Task 5: CASE Statements for Conditional Transformation

Objective

Use SQL `CASE` statements to transform and categorize data based on specified conditions.

Project Steps

1. Assign Grades Based on Total Scores

- **Goal**: Assign grades to students based on their total scores.
- **Logic**:
    - `CASE` statements can categorize students' total scores into grade brackets, e.g.:
        - `>= 90`: A
        - `>= 80`: B
        - `>= 70`: C
        - `< 70`: D (Fail)
- **Expected Output**:
    - Displays `StudentID`, `TotalScore`, and the assigned grade.

---

2. Identify Pass/Fail Status in Specific Subjects

- **Goal**: Check if a student passed or failed in each subject based on a passing threshold.
- **Logic**:
    - Use a `CASE` statement to evaluate individual subject scores:
        - Example: Pass if score ≥ 40, Fail otherwise.
- **Expected Output**:
    - Displays `StudentID`, individual subject scores, and their Pass/Fail status.

How to Execute

1. **Setup**:

Create and populate the `StudentScores` table with sample data:

```
 CREATE TABLE StudentScores (
   StudentID INT,
   TotalScore INT,
   MathScore INT,
   ScienceScore INT
);

INSERT INTO StudentScores (StudentID, TotalScore, MathScore, ScienceScore)
VALUES
(1, 95, 45, 50),
(2, 85, 35, 60),
(3, 75, 40, 30),
(4, 65, 25, 20);
```

   ○

2. **Execution**:
   - Run the queries sequentially:
     - Grade assignment query.
     - Pass/Fail status query.
3. **Validation**:
   - Compare the output against the sample data to ensure accuracy:
     - Verify grades align with total score ranges.
     - Confirm pass/fail status based on the score thresholds.

Documentation

1. Include Screenshots:

- Take screenshots of:
  - The executed queries.
  - Results of the queries.

2. Explanation:

- For each query:
  - Describe the logic used in the `CASE` statement.
  - Explain how the conditions were applied to assign grades or statuses.

3. Summary:

- Key insights:
  - Distribution of grades among students.
  - Number of students passing/failing in specific subjects.

General Guidelines

- Ensure proper testing with varied data to validate the logic.
- Handle edge cases, such as missing scores or total scores of exactly 90, 80, etc.

Use comments in queries for clarity, e.g.,:
-- Assign grades based on total scores

Would you like sample output or any help with edge cases or optimizations?

Deadline Compliance

- **Restriction**: **Submit the project within 7 days** from the start date.
- **Reason**: Meeting deadlines is crucial in the real-world software development environment. This restriction helps students practice **time management** and **task prioritization**. In professional settings, tight deadlines are often the norm, and learning to meet them without compromising quality is an essential skill.
- **Learning Outcome**: Students will learn to manage their time effectively, complete projects under pressure, and **deliver results on time**, which are all important skills in the workplace.