



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 - 5

Aim: As a database administrator for a global bank, you are responsible for managing and analyzing employee and customer data stored in the bank's database. Your tasks involve using SQL functions to manipulate and retrieve critical information efficiently. These operations ensure seamless data communication and compliance with bank regulations.

Constraints

- **Not Null Constraints:** Critical fields like names and salaries must not be null.
- **Unique Constraints:** Ensure integrity of fields like Job_ID.
- **Check Constraints:** Validate positive salary values.

The bank maintains the following schemas:

1. JobProfile Table: Stores details of employees and their job roles.

- Emp_ID (Primary Key)
- Emp_Name (Not Null)
- Emp_Salary (Not Null, Check: Greater than zero)
- Job_ID (Unique)
- Department

2. Customer Table: Stores customer details.

- Cust_ID (Primary Key)
- Cust_Name (Not Null)

Tasks:-

The HR department wants to analyze employee salaries for better planning. Write SQL queries to:

1. Calculate the average salary of employees (with and without duplicates).

The screenshot shows the SQL Worksheet interface with the following content:

```

C:\CSE206\Practicals\SQL\JobProfile\JobProfile.sql

-- Inserting data into Customer table
INSERT INTO Customer VALUES (104, 'Vijay');
INSERT INTO Customer VALUES (105, 'Keyur');

-- Selecting all columns from JobProfile and Customer tables
SELECT * FROM JobProfile;
SELECT * FROM Customer;

-- 1. Average Salary Queries
SELECT AVG(Emp_Salary) AS Average_Salary FROM JobProfile;
SELECT AVG(DISTINCT Emp_Salary) AS Distinct_Average_Salary FROM JobProfile;

-- 2. Minimum Salary Query
SELECT MIN(Emp_Salary) AS Minimum_Salary FROM JobProfile;

-- 3. Count Employees and Distinct Departments
SELECT COUNT(*) AS Total_Employees, COUNT(DISTINCT Department) AS Distinct_Departments FROM JobProfile;

-- 4. Maximum Salary Query
SELECT MAX(Emp_Salary) AS Maximum_Salary FROM JobProfile;
  
```

Below the worksheet, the Query Result pane displays the output of the first query:

AVERAGE_SALARY
6100

The screenshot shows the SQL Worksheet interface with the following content:

```

C:\CSE206\Practicals\SQL\JobProfile\JobProfile.sql

-- Inserting data into Customer table
INSERT INTO Customer VALUES (104, 'Vijay');
INSERT INTO Customer VALUES (105, 'Keyur');

-- Selecting all columns from JobProfile and Customer tables
SELECT * FROM JobProfile;
SELECT * FROM Customer;

-- 1. Average Salary Queries
SELECT AVG(Emp_Salary) AS Average_Salary FROM JobProfile;
SELECT AVG(DISTINCT Emp_Salary) AS Distinct_Average_Salary FROM JobProfile;

-- 2. Minimum Salary Query
SELECT MIN(Emp_Salary) AS Minimum_Salary FROM JobProfile;

-- 3. Count Employees and Distinct Departments
SELECT COUNT(*) AS Total_Employees, COUNT(DISTINCT Department) AS Distinct_Departments FROM JobProfile;

-- 4. Maximum Salary Query
SELECT MAX(Emp_Salary) AS Maximum_Salary FROM JobProfile;
  
```

Below the worksheet, the Query Result pane displays the output of the second query:

DISTINCT_AVERAGE_SALARY
6100

2. Retrieve the minimum salary from the JobProfile table.

The screenshot shows the SQL Worksheet interface with the following code:

```

INSERT INTO Customer VALUES (104, 'Vijay');
INSERT INTO Customer VALUES (105, 'Keyur');

SELECT * FROM JobProfile;
SELECT * FROM Customer;

-- 1. Average Salary Queries
SELECT AVG(Emp_Salary) AS Average_Salary FROM JobProfile;
SELECT AVG(DISTINCT Emp_Salary) AS Distinct_Average_Salary FROM JobProfile;

-- 2. Minimum Salary Query
SELECT MIN(Emp_Salary) AS Minimum_Salary FROM JobProfile;

-- 3. Count Employees and Distinct Departments
SELECT COUNT(*) AS Total_Employees, COUNT(DISTINCT Department) AS Distinct_Departments FROM JobProfile;

-- 4. Maximum Salary Query
SELECT MAX(Emp_Salary) AS Maximum_Salary FROM JobProfile;

```

The query `SELECT MIN(Emp_Salary) AS Minimum_Salary FROM JobProfile;` is highlighted in yellow. The results pane shows the output:

MINIMUM_SALARY	
1	3000

All Rows Fetched: 1 in 0.019 seconds

3. Count the total number of employees and distinct departments.

The screenshot shows the SQL Worksheet interface with the following code:

```

INSERT INTO Customer VALUES (104, 'Vijay');
INSERT INTO Customer VALUES (105, 'Keyur');

SELECT * FROM JobProfile;
SELECT * FROM Customer;

-- 1. Average Salary Queries
SELECT AVG(Emp_Salary) AS Average_Salary FROM JobProfile;
SELECT AVG(DISTINCT Emp_Salary) AS Distinct_Average_Salary FROM JobProfile;

-- 2. Minimum Salary Query
SELECT MIN(Emp_Salary) AS Minimum_Salary FROM JobProfile;

-- 3. Count Employees and Distinct Departments
SELECT COUNT(*) AS Total_Employees, COUNT(DISTINCT Department) AS Distinct_Departments FROM JobProfile;

-- 4. Maximum Salary Query
SELECT MAX(Emp_Salary) AS Maximum_Salary FROM JobProfile;

```

The query `SELECT COUNT(*) AS Total_Employees, COUNT(DISTINCT Department) AS Distinct_Departments FROM JobProfile;` is highlighted in yellow. The results pane shows the output:

TOTAL_EMPLOYEES	DISTINCT_DEPARTMENTS
1	5
	4

All Rows Fetched: 1 in 0.016 seconds

4. Retrieve the maximum salary from the JobProfile table.

The screenshot shows the SQL Worksheet interface with the following code:

```

SELECT * FROM Customer;

-- 1. Average Salary Queries
SELECT AVG(Emp_Salary) AS Average_Salary FROM JobProfile;
SELECT AVG(DISTINCT Emp_Salary) AS Distinct_Average_Salary FROM JobProfile;

-- 2. Minimum Salary Query
SELECT MIN(Emp_Salary) AS Minimum_Salary FROM JobProfile;

-- 3. Count Employees and Distinct Departments
SELECT COUNT(*) AS Total_Employees, COUNT(DISTINCT Department) AS Distinct_Departments FROM JobProfile;

-- 4. Maximum Salary Query
SELECT MAX(Emp_Salary) AS Maximum_Salary FROM JobProfile;

-- 5. Total and Distinct Sum of Salaries
SELECT SUM(Emp_Salary) AS Total_Salaries FROM JobProfile;
SELECT SUM(DISTINCT Emp_Salary) AS Distinct_Salaries FROM JobProfile;

```

The result set for the maximum salary query is:

	MAXIMUM_SALARY
1	9000

5. Calculate the total and distinct sum of all salaries.

The screenshot shows the SQL Worksheet interface with the following code:

```

SELECT * FROM Customer;

-- 1. Average Salary Queries
SELECT AVG(Emp_Salary) AS Average_Salary FROM JobProfile;
SELECT AVG(DISTINCT Emp_Salary) AS Distinct_Average_Salary FROM JobProfile;

-- 2. Minimum Salary Query
SELECT MIN(Emp_Salary) AS Minimum_Salary FROM JobProfile;

-- 3. Count Employees and Distinct Departments
SELECT COUNT(*) AS Total_Employees, COUNT(DISTINCT Department) AS Distinct_Departments FROM JobProfile;

-- 4. Maximum Salary Query
SELECT MAX(Emp_Salary) AS Maximum_Salary FROM JobProfile;

-- 5. Total and Distinct Sum of Salaries
SELECT SUM(Emp_Salary) AS Total_Salaries FROM JobProfile;
SELECT SUM(DISTINCT Emp_Salary) AS Distinct_Salaries FROM JobProfile;

```

The result set for the total and distinct sum of salaries query is:

	TOTAL_SALARIES
1	30500

The screenshot shows a SQL Worksheet window titled 'C23CS32_PRACTICAL_5.sql'. The worksheet contains five numbered SQL queries for calculating salaries:

```

SELECT * FROM Customer;

-- 1. Average Salary Queries
SELECT AVG(Emp_Salary) AS Average_Salary FROM JobProfile;
SELECT AVG(DISTINCT Emp_Salary) AS Distinct_Average_Salary FROM JobProfile;

-- 2. Minimum Salary Query
SELECT MIN(Emp_Salary) AS Minimum_Salary FROM JobProfile;

-- 3. Count Employees and Distinct Departments
SELECT COUNT(*) AS Total_Employees, COUNT(DISTINCT Department) AS Distinct_Departments FROM JobProfile;

-- 4. Maximum Salary Query
SELECT MAX(Emp_Salary) AS Maximum_Salary FROM JobProfile;

-- 5. Total and Distinct Sum of Salaries
SELECT SUM(Emp_Salary) AS Total_Salaries FROM JobProfile;
SELECT SUM(DISTINCT Emp_Salary) AS Distinct_Salaries FROM JobProfile;

```

Below the worksheet is a 'Query Result' tab showing the output for the fifth query:

	DISTINCT_SALARIES
1	30500

The finance team needs specific salary calculations for tax and benefits. Write SQL queries to:

1. Calculate the absolute difference between each employee's salary and ₹1,000.

The screenshot shows a SQL Worksheet window titled 'C23CS32_PRACTICAL_5.sql'. The worksheet contains several numbered SQL queries for salary calculations, including the one from the previous section and additional ones for absolute difference, salary squared, rounded salaries, and square root of salaries.

```

-- 4. Maximum Salary Query
SELECT MAX(Emp_Salary) AS Maximum_Salary FROM JobProfile;

-- 5. Total and Distinct Sum of Salaries
SELECT SUM(Emp_Salary) AS Total_Salaries FROM JobProfile;
SELECT SUM(DISTINCT Emp_Salary) AS Distinct_Salaries FROM JobProfile;

-- -----
-- 1. Absolute Difference from 1000
SELECT Emp_ID, Emp_Name, ABS(Emp_Salary - 1000) AS Salary_Difference FROM JobProfile;

-- 2. Salary Squared
SELECT Emp_ID, Emp_Name, POWER(Emp_Salary, 2) AS Salary_Squared FROM JobProfile;

-- 3. Round Salaries to Two Decimal Places
SELECT Emp_ID, Emp_Name, ROUND(Emp_Salary, 2) AS Rounded_Salary FROM JobProfile;

-- 4. Square Root of Salaries

```

Below the worksheet is a 'Query Result' tab showing the output for the first query:

EMP_ID	EMP_NAME	SALARY_DIFFERENCE
1	1 John	4000
2	2 Alice	6500
3	3 Bob	2000
4	4 Eve	8000
5	5 Charlie	5000

2. Compute the square of each employee's salary.

The screenshot shows a SQL Worksheet window titled 'C23CS32_PRACTICAL_5.sql'. The worksheet contains several SQL statements:

```
-- 4. Maximum Salary Query  
SELECT MAX(Emp_Salary) AS Maximum_Salary FROM JobProfile;  
  
-- 5. Total and Distinct Sum of Salaries  
SELECT SUM(Emp_Salary) AS Total_Salaries FROM JobProfile;  
SELECT SUM(DISTINCT Emp_Salary) AS Distinct_Salaries FROM JobProfile;  
  
--  
  
-- 1. Absolute Difference from 1000  
SELECT Emp_ID, Emp_Name, ABS(Emp_Salary - 1000) AS Salary_Difference FROM JobProfile;  
  
-- 2. Salary Squared  
SELECT Emp_ID, Emp_Name, POWER(Emp_Salary, 2) AS Salary_Squared FROM JobProfile;  
  
-- 3. Round Salaries to Two Decimal Places  
SELECT Emp_ID, Emp_Name, ROUND(Emp_Salary, 2) AS Rounded_Salary FROM JobProfile;  
  
-- 4. Square Root of Salaries
```

The 'Query Result' tab is selected, showing the output of the last query:

EMP_ID	EMP_NAME	SALARY_SQUARED
1	John	25000000
2	Alice	56250000
3	Bob	9000000
4	Eve	81000000
5	Charlie	36000000

3. Round salaries to two decimal places.

The screenshot shows the SQL Worksheet interface in SSMS. The top bar displays the title 'C23CS32_PRACTICAL_5.sql'. The main area contains several SQL statements under comments:

```
-- 4. Maximum Salary Query  
SELECT MAX(Emp_Salary) AS Maximum_Salary FROM JobProfile;  
  
-- 5. Total and Distinct Sum of Salaries  
SELECT SUM(Emp_Salary) AS Total_Salaries FROM JobProfile;  
SELECT SUM(DISTINCT Emp_Salary) AS Distinct_Salaries FROM JobProfile;  
  
--  
  
-- 1. Absolute Difference from 1000  
SELECT Emp_ID, Emp_Name, ABS(Emp_Salary - 1000) AS Salary_Difference FROM JobProfile;  
  
-- 2. Salary Squared  
SELECT Emp_ID, Emp_Name, POWER(Emp_Salary, 2) AS Salary_Squared FROM JobProfile;  
  
-- 3. Round Salaries to Two Decimal Places  
SELECT Emp_ID, Emp_Name, ROUND(Emp_Salary, 2) AS Rounded_Salary FROM JobProfile;  
  
-- 4. Square Root of Salaries
```

The bottom section shows the 'Query Result' tab with the output of the last query:

EMP_ID	EMP_NAME	ROUNDED_SALARY
1	John	5000
2	Alice	7500
3	Bob	3000
4	Eve	9000
5	Charlie	6000

4. Find the square root of salaries.

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections sidebar with two entries: C23CS32 and C23CS46. The main area is a worksheet titled 'C23CS32_PRACTICAL_5.sql' containing the following SQL code:

```

SELECT SUM(Emp_Salary) AS Total_Salaries FROM JobProfile;
SELECT SUM(DISTINCT Emp_Salary) AS Distinct_Salaries FROM JobProfile;

-- 1. Absolute Difference from 1000
SELECT Emp_ID, Emp_Name, ABS(Emp_Salary - 1000) AS Salary_Difference FROM JobProfile;

-- 2. Salary Squared
SELECT Emp_ID, Emp_Name, POWER(Emp_Salary, 2) AS Salary_Squared FROM JobProfile;

-- 3. Round Salaries to Two Decimal Places
SELECT Emp_ID, Emp_Name, ROUND(Emp_Salary, 2) AS Rounded_Salary FROM JobProfile;

-- 4. Square Root of Salaries
SELECT Emp_ID, Emp_Name, SQRT(Emp_Salary) AS Salary_Square_Root FROM JobProfile;

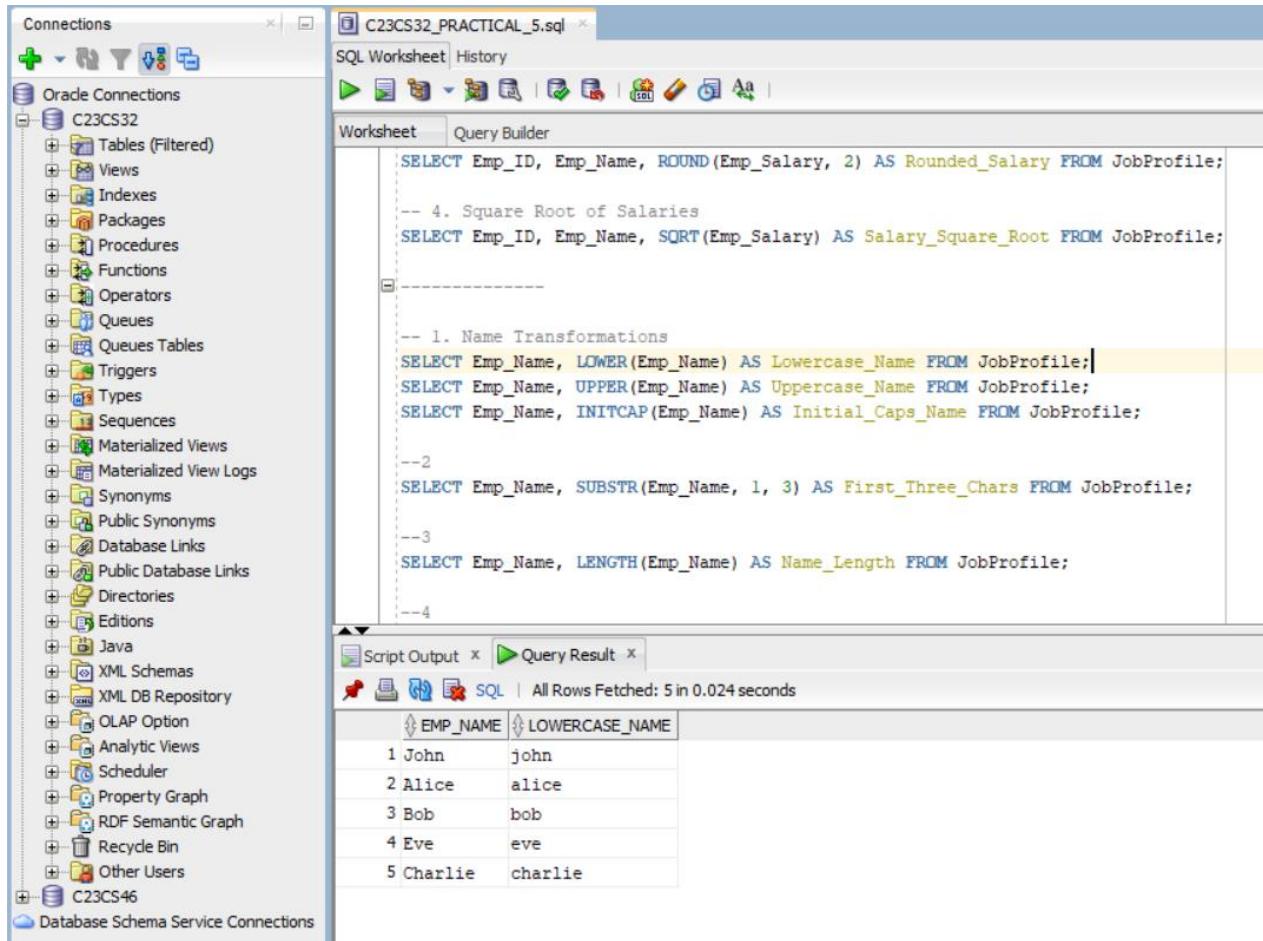
```

Below the worksheet is a 'Query Result' tab showing the output of the last query:

EMP_ID	EMP_NAME	SALARY_SQUARE_ROOT
1	John	70.71067811865475244008443621048490392848
2	Alice	86.60254037844386467637231707529361834714
3	Bob	54.77225575051661134569697828008021339527
4	Eve	94.8683298050513799599680633298155601158
5	Charlie	77.45966692414833770358530799564799221665

To ensure uniformity in names across systems, perform the following:

- Convert all employee first names to lowercase, uppercase, and initial caps.



The screenshot shows the Oracle SQL Developer interface. On the left is the Connections sidebar with 'Oracle Connections' expanded, showing 'C23CS32' and its various schema objects like Tables, Views, Indexes, etc. The main area is a 'SQL Worksheet' titled 'C23CS32_PRACTICAL_5.sql'. It contains the following SQL code:

```

SELECT Emp_ID, Emp_Name, ROUND(Emp_Salary, 2) AS Rounded_Salary FROM JobProfile;

-- 4. Square Root of Salaries
SELECT Emp_ID, Emp_Name, SQRT(Emp_Salary) AS Salary_Square_Root FROM JobProfile;

-- -----
-- 1. Name Transformations
SELECT Emp_Name, LOWER(Emp_Name) AS Lowercase_Name FROM JobProfile;
SELECT Emp_Name, UPPER(Emp_Name) AS Uppercase_Name FROM JobProfile;
SELECT Emp_Name, INITCAP(Emp_Name) AS Initial_Caps_Name FROM JobProfile;

--2
SELECT Emp_Name, SUBSTR(Emp_Name, 1, 3) AS First_Three_Chars FROM JobProfile;

--3
SELECT Emp_Name, LENGTH(Emp_Name) AS Name_Length FROM JobProfile;

--4

```

Below the worksheet is a 'Script Output' tab showing the results of the first query:

EMP_NAME	LOWERCASE_NAME
1 John	john
2 Alice	alice
3 Bob	bob
4 Eve	eve
5 Charlie	charlie

The status bar at the bottom right indicates 'All Rows Fetched: 5 in 0.024 seconds'.

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, which includes Oracle Connections (C23CS32 and C23CS46), Database Schema Service Connections, and various database objects like Tables, Views, Indexes, etc. The main area is a SQL Worksheet titled 'C23CS32_PRACTICAL_5.sql'. It contains several SQL statements demonstrating string manipulation functions:

```

SELECT Emp_ID, Emp_Name, ROUND(Emp_Salary, 2) AS Rounded_Salary FROM JobProfile;

-- 4. Square Root of Salaries
SELECT Emp_ID, Emp_Name, SQRT(Emp_Salary) AS Salary_Square_Root FROM JobProfile;

-----

-- 1. Name Transformations
SELECT Emp_Name, LOWER(Emp_Name) AS Lowercase_Name FROM JobProfile;
SELECT Emp_Name, UPPER(Emp_Name) AS Uppercase_Name FROM JobProfile;
SELECT Emp_Name, INITCAP(Emp_Name) AS Initial_Caps_Name FROM JobProfile;

--2
SELECT Emp_Name, SUBSTR(Emp_Name, 1, 3) AS First_Three_Chars FROM JobProfile;

--3
SELECT Emp_Name, LENGTH(Emp_Name) AS Name_Length FROM JobProfile;

--4

```

Below the worksheet is a 'Script Output' tab showing the results of the last query:

EMP_NAME	UPPERCASE_NAME
1 John	JOHN
2 Alice	ALICE
3 Bob	BOB
4 Eve	EVE
5 Charlie	CHARLIE

This screenshot is nearly identical to the one above, showing the same Oracle SQL Developer interface and connections. The SQL Worksheet 'C23CS32_PRACTICAL_5.sql' contains the same set of SQL statements for name transformations. The 'Script Output' tab shows the results of the last query:

EMP_NAME	INITIAL_CAPS_NAME
1 John	John
2 Alice	Alice
3 Bob	Bob
4 Eve	Eve
5 Charlie	Charlie

2. Extract the first three characters of employee first names.

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, showing Oracle Connections like C23CS32 and C23CS46. The central area is the SQL Worksheet titled 'C23CS32_PRACTICAL_5.sql'. It contains the following SQL code:

```

SELECT Emp_ID, Emp_Name, ROUND(Emp_Salary, 2) AS Rounded_Salary FROM JobProfile;

-- 4. Square Root of Salaries
SELECT Emp_ID, Emp_Name, SQRT(Emp_Salary) AS Salary_Square_Root FROM JobProfile;

-- -----
-- 1. Name Transformations
SELECT Emp_Name, LOWER(Emp_Name) AS Lowercase_Name FROM JobProfile;
SELECT Emp_Name, UPPER(Emp_Name) AS Uppercase_Name FROM JobProfile;
SELECT Emp_Name, INITCAP(Emp_Name) AS Initial_Caps_Name FROM JobProfile;

-- 2
SELECT Emp_Name, SUBSTR(Emp_Name, 1, 3) AS First_Three_Chars FROM JobProfile;

-- 3
SELECT Emp_Name, LENGTH(Emp_Name) AS Name_Length FROM JobProfile;

-- 4

```

Below the worksheet is the Script Output window, which displays the results of the query for the first three characters:

EMP_NAME	FIRST_THREE_CHARS
1 John	Joh
2 Alice	Ali
3 Bob	Bob
4 Eve	Eve
5 Charlie	Cha

3. Find the length of each employee's first name.

The screenshot shows the Oracle SQL Developer interface, identical to the previous one but with a different query selected in the worksheet. The SQL Worksheet now contains:

```

SELECT Emp_ID, Emp_Name, ROUND(Emp_Salary, 2) AS Rounded_Salary FROM JobProfile;

-- 4. Square Root of Salaries
SELECT Emp_ID, Emp_Name, SQRT(Emp_Salary) AS Salary_Square_Root FROM JobProfile;

-- -----
-- 1. Name Transformations
SELECT Emp_Name, LOWER(Emp_Name) AS Lowercase_Name FROM JobProfile;
SELECT Emp_Name, UPPER(Emp_Name) AS Uppercase_Name FROM JobProfile;
SELECT Emp_Name, INITCAP(Emp_Name) AS Initial_Caps_Name FROM JobProfile;

-- 2
SELECT Emp_Name, SUBSTR(Emp_Name, 1, 3) AS First_Three_Chars FROM JobProfile;

-- 3
SELECT Emp_Name, LENGTH(Emp_Name) AS Name_Length FROM JobProfile;

-- 4

```

The Script Output window shows the results of the LENGTH query:

EMP_NAME	NAME_LENGTH
1 John	4
2 Alice	5
3 Bob	3
4 Eve	3
5 Charlie	7

4. Remove leading 'A' and trailing 'a' from employee first names.

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, showing Oracle Connections like C23CS32 and C23CS46. The main area is a worksheet titled 'C23CS32_PRACTICAL_5.sql'. It contains the following SQL code:

```
SELECT Emp_Name, LENGTH(Emp_Name) AS Name_Length FROM JobProfile;  
--4  
SELECT Emp_Name, LTRIM(Emp_Name, 'A') AS Leading_Trimmed_Name, RTRIM(Emp_Name, 'a')  
AS Trailing_Trimmed_Name FROM JobProfile;  
--5  
SELECT Emp_Name, LPAD(Emp_Name, 10, '') AS Left_Padded, RPAD(Emp_Name, 10, '')  
AS Right_Padded FROM JobProfile;  
-----  
-- 1. Data Type Conversions  
SELECT TO_NUMBER('10000.50') AS Numeric_Conversion FROM DUAL;  
--2  
SELECT TO_CHAR(5000, '999G999D99') AS Formatted_Salary FROM DUAL;
```

Below the worksheet is a 'Script Output' tab showing the results of the queries. The 'Query Result' tab shows the output of the second query:

EMP_NAME	LEADING_TRIMMED_NAME	TRAILING_TRIMMED_NAME
1 John	John	John
2 Alice	lice	Alice
3 Bob	Bob	Bob
4 Eve	Eve	Eve
5 Charlie	Charlie	Charlie

5. Pad employee first names with '*' on the left and right, ensuring a total length of 10.

The screenshot shows the Oracle SQL Developer interface. The left pane displays the database connections and schema for 'C23CS32'. The central 'Worksheet' pane contains the following SQL code:

```

SELECT Emp_Name, LENGTH(Emp_Name) AS Name_Length FROM JobProfile;

--4
SELECT Emp_Name, LTRIM(Emp_Name, 'A') AS Leading_Trimmed_Name, RTRIM(Emp_Name, 'a')
AS Trailing_Trimmed_Name FROM JobProfile;

--5
SELECT Emp_Name, LPAD(Emp_Name, 10, '*') AS Left_Padded, RPAD(Emp_Name, 10, '*')
AS Right_Padded FROM JobProfile;

-----1. Data Type Conversions
SELECT TO_NUMBER('10000.50') AS Numeric_Conversion FROM DUAL;

-----2
SELECT TO_CHAR(5000, '999G999D99') AS Formatted_Salary FROM DUAL;

```

The 'Script Output' pane at the bottom shows the results of the padding queries:

EMP_NAME	LEFT_PADDED	RIGHT_PADDED
1 John	*****John	John*****
2 Alice	*****Alice	Alice*****
3 Bob	*****Bob	Bob*****
4 Eve	*****Eve	Eve*****
5 Charlie	***Charlie	Charlie***

The data migration team requires conversions between different data types:

1. Convert a string representation of a salary to a numeric format.

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections sidebar with various database objects listed under Oracle Connections. The main area is a SQL Worksheet titled 'C23CS32_PRACTICAL_5.sql'. The worksheet contains the following SQL code:

```
SELECT Emp_Name, LENGTH(Emp_Name) AS Name_Length FROM JobProfile;
--4
SELECT Emp_Name, LTRIM(Emp_Name, 'A') AS Leading_Trimmed_Name, RTRIM(Emp_Name, 'a')
AS Trailing_Trimmed_Name FROM JobProfile;
--5
SELECT Emp_Name, LPAD(Emp_Name, 10, '**') AS Left_Padded, RPAD(Emp_Name, 10, '**')
AS Right_Padded FROM JobProfile;
-----1. Data Type Conversions
SELECT TO_NUMBER('10000.50') AS Numeric_Conversion FROM DUAL;
--2
SELECT TO_CHAR(5000, '999G999D99') AS Formatted_Salary FROM DUAL;
```

Below the worksheet is a Script Output window showing the result of the first query:

NUMERIC_CONVERSION
1 10000.5

The output indicates "All Rows Fetched: 1 in 0.016 seconds".

2. Format a numeric salary value into a string with specific formatting.

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections sidebar with various database connection nodes. The main area has a tab titled 'C23CS32_PRACTICAL_5.sql' which contains the following SQL code:

```
SELECT Emp_Name, LENGTH(Emp_Name) AS Name_Length FROM JobProfile;  
--4  
SELECT Emp_Name, LTRIM(Emp_Name, 'A') AS Leading_Trimmed_Name, RTRIM(Emp_Name, 'a')  
AS Trailing_Trimmed_Name FROM JobProfile;  
--5  
SELECT Emp_Name, LPAD(Emp_Name, 10, '**') AS Left_Padded, RPAD(Emp_Name, 10, '**')  
AS Right_Padded FROM JobProfile;  
--  
-- 1. Data Type Conversions  
SELECT TO_NUMBER('10000.50') AS Numeric_Conversion FROM DUAL;  
--2  
SELECT TO_CHAR(5000, '999G999D99') AS Formatted_Salary FROM DUAL;
```

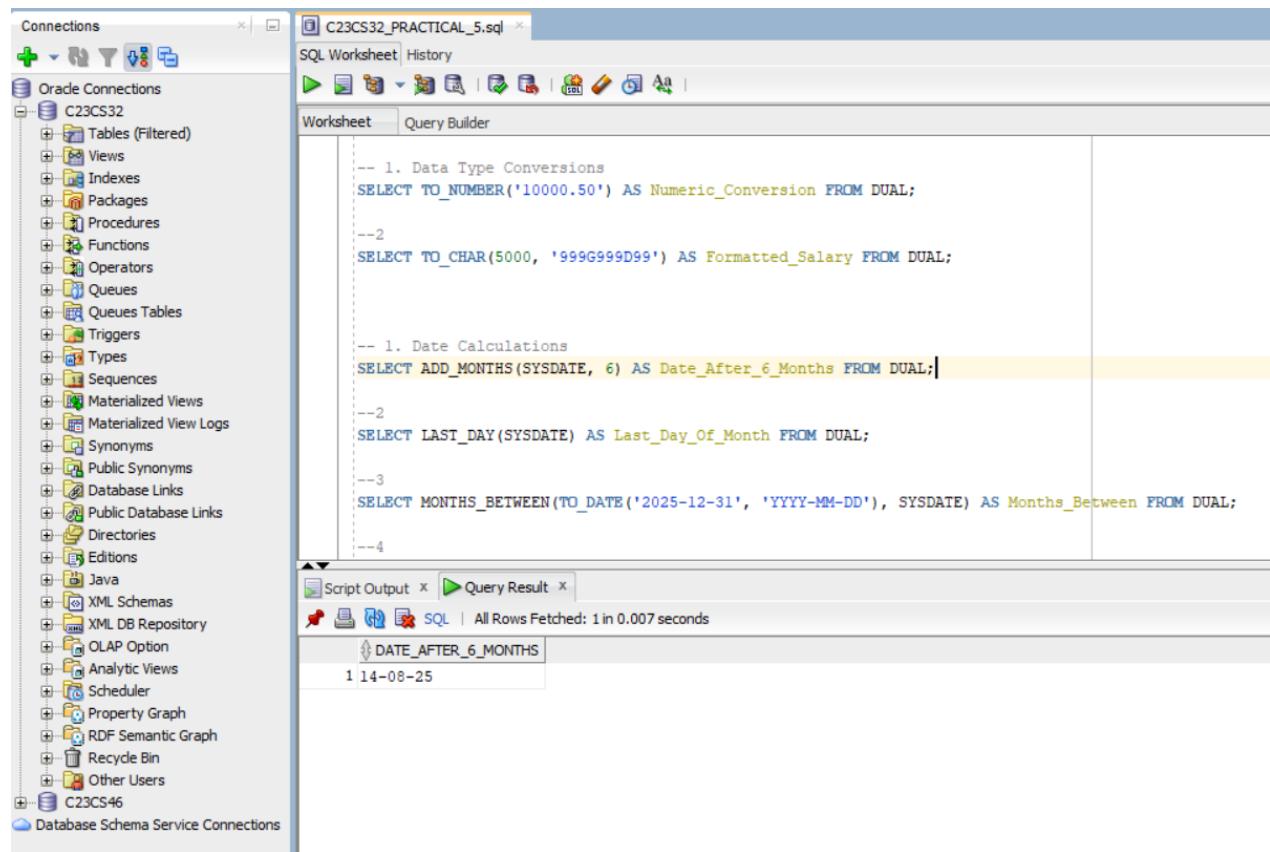
Below the worksheet is a 'Script Output' window showing the results of the last query:

	FORMATTED_SALARY
1	5,000.00

The status bar at the bottom indicates "All Rows Fetched: 1 in 0.006 seconds".

To assist in employee scheduling, perform the following:

1. Calculate the date after adding 6 months to the current date.



The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, with the C23CS32 connection selected. The central area is the Worksheet tab, containing the following SQL code:

```

-- 1. Data Type Conversions
SELECT TO_NUMBER('10000.50') AS Numeric_Conversion FROM DUAL;

--2
SELECT TO_CHAR(5000, '999G999D99') AS Formatted_Salary FROM DUAL;

-- 1. Date Calculations
SELECT ADD_MONTHS(SYSDATE, 6) AS Date_After_6_Months FROM DUAL;

--2
SELECT LAST_DAY(SYSDATE) AS Last_Day_Of_Month FROM DUAL;

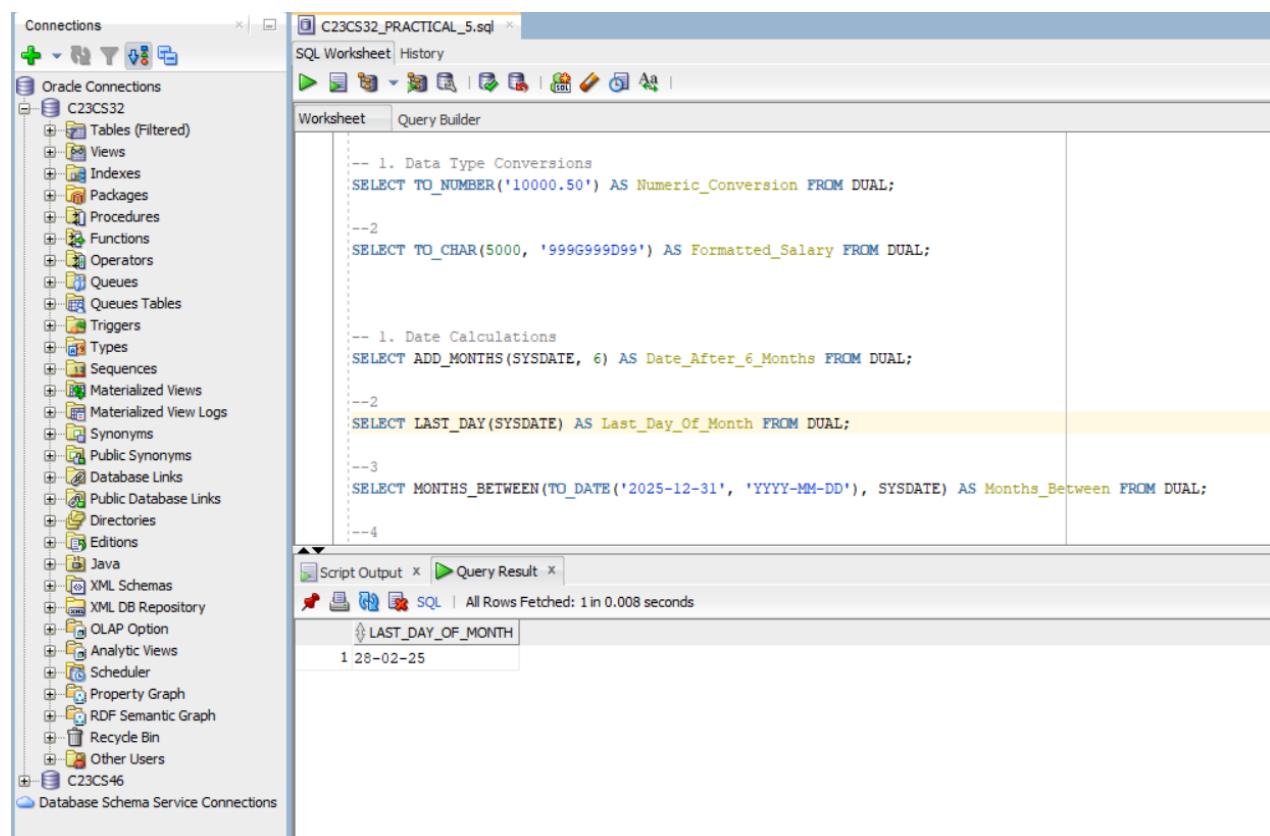
--3
SELECT MONTHS_BETWEEN(TO_DATE('2025-12-31', 'YYYY-MM-DD'), SYSDATE) AS Months_Between FROM DUAL;

```

The third query, `SELECT ADD_MONTHS(SYSDATE, 6) AS Date_After_6_Months FROM DUAL;`, is highlighted. Below the worksheet is the Script Output tab, which shows the result:

DATE_AFTER_6_MONTHS
1 14-08-25

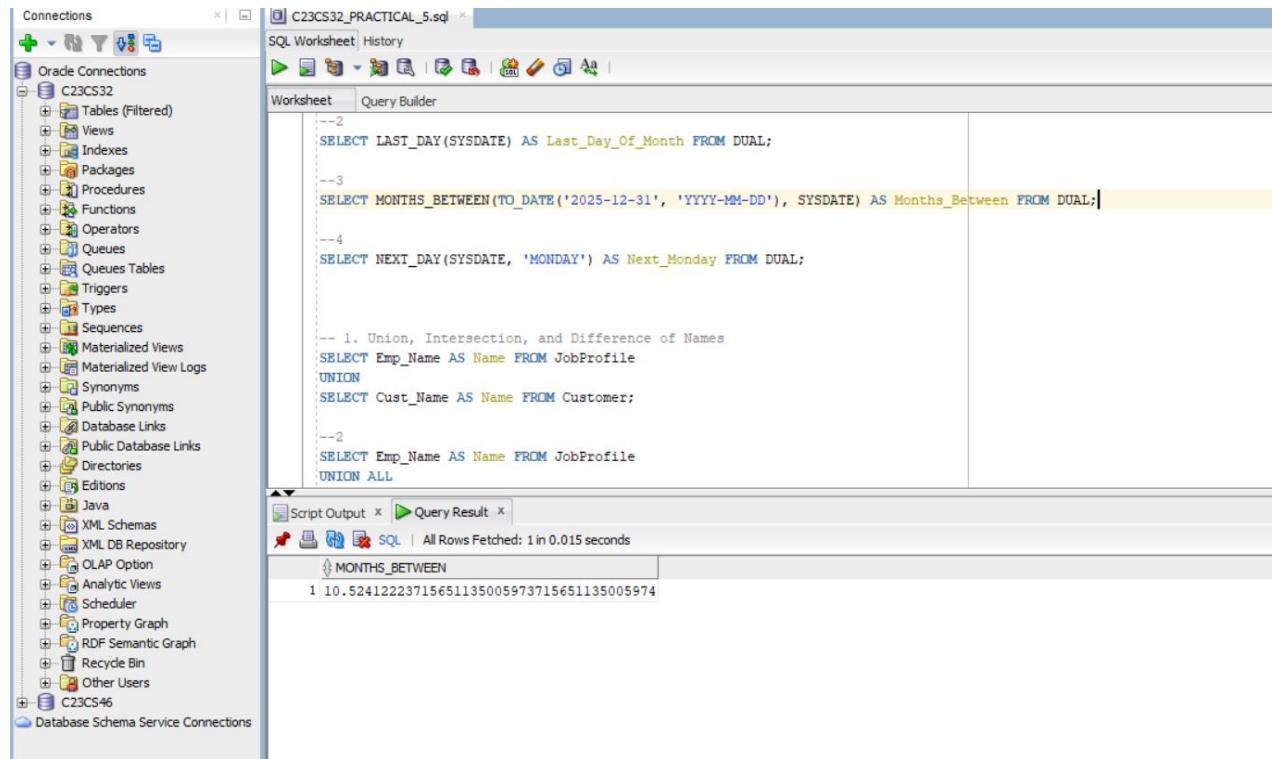
2. Retrieve the last day of the current month.



The screenshot shows the Oracle SQL Developer interface, identical to the previous one but with a different query selected in the Worksheet tab. The third query, `SELECT LAST_DAY(SYSDATE) AS Last_Day_Of_Month FROM DUAL;`, is highlighted. Below the worksheet is the Script Output tab, which shows the result:

LAST_DAY_OF_MONTH
1 28-02-25

3. Calculate the number of months between two dates.



```
--2
SELECT LAST_DAY(SYSDATE) AS Last_Day_Of_Month FROM DUAL;

--3
SELECT MONTHS_BETWEEN(TO_DATE('2025-12-31', 'YYYY-MM-DD'), SYSDATE) AS Months_Between FROM DUAL;

--4
SELECT NEXT_DAY(SYSDATE, 'MONDAY') AS Next_Monday FROM DUAL;

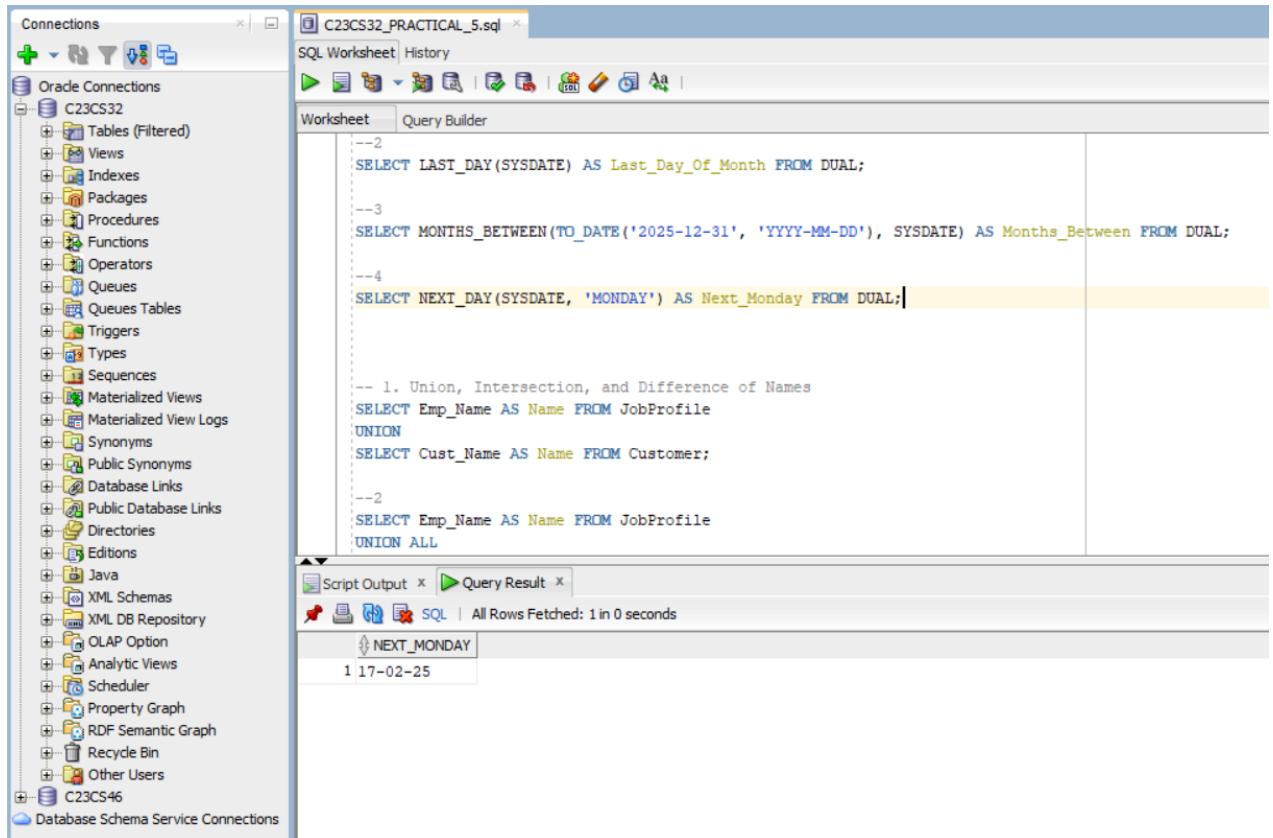
-- 1. Union, Intersection, and Difference of Names
SELECT Emp_Name AS Name FROM JobProfile
UNION
SELECT Cust_Name AS Name FROM Customer;

--2
SELECT Emp_Name AS Name FROM JobProfile
UNION ALL
```

Script Output | Query Result | SQL | All Rows Fetched: 1 in 0.015 seconds

MONTHS_BETWEEN
1 10.52412223715651135005973715651135005974

4. Find the next Monday from the current date.



```
--2
SELECT LAST_DAY(SYSDATE) AS Last_Day_Of_Month FROM DUAL;

--3
SELECT MONTHS_BETWEEN(TO_DATE('2025-12-31', 'YYYY-MM-DD'), SYSDATE) AS Months_Between FROM DUAL;

--4
SELECT NEXT_DAY(SYSDATE, 'MONDAY') AS Next_Monday FROM DUAL;

-- 1. Union, Intersection, and Difference of Names
SELECT Emp_Name AS Name FROM JobProfile
UNION
SELECT Cust_Name AS Name FROM Customer;

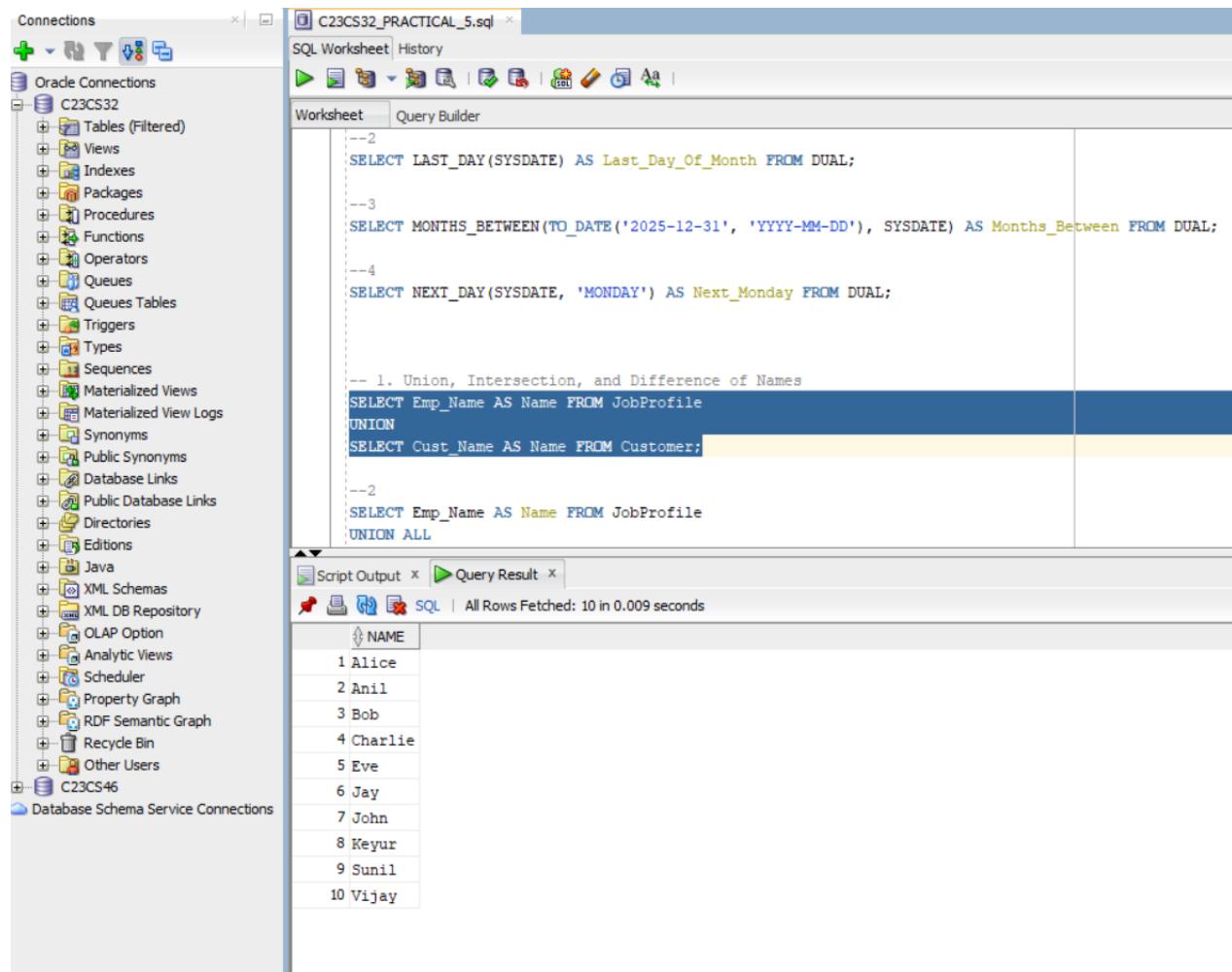
--2
SELECT Emp_Name AS Name FROM JobProfile
UNION ALL
```

Script Output | Query Result | SQL | All Rows Fetched: 1 in 0 seconds

NEXT_MONDAY
1 17-02-25

To identify overlaps and differences between employees and customers, write SQL queries to:

1. Retrieve the union of first names from employees and customers.



The screenshot shows the Oracle SQL Developer interface. On the left is the Object Navigator pane, which lists various database objects under the connection C23CS32. In the center is the SQL Worksheet pane, titled 'C23CS32_PRACTICAL_5.sql'. The worksheet contains several SQL statements. The first four statements are comments and SELECT statements related to dates. The fifth statement is a UNION query combining employee names from 'JobProfile' and customer names from 'Customer'. The sixth statement is another UNION query combining employee names from 'JobProfile'. Below the worksheet is the Script Output pane, which displays the results of the queries. The results show ten names: Alice, Anil, Bob, Charlie, Eve, Jay, John, Keyur, Sunil, and Vijay.

```

--2
SELECT LAST_DAY(SYSDATE) AS Last_Day_Of_Month FROM DUAL;

--3
SELECT MONTHS_BETWEEN(TO_DATE('2025-12-31', 'YYYY-MM-DD'), SYSDATE) AS Months_Between FROM DUAL;

--4
SELECT NEXT_DAY(SYSDATE, 'MONDAY') AS Next_Monday FROM DUAL;

-- 1. Union, Intersection, and Difference of Names
SELECT Emp_Name AS Name FROM JobProfile
UNION
SELECT Cust_Name AS Name FROM Customer;

--2
SELECT Emp_Name AS Name FROM JobProfile
UNION ALL

```

NAME
1 Alice
2 Anil
3 Bob
4 Charlie
5 Eve
6 Jay
7 John
8 Keyur
9 Sunil
10 Vijay

2. Retrieve the union of first names (including duplicates).

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections sidebar with several database connections listed. The main area is a worksheet titled 'C23CS32_PRACTICAL_5.sql' containing four numbered SQL queries:

```
SELECT Emp_Name AS Name FROM JobProfile
UNION
SELECT Cust_Name AS Name FROM Customer;

--2
SELECT Emp_Name AS Name FROM JobProfile
UNION ALL
SELECT Cust_Name AS Name FROM Customer;

--3
SELECT Emp_Name AS Name FROM JobProfile
INTERSECT
SELECT Cust_Name AS Name FROM Customer;

--4
SELECT Emp_Name AS Name FROM JobProfile
MINUS
SELECT Cust_Name AS Name FROM Customer;
```

Below the worksheet is a 'Script Output' tab showing the execution status: 'All Rows Fetched: 10 in 0.012 seconds'. The 'Query Result' tab displays the output as a table:

NAME
1 John
2 Alice
3 Bob
4 Eve
5 Charlie
6 Anil
7 Sunil
8 Jay
9 Vijay
10 Keyur

3. Find the intersection of first names from employees and customers.

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections sidebar with 'C23CS32' selected. The main area is a 'Worksheet' tab showing four numbered SQL queries:

```
--1
SELECT Emp_Name AS Name FROM JobProfile
UNION
SELECT Cust_Name AS Name FROM Customer;

--2
SELECT Emp_Name AS Name FROM JobProfile
UNION ALL
SELECT Cust_Name AS Name FROM Customer;

--3
SELECT Emp_Name AS Name FROM JobProfile
INTERSECT
SELECT Cust_Name AS Name FROM Customer;

--4
SELECT Emp_Name AS Name FROM JobProfile
MINUS
SELECT Cust_Name AS Name FROM Customer;
```

Below the worksheet is a 'Script Output' tab showing 'All Rows Fetched: 0 in 0.011 seconds' and a 'Query Result' tab showing a table with a single column 'NAME' containing no data.

4. Identify first names present in the employees table but not in customers.

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, showing a connection to 'C23CS32'. The main area is a 'SQL Worksheet' tab with the following SQL code:

```
SELECT Emp_Name AS Name FROM JobProfile
UNION
SELECT Cust_Name AS Name FROM Customer;

--2
SELECT Emp_Name AS Name FROM JobProfile
UNION ALL
SELECT Cust_Name AS Name FROM Customer;

--3
SELECT Emp_Name AS Name FROM JobProfile
INTERSECT
SELECT Cust_Name AS Name FROM Customer;

--4
SELECT Emp_Name AS Name FROM JobProfile
MINUS
SELECT Cust_Name AS Name FROM Customer;
```

Below the worksheet is a 'Script Output' tab showing the results of the fourth query:

NAME
1 Alice
2 Bob
3 Charlie
4 Eve
5 John

At the bottom, a status bar indicates 'All Rows Fetched: 5 in 0.017 seconds'.