

**Foundations of Databases - Final Project, Part 3**

Case Scenario - Public Library (Circulation)

Umma Islam

25FL-KG573-101 - Foundations of Database

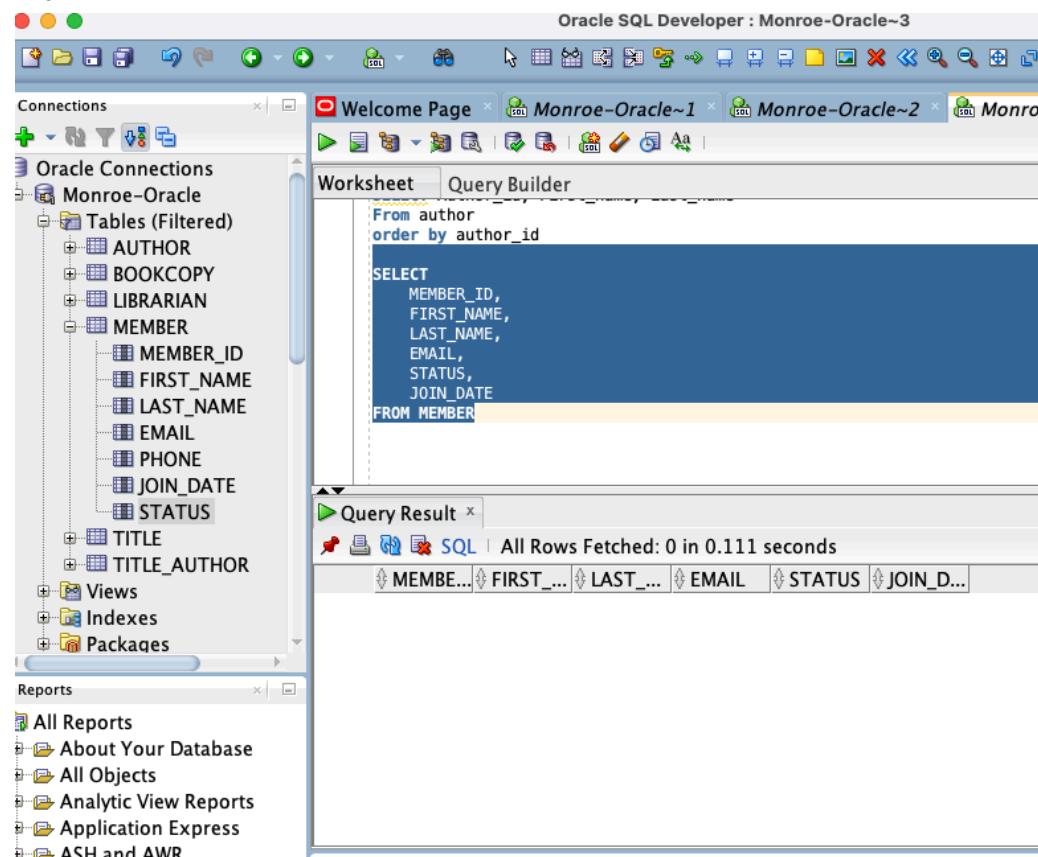
Date:11/18/2025

## 1. Basic Queries (SELECT)

Write at least 3 SELECT statements that use:

- Specific columns (projection)

```
SELECT  
    MEMBER_ID,  
    FIRST_NAME,  
    LAST_NAME,  
    EMAIL,  
    STATUS,  
    JOIN_DATE  
FROM MEMBER
```



- Conditions using WHERE

```
SELECT *  
FROM AUTHOR  
WHERE FIRST_NAME = 'Jane';
```

```

SELECT *
FROM AUTHOR
WHERE LAST_NAME = 'Jane';

```

AUTHOR_ID	FIRST_NAME	LAST_NAME
10	Jane	Austen
3	Jane	Austen

#### ▪ Sorting using ORDER BY

```

SELECT Author_id, First_name, Last_name
From author
order by author_id

```

```

SELECT Author_id, First_name, Last_name
From author
order by author_id

```

AUTHOR_ID	FIRST_NAME	LAST_NAME
1	J.K.	Rowling
2	George	Orwell
3	Jane	Austen
4	Mark	Twain
5	Agatha	Christie

## 2. Join Queries

Write at least 3 queries that combine data from two or more tables using JOINs.

Examples: INNER JOIN, LEFT JOIN

1. Using Inner join command:

```
SELECT
    a.FIRST_NAME,
    a.LAST_NAME,
    t.TITLE
FROM
    AUTHOR a
INNER JOIN
    TITLE_AUTHOR ta
ON
    a.AUTHOR_ID = ta.AUTHOR_ID
INNER JOIN
    TITLE t
ON
    ta.TITLE_ID = t.TITLE_ID;
```

The screenshot shows a database interface with a code editor and a results viewer. The code editor contains the SQL query for an inner join. Below it, a message says "Task completed in 0.562 seconds". The results viewer displays a table with three columns: FIRST\_NAME, LAST\_NAME, and TITLE. The data is as follows:

FIRST_NAME	LAST_NAME	TITLE
J.K.	Rowling	Harry Potter and the 1984
George	Orwell	Pride and Prejudice
Jane	Austen	The Adventures of Tom
Mark	Twain	Murder on the Orient
Agatha	Christie	

2. Using left join Command:

```
SELECT
    t.TITLE,
    a.FIRST_NAME,
    a.LAST_NAME
FROM TITLE t
LEFT JOIN TITLE_AUTHOR ta
ON t.TITLE_ID = ta.TITLE_ID
LEFT JOIN AUTHOR a
```

```
ON ta.AUTHOR_ID = a.AUTHOR_ID;
```

The screenshot shows a database interface with a query editor and a results viewer.

**Query Editor:**

```
SELECT
    a.FIRST_NAME,
    a.LAST_NAME,
    t.TITLE
FROM
    AUTHOR a
INNER JOIN
    TITLE_AUTHOR ta
ON
    a.AUTHOR_ID = ta.AUTHOR_ID
INNER JOIN
    TITLE t
ON
    ta.TITLE_ID = t.TITLE_ID;
```

**Results View:**

FIRST_NAME	LAST_NAME	TITLE
J.K.	Rowling	Harry Potter and the
George	Orwell	1984
Jane	Austen	Pride and Prejudice
Mark	Twain	The Adventures of Tom
Agatha	Christie	Murder on the Orient

### 3. Using Borrow command.

```
CREATE TABLE BORROW (
    BORROW_ID NUMBER GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    MEMBER_ID NUMBER,
    TITLE_ID NUMBER,
    BORROW_DATE DATE,
    RETURN_DATE DATE,
    CONSTRAINT FK_BORROW_MEMBER FOREIGN KEY (MEMBER_ID) REFERENCES
    MEMBER(MEMBER_ID),
    CONSTRAINT FK_BORROW_TITLE FOREIGN KEY (TITLE_ID) REFERENCES
    TITLE(TITLE_ID)
);
```

The screenshot shows the Oracle SQL Developer interface with the 'Welcome Page' selected. There are four tabs open in the top navigation bar: 'Monroe-Oracle', 'Monroe-Oracle~1', 'Monroe-Oracle~2', and 'AUTHOR'. The 'AUTHOR' tab is active. The main workspace is titled 'Worksheet' and contains a 'Query Builder' window. The query being run is:

```

CREATE TABLE BORROW (
    BORROW_ID NUMBER GENERATED ALWAYS AS IDENTITY PRIMARY KEY,
    MEMBER_ID NUMBER,
    TITLE_ID NUMBER,
    BORROW_DATE DATE,
    RETURN_DATE DATE,
    CONSTRAINT FK_BORROW_MEMBER FOREIGN KEY (MEMBER_ID) REFERENCES MEMBER(MEMBER_ID),
    CONSTRAINT FK_BORROW_TITLE FOREIGN KEY (TITLE_ID) REFERENCES TITLE(TITLE_ID)
);

```

Below the query, the 'Script Output' window shows the message: 'Task completed in 0.544 seconds'. The output pane displays the message: 'Table BORROW created.'

### 3. Aggregate Queries

- o Write at least 3 queries using aggregate functions such as COUNT, SUM, AVG, MIN, or MAX.
- o Use GROUP BY and HAVING where appropriate.

#### Group by:

```

SELECT genre, COUNT(*) AS total_titles
FROM Title
GROUP BY genre;

```

The screenshot shows the Oracle SQL Developer interface with the 'Welcome Page' selected. There are five tabs open in the top navigation bar: 'Monroe-Oracle~1', 'Monroe-Oracle~2', 'Monroe-Oracle~3', 'Monroe-Oracle~4', and 'MEMBER'. The 'MEMBER' tab is active. The main workspace is titled 'Worksheet' and contains a 'Query Builder' window. The query being run is:

```

SELECT t.title, a.first_name, a.last_name
FROM Title t
INNER JOIN TitleAuthor ta ON t.title_id = ta.title_id
INNER JOIN Author a ON ta.author_id = a.author_id;

SELECT genre, COUNT(*) AS total_titles
FROM Title
GROUP BY genre;

```

Below the query, the 'Script Output' window shows the message: 'Task completed in 0.143 seconds'. The output pane displays the results of the query:

GENRE	TOTAL_TITLE
Mystery	1
Adventure	1
Fantasy	1
Romance	1
Dystopian	1

### Count:

```
SELECT
    t.TITLE,
    COUNT(ta.AUTHOR_ID) AS AUTHOR_COUNT
FROM
    TITLE t
JOIN
    TITLE_AUTHOR ta
ON
    t.TITLE_ID = ta.TITLE_ID
GROUP BY
    t.TITLE
HAVING
    COUNT(ta.AUTHOR_ID) > 1;
```

The screenshot shows the Oracle SQL Developer interface. On the left is the Connections tree, with 'Monroe-Oracle' selected. The central area is the 'Worksheet' window containing the SQL query. The bottom right is the 'Script Output' window showing the execution results.

```
SELECT *
FROM AUTHOR
WHERE FIRST_NAME = 'Jane';

INSERT INTO TITLE (ISBN, TITLE, PUBLISHER, PUBLICATION_YEAR, GENRE)
SELECT '9780439708180', 'Harry Potter and the Sorcerer''s Stone', 'Scholastic', 1997, 'Fantasy'
FROM dual
WHERE NOT EXISTS (
    SELECT 1 FROM TITLE t WHERE t.ISBN = '9780439708180'
);
```

Script Output

Task completed in 0.111 seconds

\*Cause: The identifier or column name entered was invalid.  
\*Action: Ensure the following  
More Details : <https://docs.oracle.com/error-help/db/ora-00904/>

0 rows merged.

0 rows inserted.

### Sum:

```
SELECT
    t.TITLE,
    COUNT(ta.AUTHOR_ID) AS TOTAL_AUTHORS
FROM
```

```

TITLE t
LEFT JOIN
    TITLE_AUTHOR ta
ON
    t.TITLE_ID = ta.TITLE_ID
GROUP BY
    t.TITLE;

```

The screenshot shows a database interface with a query editor and a results viewer.

**Query Editor:**

```

SELECT
    t.TITLE,
    COUNT(ta.AUTHOR_ID) AS TOTAL_AUTHORS
FROM
    TITLE t
LEFT JOIN
    TITLE_AUTHOR ta
ON
    t.TITLE_ID = ta.TITLE_ID
GROUP BY
    t.TITLE;

```

**Results View:**

Script Output x | Query Result x

Task completed in 0.136 seconds

TITLE	TOTAL_AUTHORS
Pride and Prejudice	1
The Adventures of Tom Sawyer	1
Murder on the Orient Express	1
Harry Potter and the Sorcerer's Stone	1
1984	1

#### 4. Subqueries

o Write at least 2 queries that use subqueries (nested SELECT statements).

##### 1. Using nested SELECT statements

```

SELECT
    a.FIRST_NAME,
    a.LAST_NAME,
    (SELECT COUNT(*) FROM TITLE_AUTHOR ta WHERE ta.AUTHOR_ID =
    a.AUTHOR_ID) AS TOTAL_TITLES
FROM
    AUTHOR a
WHERE
    a.AUTHOR_ID IN (SELECT AUTHOR_ID FROM TITLE_AUTHOR WHERE TITLE_ID =
    1);

```

```

SELECT
    a.FIRST_NAME,
    a.LAST_NAME,
    (SELECT COUNT(*) FROM TITLE_AUTHOR ta WHERE ta.AUTHOR_ID = a.AUTHOR_ID) AS TOTAL_TITLES
FROM
    AUTHOR a
WHERE
    a.AUTHOR_ID IN (SELECT AUTHOR_ID FROM TITLE_AUTHOR WHERE TITLE_ID = 1);

```

Script Output x Task completed in 0.189 seconds

FIRST_NAME	LAST_NAME	TOTAL_TITLES
J.K.	Rowling	1

## 2. Using nested SELECT statements

```

SELECT FIRST_NAME, LAST_NAME
FROM AUTHOR
WHERE AUTHOR_ID = (
    SELECT MAX(AUTHOR_ID)
    FROM AUTHOR
);

```

```

SELECT FIRST_NAME, LAST_NAME
FROM AUTHOR
WHERE AUTHOR_ID = (
    SELECT MAX(AUTHOR_ID)
    FROM AUTHOR
);

```

Script Output x Query Result x Task completed in 0.129 seconds

FIRST_NAME	LAST_NAME	TOTAL_TITLES
J.K.	Rowling	1
FIRST_NAME	LAST_NAME	
Agatha	Christie	

## 5.Data Manipulation (DML)

- Include examples of the following commands:
  - INSERT – Add new records to a table
  - UPDATE – Modify existing records
  - DELETE – Remove records from a table

oProvide at least 2 examples of each command type

### 1. Using Update command:

INSERT INTO MEMBER (FIRST\_NAME, LAST\_NAME, EMAIL, PHONE, JOIN\_DATE, STATUS)

VALUES ('Carla', 'Gomez', 'carla.gomez@example.com', '555-111-0003',  
TO\_DATE('2023-03-15', 'YYYY-MM-DD'), 'SUSPENDED');

UPDATE Member SET status = 'SUSPENDED'

WHERE member\_id =1;

The screenshot shows the Oracle SQL Developer interface. In the top-left pane, there is a code editor window containing the following SQL script:

```
INSERT INTO MEMBER (FIRST_NAME, LAST_NAME, EMAIL, PHONE, JOIN_DATE, STATUS)
VALUES ('Carla', 'Gomez', 'carla.gomez@example.com', '555-111-0003',
       TO_DATE('2023-03-15', 'YYYY-MM-DD'), 'SUSPENDED');
UPDATE Member SET status = 'SUSPENDED'
WHERE member_id =1;
```

In the bottom-right pane, there is a "Script Output" window with the following log:

```
Script Output x
Task completed in 0.357 seconds

Error starting at line : 36 in command -
INSERT INTO MEMBER (FIRST_NAME, LAST_NAME, EMAIL, PHONE, JOIN_DATE, STATUS)
VALUES ('Carla', 'Gomez', 'carla.gomez@example.com', '555-111-0003',
       TO_DATE('2023-03-15', 'YYYY-MM-DD'), 'SUSPENDED')
Error report -
ORA-00001: unique constraint (UISLAM8880.UQ_MEMBER_EMAIL) violated
https://docs.oracle.com/error-help/db/ora-00001/
More Details :
https://docs.oracle.com/error-help/db/ora-00001/
1 row updated.
```

### 2. Using DELETE command:

INSERT INTO MEMBER (FIRST\_NAME, LAST\_NAME, EMAIL, PHONE, JOIN\_DATE, STATUS)

VALUES ('Emma', 'Brown', 'emma.brown@example.com', '555-111-0005',  
TO\_DATE('2023-05-01', 'YYYY-MM-DD'), 'INACTIVE');

DELETE FROM MEMBER

WHERE EMAIL = 'emma.brown@example.com';

```
INSERT INTO MEMBER (FIRST_NAME, LAST_NAME, EMAIL, PHONE, JOIN_DATE, STATUS)
VALUES ('Emma', 'Brown', 'emma.brown@example.com', '555-111-0005',
       TO_DATE('2023-05-01', 'YYYY-MM-DD'), 'INACTIVE');
DELETE FROM MEMBER
WHERE EMAIL = 'emma.brown@example.com';
```

Script Output 

    | Task completed in 0.717 seconds

1 row inserted.

1 row deleted.