Python_Activity46

DB_Activity46 - Utilizing Advanced Database Queries

(Save this to your Drive, folder: **__DATABASE** and File name: **DB_Activity46**)

🏷️ Topic:

Joins, Subqueries, Aggregate Functions, and Window Functions

🎯 Learning Objectives:

By the end of this activity, students will:

- Perform different types of SQL JOINs (INNER, LEFT, RIGHT)
- Use subqueries (in `SELECT`, `WHERE`, and `FROM`)
- Apply common aggregate functions (SUM, AVG, COUNT, MIN, MAX)
- Master window functions (ROW_NUMBER, RANK, OVER, PARTITION BY)

📝 Instructions:

👨‍🎓 Scenario:

You continue using your **student enrollment database** (`enrollment.db`) with `Students`, `Courses`, `Instructors`, `Enrollments`.

🔍 Part 1: JOINS

Task 1.1 – INNER JOIN

- Display a list of all students with their enrolled courses and instructors.

sql

Copy code

```
SELECT s.name, c.course_name, i.full_name AS instructor
FROM Enrollments e
JOIN Students s ON e.student_id = s.student_id
```

```
JOIN Courses c ON e.course_id = c.course_id
JOIN Instructors i ON e.instructor_id = i.instructor_id;
```

Task 1.2 – LEFT JOIN

- Show all students and the courses they are enrolled in (including students not enrolled in any course).

🔁 Part 2: SUBQUERIES

Task 2.1 – Subquery in WHERE

- Display students who are enrolled in more than one course.

sql

Copy code

```
SELECT name FROM Students
WHERE student_id IN (SELECT student_id FROM EnrollmentsGROUP BY
student_idHAVING COUNT(*) > 1
);
```

Task 2.2 – Subquery in SELECT

- Show each student’ s name and total number of courses enrolled.

📊 Part 3: AGGREGATE FUNCTIONS

Task 3.1

- Show total number of students enrolled in each course.

sql

Copy code

```
SELECT course_id, COUNT(student_id) AS total_students
FROM Enrollments
GROUP BY course_id;
```

Task 3.2

- Show average final grade per course (if you have `final_grade` in Enrollments).

🔭 Part 4: WINDOW FUNCTIONS (Advanced)

Task 4.1 – ROW_NUMBER

- Assign a row number to each student per course, ordered by name.

sql

Copy code

```
SELECT student_id, course_id,ROW_NUMBER() OVER (PARTITION BY course_id
ORDER BY name) AS row_num
FROM Enrollments
JOIN Students USING(student_id);
```

Task 4.2 – RANK

- Rank students per course by grade.

🧠 Bonus Challenge:

Create a Python script `advanced_queries.py` to execute each query and print the results.
Example:

python

Copy code

```
cursor.execute(""" -- SQL QUERY HERE -- """)
rows = cursor.fetchall()
for row in rows:print(row)
```

📩 Deliverables:

- SQL script or `.py` file with all queries
- Console screenshots
- Short explanation of what each query does