**SQL Developer Task 3: Subqueries and Aggregations Documentation**

**Objective**

This project demonstrates the use of subqueries to extract insights and perform data aggregations to analyze and summarize the dataset effectively.

**Step 1: Database Setup**

**1.1 Purpose of Queries**

* **Students Table**: To store student details and their scores in Math, Science, and English.
* Sample data includes a range of scores to test subqueries and aggregations.

**Query to Create the Students Table**

CREATE TABLE Students (

student\_id INT AUTO\_INCREMENT PRIMARY KEY,

name VARCHAR(100),

math\_score INT,

science\_score INT,

english\_score INT

);

**Sample Data Insertion**

INSERT INTO Students (name, math\_score, science\_score, english\_score)

VALUES

('Alice', 85, 90, 88),

('Bob', 72, 80, 70),

('Charlie', 95, 88, 92),

('Diana', 65, 78, 72),

('Ethan', 88, 84, 85),

('Fiona', 70, 75, 68),

('George', 85, 85, 90),

('Hannah', 95, 90, 89);

**Step 2: Tasks to Perform**

**Task 1: Identify Top Students by Total Scores**

**Purpose**: To rank students by their total scores and identify the top performers.

**Query**:

SELECT

name,

(math\_score + science\_score + english\_score) AS total\_score

FROM

Students

ORDER BY

total\_score DESC

LIMIT 5;

**Explanation**:

* A subquery calculates the total score by summing up individual scores.
* Results are ordered in descending order to rank students by performance.
* The LIMIT clause restricts the output to the top 5 students.

**Observations**:

* Provides a ranked list of students with their total scores.

**Task 2: Calculate Averages Based on Specific Conditions**

**Purpose**: To analyze average scores under different conditions.

* **Query 1**: Average Math Score of Students Scoring Above 70 in Math
* SELECT AVG(math\_score) AS average\_math\_score
* FROM Students

WHERE math\_score > 70;

**Explanation**:

* + Filters students with math\_score > 70.
  + Uses AVG to calculate the average score of the filtered dataset.

**Observations**:

* + Provides the average Math score for high-performing students.
* **Query 2**: Average Total Score for Students Scoring 200–250
* SELECT AVG(total\_score) AS average\_total\_score
* FROM (
* SELECT
* (math\_score + science\_score + english\_score) AS total\_score
* FROM
* Students
* ) AS TotalScores

WHERE total\_score BETWEEN 200 AND 250;

**Explanation**:

* + Subquery calculates the total score for each student.
  + Outer query filters students whose total score is between 200 and 250, and calculates the average.

**Observations**:

* + Identifies trends in scores within the specified range.

**Task 3: Find Second-Highest Math Score**

**Purpose**: To determine the second-highest Math score.

**Query**:

SELECT MAX(math\_score) AS second\_highest\_math\_score

FROM Students

WHERE math\_score < (SELECT MAX(math\_score) FROM Students);

**Explanation**:

* Subquery finds the highest Math score using MAX.
* Outer query excludes the highest score and determines the next highest value.

**Observations**:

* Provides the second-highest Math score, useful for identifying near-top performers.

**Deliverables**

**1. SQL Queries**

* Queries provided for each task include inline comments for clarity.

**2. Observations and Insights**

* **Task 1**: Ranked list of students by total score highlights top performers.
* **Task 2**: Averages reveal performance trends under specific conditions.
* **Task 3**: Identifies second-highest Math scorer, capturing near-top performance.

**3. Screenshots of Results**

* Execute each query in a database environment and capture the outputs for reporting.

**Conclusion**

This project demonstrated how to use subqueries and aggregations to analyze student performance data effectively. These techniques are essential for summarizing and extracting meaningful insights from datasets, particularly in educational data analysis.

**Key Outcomes**:

1. Ranked top-performing students.
2. Average scores under specific conditions.
3. Identification of the second-highest Math score.

By completing this project, users gain proficiency in SQL subqueries, filtering, and aggregation functions.