Practical - 6: AWS Athena Data Partitioning (Pranav Paralkar)

Objective: To compare the query performance and cost efficiency between non-partitioned and partitioned data structures in Amazon S3 using AWS Athena.

Part 1: Setup Steps and Table Definitions

Step 1: Prepare S3 Buckets and Data Structure

1. **Create S3 Buckets:** Create two S3 buckets (in the same region):
   - Data Bucket: **`pranav-athena-practical`**
   - Results Bucket: **`pranav-athena-practical-results`** (for Athena query output)

2. **Upload Data into S3 bucket :** Upload the complete file (`student_habits_performance.csv`) into bucket

Step 2: Configure Athena and Create Database

1. **Set Athena Result Location:** In the AWS Athena console, go to **Manage settings**. Set the **Query result location** to **`s3://pranav-athena-practical-results`**.
2.
1. **Create Database:** Run the following command in the Query Editor:
   SQL
   ```
   CREATE DATABASE IF NOT EXISTS prantest;
   ```

Step 3: Create Non-Partitioned Table (`students_data`)

This table scans the entire folder containing the full dataset.

SQL

```
CREATE EXTERNAL TABLE IF NOT EXISTS students.students_data (
```

```
   student_id                    STRING,
   age                           INT,
   gender                        STRING,
   study_hours_per_day           DOUBLE,
   social_media_hours            DOUBLE,
   netflix_hours                 DOUBLE,
   part_time_job                 STRING,
   attendance_percentage         DOUBLE,
   sleep_hours                   DOUBLE,
   diet_quality                  STRING,
   exercise_frequency            INT,
   internet_quality              STRING,
   mental_health_rating          INT,
   extracurricular_participation   STRING,
   exam_score                    DOUBLE
)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION 's3://athena-pranav-pract6/'
TBLPROPERTIES ('skip.header.line.count'='1');
```

Step 4: Create Partitioned Table (`partitioned_students_data`)

This table is defined with `parental_education_level` as the partition key.

SQL

```
CREATE EXTERNAL TABLE IF NOT EXISTS students.partitioned_students_data (
   student_id                 STRING,
   age                        INT,
   gender                     STRING,
   study_hours_per_day        DOUBLE,
   social_media_hours         DOUBLE,
   netflix_hours              DOUBLE,
   part_time_job              STRING,
   attendance_percentage      DOUBLE,
   sleep_hours                DOUBLE,
   diet_quality               STRING,
   exercise_frequency         INT,
   internet_quality           STRING,
   mental_health_rating       INT,
   extracurricular_participation   STRING,
   exam_score                 DOUBLE
)
PARTITIONED BY (parental_education_level STRING) -- Partition Key
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
STORED AS TEXTFILE
LOCATION 's3://athena-pranav-pract6/'
TBLPROPERTIES ('skip.header.line.count'='1');
```

Step 5: Load Partitions

Run this command to tell Athena where the partitioned data resides in S3:

SQL

2. MSCK REPAIR TABLE prantest.partitioned_students_data;

Part 2: SQL Queries

4 Easy Queries (Non-Partitioned Focus)

These queries are used for general analysis and typically scan the entire table, regardless of partitioning setup.

1️⃣ NEW: Find the highest exam score and the student who achieved it

```
SELECT student_id, exam_score

FROM students.students_data

ORDER BY exam_score DESC

LIMIT 1;
```

2️⃣ NEW: Count students who sleep less than 6 hours per day

```
SELECT COUNT(student_id) AS sleep_less_than_6hrs

FROM students.students_data

WHERE sleep_hours < 6;
```

4️⃣ NEW: Find total students grouped by gender

```
SELECT gender, COUNT(*) AS student_count

FROM students.students_data

GROUP BY gender;
```

5️⃣ NEW: List students who study more than 5 hours AND have exam_scores above 80

```
SELECT student_id, study_hours_per_day, exam_score

FROM students.students_data

WHERE study_hours_per_day > 5

  AND exam_score > 80;
```

7️⃣ NEW: Average exam score by parental education level

```
SELECT part_time_job, COUNT(student_id) AS total_students

FROM students.students_data

GROUP BY part_time_job;
```