Name: Dev Parekh

Class: TE9-B-42

Subject: DWM

Experiment No. 2

Title: Implementation of OLAP operations.

Problem Statement:

The Mumbai University wants to help to design star schema to record grade for course completed by students. There are 4 dimensions tables: Course section, Professor, Student & Period with attributes as follows.

- **Course sections:** Course_ID, Section_ID, Course_Name, Units, Room_ID, Room_Capacity. During a given semester the college offers an average of 500 Course Sections.
- **Professor:** Professor_ID, Professor_Name, Title, Department_ID, Department_Name
- **Student:** Student_ID, Student_Major. Each course section has an average of 60 students.
- **Period:** Semester_ID, Year

It will contain data for 30 months period. The only fact that is to be recorded in the fact table is course grade. Design Star schema and Snowflake schema for above case.

Aim: Implementation of OLAP operations: Slice, Dice, Rollup, Drilldown and Pivot for the above problem statement (experiment 1).

Theory:

OLAP (Online Analytical Processing) operations help in analyzing multidimensional data from various perspectives to support business decision-making. The core OLAP operations include:

- **Slice**: Selects a single dimension from a cube, resulting in a sub-cube. For example, filtering the data for only the year 2024.

- **Dice**: Selects multiple dimensions to create a sub-cube. For example, filtering the data for "Computer Engineering" department and "2023–2024" years.
- **Roll-up**: Aggregates data by climbing up the hierarchy or reducing dimensions. For example, calculating the total number of grades per course, regardless of year.
- **Drill-down**: Opposite of roll-up; allows viewing more detailed data. For example, viewing grades not just by course, but by course and student.
- **Pivot**: Also called rotation; it reorients the multidimensional view of data. For example, turning year values into columns instead of rows to view grade averages side-by-side.

These operations are fundamental to decision support systems and allow for interactive and intuitive data exploration.

Program Output:

Create Tables

```
-- Dimension Tables
CREATE TABLE Course_Section (
    Course_ID INT,
    Section_ID INT,
    Course_Name VARCHAR(50),
    Units INT,
    Room_ID VARCHAR(10),
    Room_Capacity INT
);

CREATE TABLE Professor (
    Professor_ID INT,
    Professor_Name VARCHAR(50),
    Title VARCHAR(10),
    Department_ID INT,
    Department_Name VARCHAR(50)
);

CREATE TABLE Student (
    Student_ID INT,
    Student_Major VARCHAR(50)
);
```

```sql
CREATE TABLE Period (
    Semester_ID INT,
    Year INT
);

-- Fact Table
CREATE TABLE Fact_Grades (
    Course_ID INT,
    Section_ID INT,
    Professor_ID INT,
    Student_ID INT,
    Semester_ID INT,
    Course_Grade CHAR(1)
);
```

->

```
Table created.

Table created.

Table created.

Table created.

Table created.
```

Insert Sample Data

```sql
-- Course_Section
INSERT INTO Course_Section VALUES (101, 1, 'Data Mining', 4, 'R101', 60);
INSERT INTO Course_Section VALUES (102, 1, 'DBMS', 3, 'R102', 60);
INSERT INTO Course_Section VALUES (103, 1, 'Operating Systems', 4, 'R103', 60);
INSERT INTO Course_Section VALUES (104, 2, 'Machine Learning', 3, 'R104', 50);
INSERT INTO Course_Section VALUES (105, 1, 'Computer Networks', 3, 'R105', 55);

-- Professor
INSERT INTO Professor VALUES (1, 'Dr. Sharma', 'Prof.', 10, 'Computer Engineering');
INSERT INTO Professor VALUES (2, 'Ms. Iyer', 'Asst.', 11, 'IT');
INSERT INTO Professor VALUES (3, 'Mr. Deshmukh', 'Lect.', 12, 'Computer Science');
INSERT INTO Professor VALUES (4, 'Dr. Rao', 'Prof.', 13, 'Electronics');
INSERT INTO Professor VALUES (5, 'Ms. Naik', 'Asst.', 10, 'Computer Engineering');

-- Student
INSERT INTO Student VALUES (1001, 'Computer Science');
INSERT INTO Student VALUES (1002, 'Information Technology');
INSERT INTO Student VALUES (1003, 'Electronics');
INSERT INTO Student VALUES (1004, 'Computer Engineering');
INSERT INTO Student VALUES (1005, 'Artificial Intelligence');
```

```sql
-- Period
INSERT INTO Period VALUES (1, 2022);
INSERT INTO Period VALUES (2, 2023);
INSERT INTO Period VALUES (3, 2024);
INSERT INTO Period VALUES (4, 2025);
INSERT INTO Period VALUES (5, 2026);
INSERT INTO Period VALUES (6, 2027);

-- Fact_Grades
INSERT INTO Fact_Grades VALUES (101, 1, 1, 1001, 2, 'A');
INSERT INTO Fact_Grades VALUES (101, 1, 1, 1002, 3, 'B');
INSERT INTO Fact_Grades VALUES (102, 1, 2, 1001, 3, 'C');
INSERT INTO Fact_Grades VALUES (103, 1, 3, 1003, 4, 'B');
INSERT INTO Fact_Grades VALUES (104, 2, 4, 1004, 5, 'A');
INSERT INTO Fact_Grades VALUES (105, 1, 5, 1005, 6, 'A');
```

->

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

```
1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.

1 row(s) inserted.
```

Display Full Joined Table

```
SELECT DISTINCT
    f.Course_ID,
    cs.Section_ID,
    cs.Course_Name,
    cs.Units,
    cs.Room_ID,
    cs.Room_Capacity,

    f.Professor_ID,
    p.Professor_Name,
    p.Title,
    p.Department_ID,
    p.Department_Name,

    f.Student_ID,
    s.Student_Major,

    f.Semester_ID,
    pr.Year,

    f.Course_Grade

FROM Fact_Grades f
JOIN Course_Section cs ON f.Course_ID = cs.Course_ID AND f.Section_ID = cs.Section_ID
JOIN Professor p ON f.Professor_ID = p.Professor_ID
JOIN Student s ON f.Student_ID = s.Student_ID
JOIN Period pr ON f.Semester_ID = pr.Semester_ID;
```

->

COURSE_ID	SECTION_ID	COURSE_NAME	UNITS	ROOM_ID	ROOM_CAPACITY	PROFESSOR_ID	PROFESSOR_NAME	TITLE	DEPARTMENT_ID	DEPARTMENT_NAME	STUDENT_ID	STUDENT_MAJOR	SEMESTER_ID	YEAR	COURSE_
101	1	Data Mining	4	R101	60	1	Dr. Sharma	Prof.	10	Computer Engineering	1001	Computer Science	2	2023	A
102	1	DBMS	3	R102	60	2	Ms. Iyer	Asst.	11	IT	1001	Computer Science	3	2024	C
101	1	Data Mining	4	R101	60	1	Dr. Sharma	Prof.	10	Computer Engineering	1002	Information Technology	3	2024	B
103	1	Operating Systems	4	R103	60	3	Mr. Deshmukh	Lect.	12	Computer Science	1003	Electronics	4	2025	B
104	2	Machine Learning	3	R104	50	4	Dr. Rao	Prof.	13	Electronics	1004	Computer Engineering	5	2026	A
105	1	Computer Networks	3	R105	55	5	Ms. Naik	Asst.	10	Computer Engineering	1005	Artificial Intelligence	6	2027	A

6 rows selected.

Operations

1. SLICE :

Get all course grades for a **specific semester (e.g., Semester 3)**

```
SELECT *
FROM Fact_Grades
WHERE Semester_ID = 3;
```

->

COURSE_ID	SECTION_ID	PROFESSOR_ID	STUDENT_ID	SEMESTER_ID	COURSE_GRADE
101	1	1	1002	3	B
102	1	2	1001	3	C

COURSE_ID	SECTION_ID	PROFESSOR_ID	STUDENT_ID	SEMESTER_ID	COURSE_GRADE
101	1	1	1002	3	B
102	1	2	1001	3	C

Download CSV

2 rows selected.

2. DICE :

Get course grades for:

- Students with **major 'Computer Science'**
- Taught by professors from **'Computer Engineering'** department
- In **year 2023**

```
SELECT *
FROM Fact_Grades f
JOIN Student s ON f.Student_ID = s.Student_ID
JOIN Professor p ON f.Professor_ID = p.Professor_ID
JOIN Period t ON f.Semester_ID = t.Semester_ID
WHERE s.Student_Major = 'Computer Science'
  AND p.Department_Name = 'Computer Engineering'
  AND t.Year = 2023;
```

->

COURSE_ID	SECTION_ID	COURSE_NAME	UNITS	ROOM_ID	ROOM_CAPACITY	PROFESSOR_ID	PROFESSOR_NAME	TITLE	DEPARTMENT_ID	DEPARTMENT_NAME	STUDENT_ID	STUDENT_MAJOR	SEMESTER_ID	YEAR	COURSE_
101	1	Data Mining	4	R101	60	1	Dr. Sharma	Prof.	10	Computer Engineering	1001	Computer Science	2	2023	A

1 row inserted.

3. ROLL-UP :

Get **total number of students graded** by:

- **Course Name**
- **Year**

Including subtotals per course and grand total.

```
SELECT cs.Course_Name, pr.Year, COUNT(f.Course_Grade) AS Total_Grades
FROM Fact_Grades f
JOIN Course_Section cs ON f.Course_ID = cs.Course_ID
JOIN Period pr ON f.Semester_ID = pr.Semester_ID
GROUP BY ROLLUP(cs.Course_Name, pr.Year);
```

->

COURSE_NAME	YEAR	TOTAL_GRADES
DBMS	2024	1
DBMS	-	1
Data Mining	2023	1
Data Mining	2024	1
Data Mining	-	2
Machine Learning	2026	1
Machine Learning	-	1
Computer Networks	2027	1
Computer Networks	-	1
Operating Systems	2025	1
Operating Systems	-	1
-	-	6

Download CSV

12 rows selected.

4. DRILL-DOWN :

Break down the number of students graded from:

Department → Professor

```sql
SELECT p.Department_Name, p.Professor_Name, COUNT(f.Student_ID) AS Students_Graded
FROM Fact_Grades f
JOIN Professor p ON f.Professor_ID = p.Professor_ID
GROUP BY p.Department_Name, p.Professor_Name;
```

->

DEPARTMENT_NAME	PROFESSOR_NAME	STUDENTS_GRADED
Computer Science	Mr. Deshmukh	1
Electronics	Dr. Rao	1
Computer Engineering	Dr. Sharma	2
IT	Ms. Iyer	1
Computer Engineering	Ms. Naik	1

Download CSV

5 rows selected.

5. PIVOT :

Show the **average grade per course across years**

```sql
SELECT cs.Course_Name,
       AVG(CASE WHEN pr.Year = 2022 THEN ASCII(f.Course_Grade) END) AS "2022",
       AVG(CASE WHEN pr.Year = 2023 THEN ASCII(f.Course_Grade) END) AS "2023",
       AVG(CASE WHEN pr.Year = 2024 THEN ASCII(f.Course_Grade) END) AS "2024",
       AVG(CASE WHEN pr.Year = 2025 THEN ASCII(f.Course_Grade) END) AS "2025",
       AVG(CASE WHEN pr.Year = 2026 THEN ASCII(f.Course_Grade) END) AS "2026",
       AVG(CASE WHEN pr.Year = 2027 THEN ASCII(f.Course_Grade) END) AS "2027"
FROM Fact_Grades f
JOIN Course_Section cs ON f.Course_ID = cs.Course_ID
JOIN Period pr ON f.Semester_ID = pr.Semester_ID
GROUP BY cs.Course_Name;
```

->

COURSE_NAME	2022	2023	2024	2025	2026	2027
DBMS	-	-	67	-	-	-
Machine Learning	-	-	-	-	65	-
Operating Systems	-	-	-	66	-	-
Computer Networks	-	-	-	-	-	65
Data Mining	-	65	66	-	-	-

Download CSV

5 rows selected.

Review Questions:

1. What is the difference between the Slice and Dice operations in OLAP?

Ans:

- Slice reduces the dimensionality of the data by selecting a single value for one of the dimensions (e.g., selecting data for Year = 2024).
- Dice creates a sub-cube by selecting multiple values from multiple dimensions (e.g., selecting data for Department = 'IT' and Years = 2023 or 2024).

2. How does the Roll-up operation help in summarizing large volumes of data?

Roll-up helps in summarizing large data sets by aggregating values along a dimension hierarchy. For example, it can summarize student grades by course, and then further summarize by department. This reduces the data complexity and helps in identifying overall trends.

3. Give an example of a scenario where Pivoting the data provides a clearer insight than traditional tabular view.

Ans: When analyzing student grades across multiple years, using a pivot table allows you to view each year's average grade side-by-side per course. This makes it easier to compare performance trends over time than a vertical listing of year-wise entries.

Conclusion: In this experiment, we successfully implemented core OLAP operations—Slice, Dice, Roll-up, Drill-down, and Pivot—on a university course grading dataset. These operations provided deeper insights and simplified data analysis by allowing multidimensional views and summaries. OLAP techniques are essential for decision-making and performance evaluation in academic and business environments.

GitHub Link: https://github.com/Devp71/DWM