of the SELECT statement, displaying 7 rows of data with columns DETORDER_NO, PRODUCT_NO, and QTY_ORDER.

```

VALUES ('019003', 'P00004', 2);

-- Insert row 5
INSERT INTO Product (Detorder_no, Product_no, Qty_order)
VALUES ('019004', 'P00003', 6);

-- Insert row 6
INSERT INTO Product (Detorder_no, Product_no, Qty_order)
VALUES ('019005', 'P00005', 2);

-- Insert row 7
INSERT INTO Product (Detorder_no, Product_no, Qty_order)
VALUES ('019006', 'P00004', 7);

-- Step 4: Display all rows from the Product table to verify the data
SELECT * FROM Product;

SELECT Product_no, SUM(Qty_order) AS Total_Quantity

```

	DETORDER_NO	PRODUCT_NO	QTY_ORDER
1	019001	P00001	10
2	019001	P00002	3
3	019002	P00001	4
4	019003	P00004	2
5	019004	P00003	6
6	019005	P00005	2
7	019006	P00004	7

2. Total Quantity per Product:

- Retrieve the product numbers and total quantities ordered for each product.

The screenshot displays the Oracle SQL Developer interface. On the left, the 'Connections' pane shows the 'C23CS32' database selected. The main window is titled 'C23CS32_PRACTICAL_6.sql' and contains the following SQL script:

```
VALUES ('019005', 'P00005', 2);

-- Insert row 7
INSERT INTO Product (Detorder_no, Product_no, Qty_order)
VALUES ('019006', 'P00004', 7);

-- Step 4: Display all rows from the Product table to verify the data
SELECT * FROM Product;

SELECT Product_no, SUM(Qty_order) AS Total_Quantity
FROM Product
GROUP BY Product_no;

SELECT Product_no, SUM(Qty_order) AS Total_Quantity
FROM Product
WHERE Product_no IN ('P00001', 'P00004')
GROUP BY Product_no;
```

The 'Query Result' pane at the bottom shows the output of the last query, displaying a table with two columns: 'PRODUCT_NO' and 'TOTAL_QUANTITY'.

PRODUCT_NO	TOTAL_QUANTITY
1 P00002	3
2 P00001	14
3 P00003	6
4 P00004	9
5 P00005	2

3. Filter Specific Products:

- Retrieve the product numbers and total quantities ordered for products P00001 and P00004.

The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' pane with 'C23CS32' selected. The main window displays a SQL worksheet with the following code:

```
VALUES ('019006', 'P00004', 7);

-- Step 4: Display all rows from the Product table to verify the data
SELECT * FROM Product;

SELECT Product_no, SUM(Qty_order) AS Total_Quantity
FROM Product
GROUP BY Product_no;

SELECT Product_no, SUM(Qty_order) AS Total_Quantity
FROM Product
WHERE Product_no IN ('P00001', 'P00004')
GROUP BY Product_no;

-- Create emp_company table
```

Below the code, the 'Query Result' pane shows the output of the last query:

PRODUCT_NO	TOTAL_QUANTITY
1 P00001	14
2 P00004	9

The HR department has provided the following employee data to be inserted into the emp_company table:

1. Insert Values:

- Insert the above data into the emp_company table.

The screenshot shows the Oracle SQL Developer interface. On the left is the 'Connections' pane with a tree view of database objects for 'C23CS32'. The main window is titled 'C23CS32_PRACTICAL_6.sql' and contains a SQL worksheet with the following code:

```

INSERT INTO emp_company (ENAME, CNAME, SALARY)
VALUES ('Ajay', 'ACC', 8000);

INSERT INTO emp_company (ENAME, CNAME, SALARY)
VALUES ('Abhay', 'ACC', 1800);

-- Test Case 1: Display all rows from the emp_company table
SELECT * FROM emp_company;

-- Test Case 2: Maximum salary per company
SELECT CNAME, MAX(SALARY) AS Max_Salary
FROM emp_company
GROUP BY CNAME;

-- Test Case 3: Average salary per company
SELECT CNAME, AVG(SALARY) AS Avg_Salary
FROM emp_company

```

Below the worksheet, the 'Query Result' tab is active, displaying the results of the first query. It shows a table with 8 rows and 3 columns: ENAME, CNAME, and SALARY.

	ENAME	CNAME	SALARY
1	Anil	ACC	1500
2	Shankar	TATA	2000
3	Jay	WIPRO	1800
4	Sunil	WIPRO	1700
5	Vijay	TATA	5000
6	Prakash	TATA	3000
7	Ajay	ACC	8000
8	Abhay	ACC	1800

2. Maximum Salary per Company:

- List the company names and the maximum salary in each company .

The screenshot displays the Oracle SQL Developer interface. On the left, the 'Connections' pane shows a connection to 'C23CS32'. The main window is titled 'C23CS32_PRACTICAL_6.sql' and contains a SQL worksheet with the following queries:

```
INSERT INTO emp_company (ENAME, CNAME, SALARY)
VALUES ('Ajay', 'ACC', 8000);

INSERT INTO emp_company (ENAME, CNAME, SALARY)
VALUES ('Abhay', 'ACC', 1800);

-- Test Case 1: Display all rows from the emp_company table
SELECT * FROM emp_company;

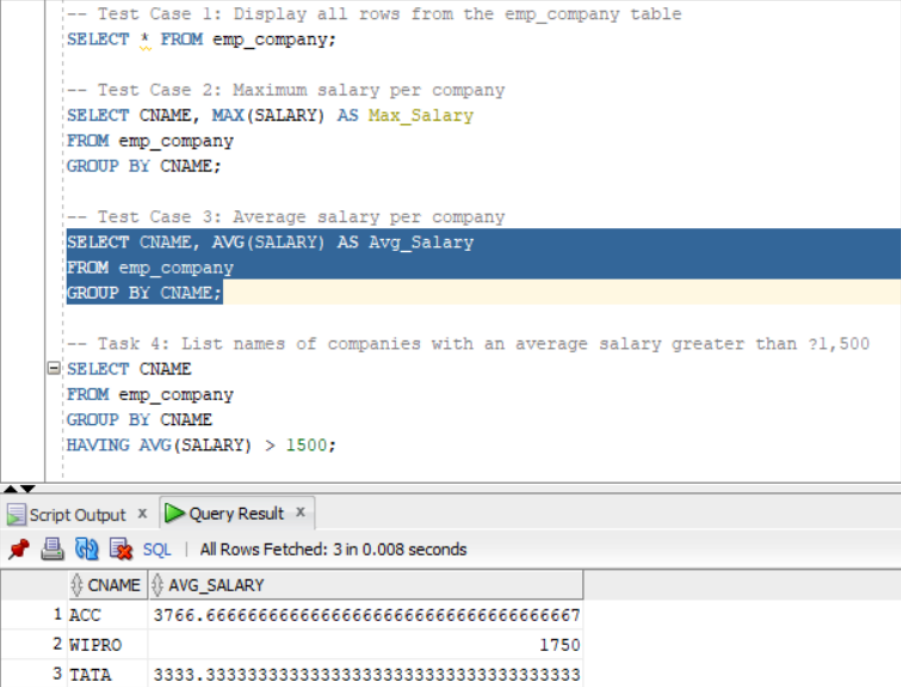
-- Test Case 2: Maximum salary per company
SELECT CNAME, MAX(SALARY) AS Max_Salary
FROM emp_company
GROUP BY CNAME;

-- Test Case 3: Average salary per company
SELECT CNAME, AVG(SALARY) AS Avg_Salary
FROM emp_company
```

The 'Query Result' pane at the bottom shows the results of the third query, displaying a table with columns 'CNAME' and 'MAX_SALARY'.

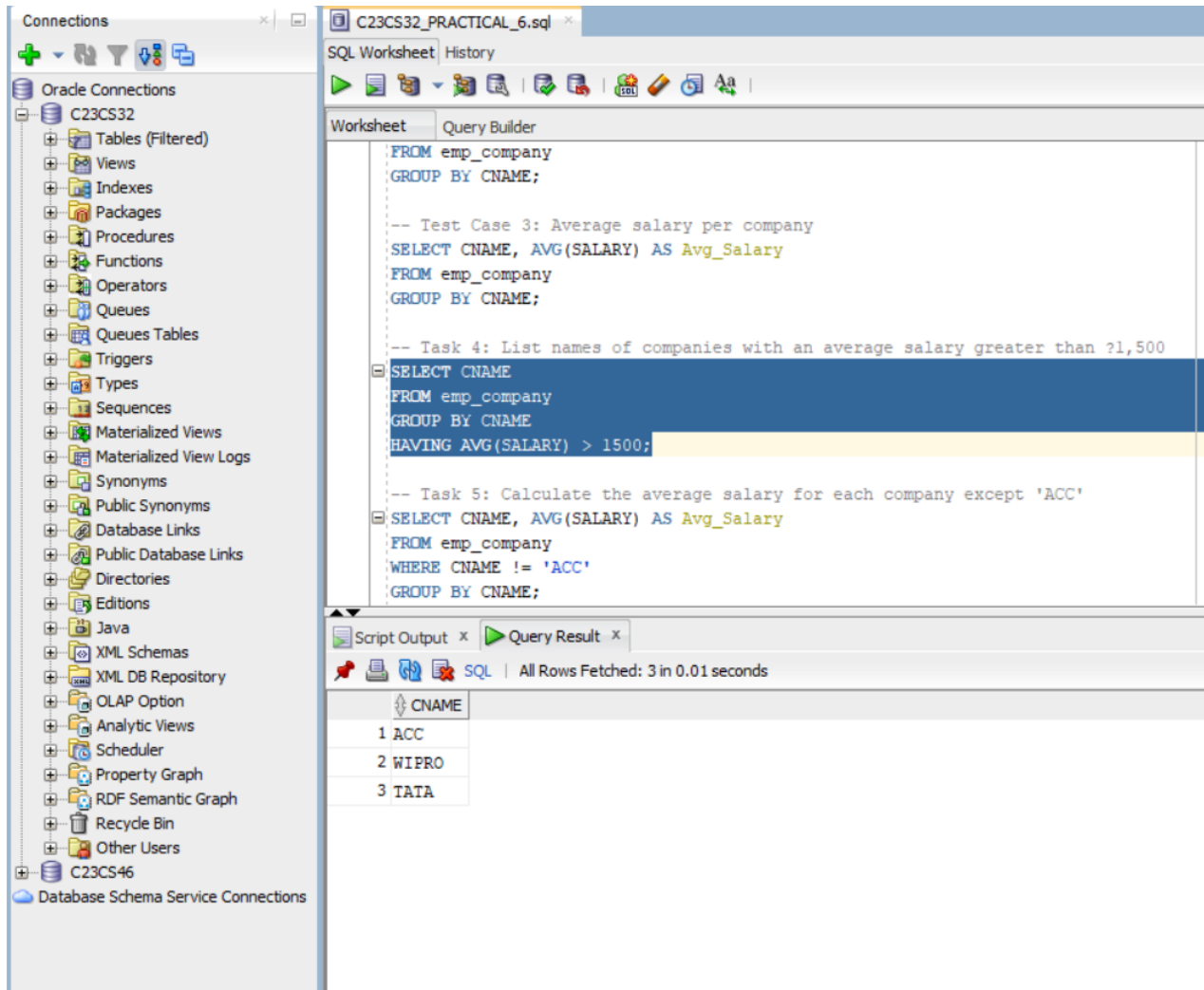
	CNAME	MAX_SALARY
1	ACC	8000
2	WIPRO	1800
3	TATA	5000

- Calculate the average salary for each company.



4. Filter Companies by Average Salary:

- List the names of companies with an average salary greater than ₹1,500.



The screenshot displays the Oracle SQL Developer interface. On the left, the 'Connections' pane shows the 'C23CS32' connection selected. The main window is titled 'C23CS32_PRACTICAL_6.sql' and contains the following SQL code:

```
FROM emp_company
GROUP BY CNAME;

-- Test Case 3: Average salary per company
SELECT CNAME, AVG(SALARY) AS Avg_Salary
FROM emp_company
GROUP BY CNAME;

-- Task 4: List names of companies with an average salary greater than ₹1,500
SELECT CNAME
FROM emp_company
GROUP BY CNAME
HAVING AVG(SALARY) > 1500;

-- Task 5: Calculate the average salary for each company except 'ACC'
SELECT CNAME, AVG(SALARY) AS Avg_Salary
FROM emp_company
WHERE CNAME != 'ACC'
GROUP BY CNAME;
```

The 'Query Result' pane at the bottom shows the output of the Task 4 query, which is a list of company names with an average salary greater than ₹1,500. The results are as follows:

CNAME
1 ACC
2 WIPRO
3 TATA

- Calculate the average salary for each company except ACC.

