SQL DEVELOPER

NAME: Akhil Sai Sammeta

TASK 2

Advanced Queries with Joins and Filtering

Objective

Analyze relationships between multiple tables and use SQL joins and filtering techniques to extract meaningful insights from the data.

Database Setup

Tables to Create

1. Students: Already created in Task 1. Contains student details such as student_id, name, and email.

2. Courses:

```
CREATE TABLE Courses (
   CourseID INT AUTO_INCREMENT PRIMARY KEY,
   CourseName VARCHAR(100) NOT NULL,
   CourseDescription TEXT
);
```

3. Enrolments:

```
CREATE TABLE Enrolments (
   EnrolmentID INT AUTO_INCREMENT PRIMARY KEY,
   StudentID INT,
   CourseID INT,
   EnrolmentDate DATE,
   FOREIGN KEY (StudentID) REFERENCES Students(StudentID),
   FOREIGN KEY (CourseID) REFERENCES Courses(CourseID)
);
```

Tasks to Perform

Task 1: List All Students and the Courses They Are Enrolled In

```
SELECT
   s.Name AS StudentName,
   c.CourseName AS CourseName
FROM
   Students s
INNER JOIN
   Enrolments e ON s.StudentID = e.StudentID
INNER JOIN
   Courses c ON e.CourseID = c.CourseID;
```

Purpose: Retrieve the names of students and the courses they are enrolled in using an INNER JOIN.

Observations: This query helps understand which students are enrolled in specific courses, ensuring relationships between tables are functioning as expected.

Task 2: Find the Number of Students Enrolled in Each Course

```
SELECT
   c.CourseName,
   COUNT(e.StudentID) AS NumberOfStudents
FROM
   Courses c
LEFT JOIN
   Enrolments e ON c.CourseID = e.CourseID
GROUP BY
   c.CourseID, c.CourseName;
```

Purpose: Count the number of students per course using LEFT JOIN and GROUP BY.

Observations: Courses with no enrolments are included, providing a complete view of course popularity.

Task 3: List Students Who Have Enrolled in More Than One Course

```
SELECT
   s.Name AS StudentName,
   COUNT(e.CourseID) AS CourseCount
FROM
```

```
   Students s
INNER JOIN
   Enrolments e ON s.StudentID = e.StudentID
GROUP BY
   s.StudentID, s.Name
HAVING
   COUNT(e.CourseID) > 1;
```

Purpose: Identify students enrolled in more than one course by grouping and using the HAVING clause.

Observations: Useful for finding students with diverse academic interests or heavy workloads.

Task 4: Find Courses with No Enrolled Students

```
SELECT
   c.CourseName
FROM
   Courses c
LEFT JOIN
   Enrolments e ON c.CourseID = e.CourseID
WHERE
   e.EnrolmentID IS NULL;
```

Purpose: Identify courses with no enrolments using a LEFT JOIN and filtering with WHERE.

Observations: Courses without enrolments can be flagged for potential removal or revision.

Output for Task 1

Sample Output Table: List All Students and the Courses They Are Enrolled In

-

StudentName	CourseName
Alice Johnson	Math
Bob Smith	Science
Charlie Brown	English
Diana Prince	Science
Ethan Hunt	History

Output for Task 2

Sample Output Table: Find the Number of Students Enrolled in Each Course

CourseName	*NumberOfStudents*
Math	*2*
Science	*2*
English	*1*
History	*1*

Output for Task 3

Sample Output Table: List Students Who Have Enrolled in More Than One Course

StudentName	*CourseCount*
Alice Johnson	*2*
Diana Prince	*3*

Output for Task 4

Sample Output Table: Find Courses with No Enrolled Students

CourseName
Philosophy
Economics