



Veermata Jijabai Technological Institute, Mumbai 400019

Experiment No.: 02

Aim: To perform various OLAP operations such as: slice, dice, roll up, drill up etc.

Name: Kiran K Patil

Enrolment No.: 211070904

Branch: Computer Engineering

Batch: D

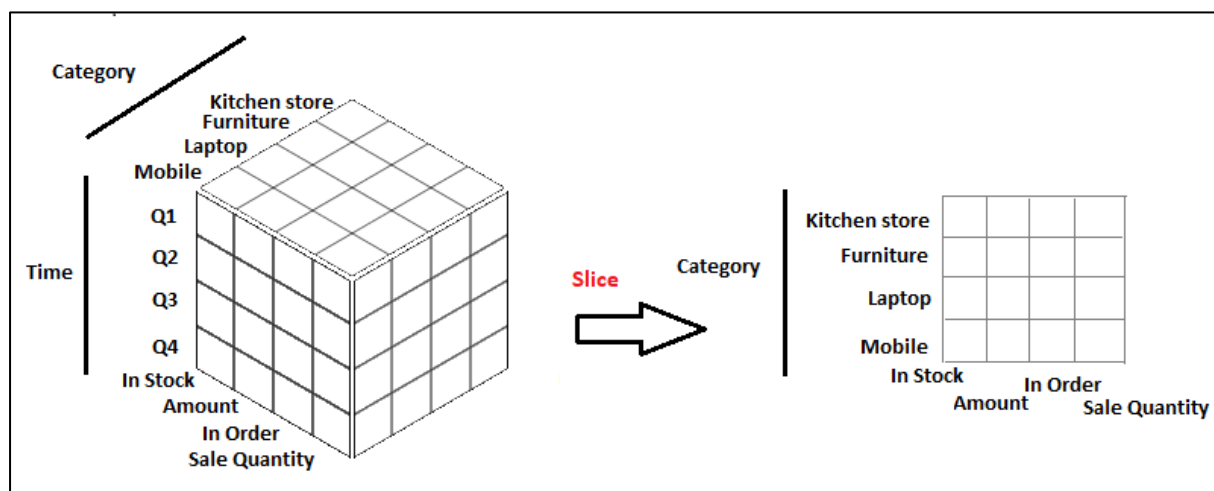
Theory:

Online Analytical Processing (OLAP) operations are fundamental for analyzing and gaining insights from multidimensional data. They allow users to navigate, summarize, and manipulate data in a way that's tailored to their specific analysis needs. Here's a brief overview of various OLAP operations:

Slice:

Definition: Slicing is the process of selecting a single dimension from a multidimensional cube, thereby reducing the cube to a two-dimensional slice. It allows you to view a cross-section of data.

Example: If you have a three-dimensional cube representing sales data with dimensions for time, product, and region, you can slice the cube by selecting a specific time period (e.g., a particular quarter), resulting in a two-dimensional table displaying sales data for that quarter across different products and regions.

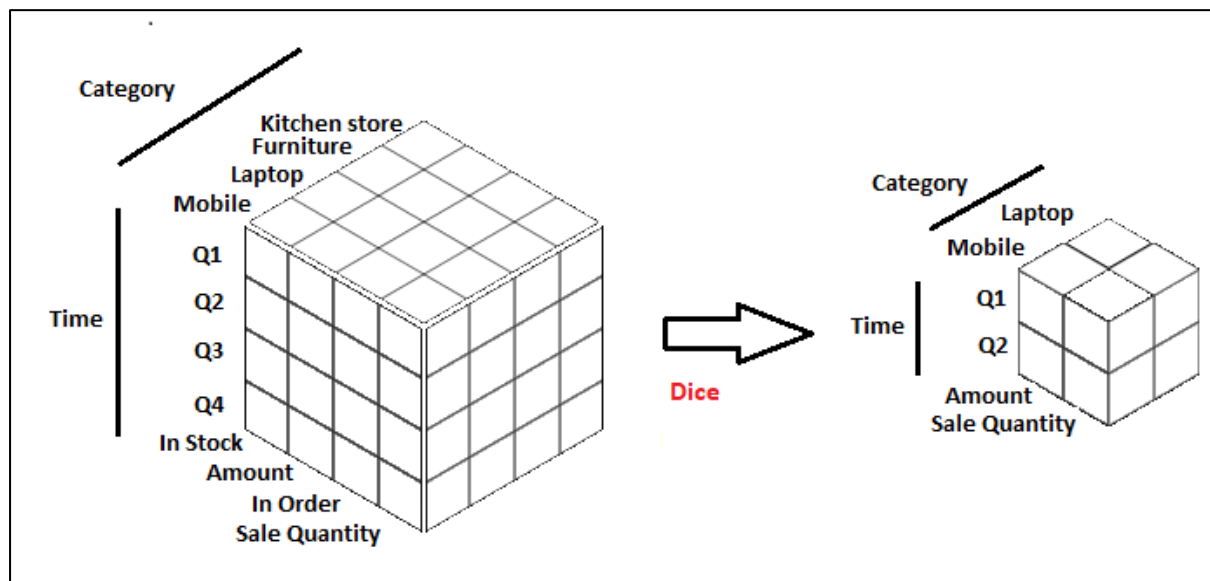


Dice:

Definition: Dicing is the operation of creating a subcube by selecting specific values or ranges along multiple dimensions. It's a more targeted operation than slicing and allows you to focus on a subset of the data.

Example: In the sales data cube example, dicing could involve selecting a particular quarter, a specific product category, and a particular region to analyze sales for a specific segment of the data.

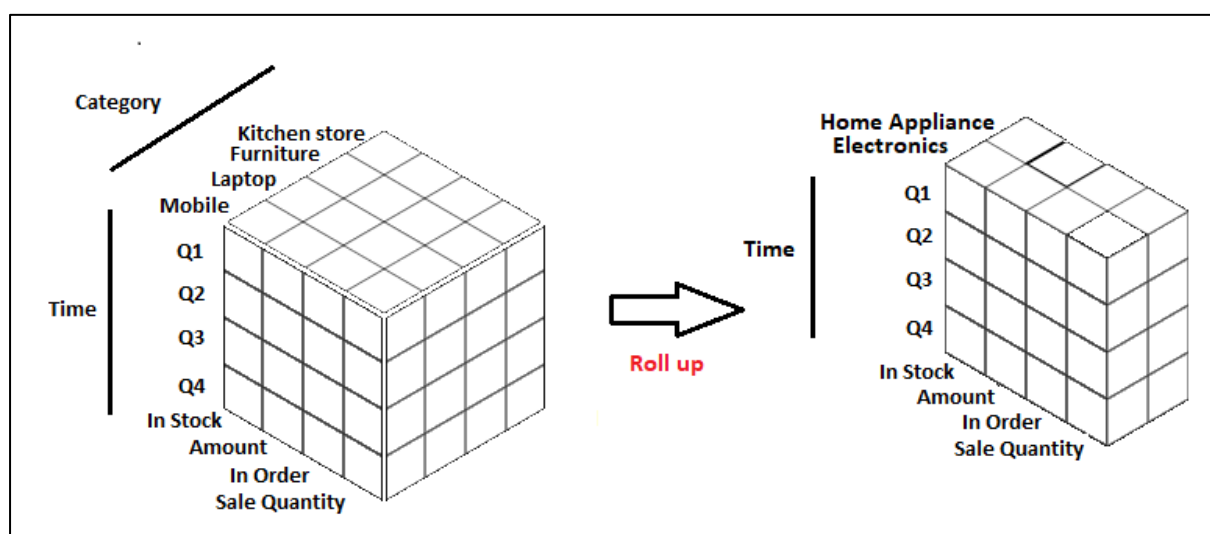
Dicing



Roll-Up:

Definition: Rolling up, also known as aggregation, is the process of summarizing data from a lower level of detail to a higher level within a single dimension. It allows you to view the data at a coarser granularity.

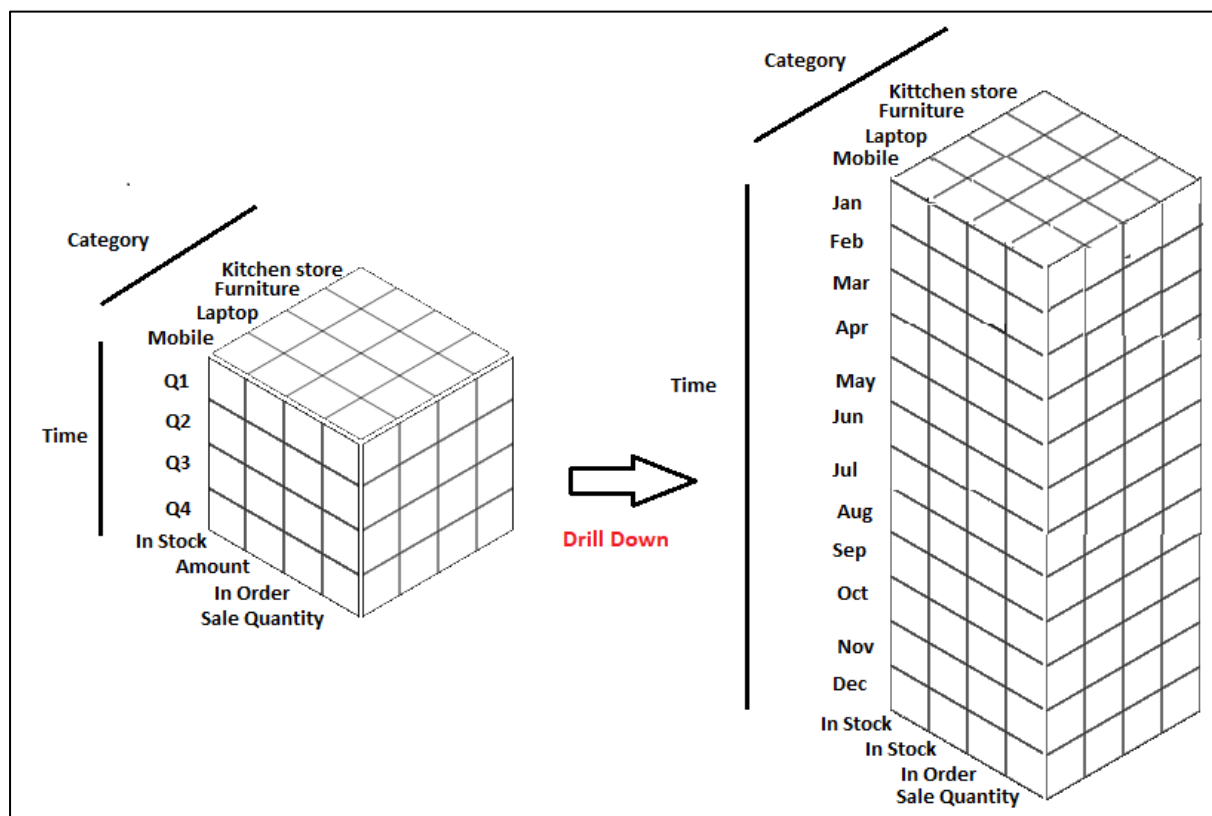
Example: You can roll up sales data from the product level to the product category level. This aggregates sales figures for individual products to provide total sales for product categories, simplifying the view of the data.



Drill-Down:

Definition: Drilling down is the reverse of rolling up. It involves navigating from a higher level of aggregation to a lower level of detail within a single dimension. It allows you to explore detailed data.

Example: From the product category level, you can drill down to see sales figures for individual products within a specific category, providing more granularity in your analysis.

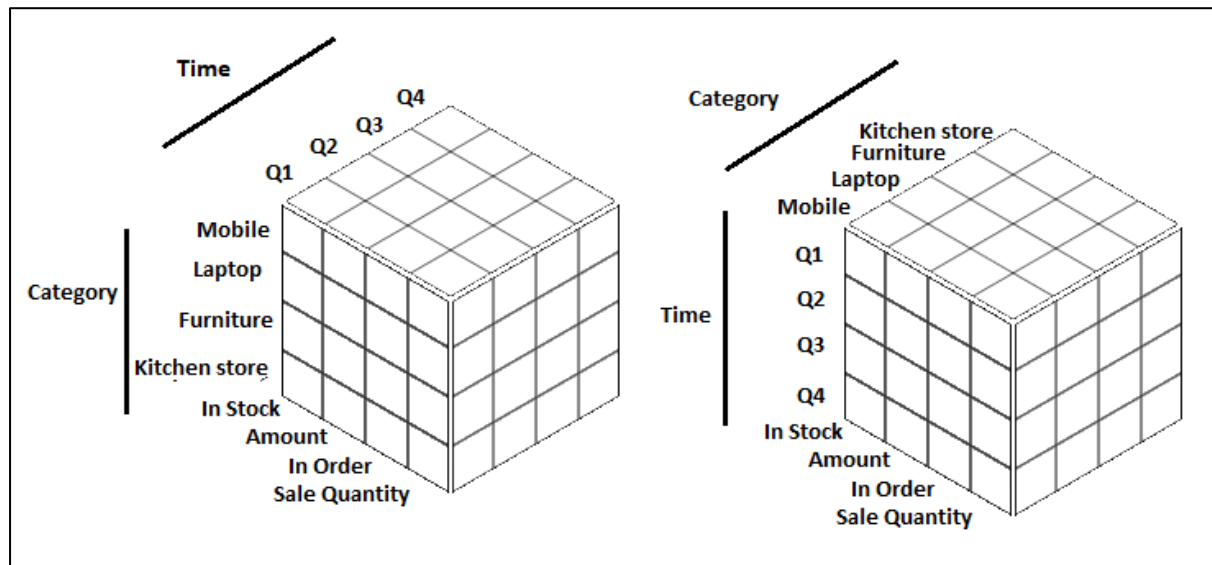


Pivot:

Definition: Pivoting is a method for reorienting the axes of a multidimensional cube. It can change the orientation of rows and columns to view data from a different perspective.

Example: Instead of viewing sales data by time, product, and region, you can pivot the cube to see sales data by region, time, and product. This can be useful for comparing sales across regions over time.

Pivot :



Top-N Analysis:

Definition: Top-N analysis involves identifying and displaying the top or bottom N members within a dimension based on a specific measure. It helps identify the most significant or least significant data points.

Example: You can use Top-N analysis to find the top 10 customers with the highest sales figures within a specific time period.

These OLAP operations are essential for business intelligence and data analysis, enabling users to interact with data in a way that makes it easier to uncover insights, trends, and patterns within multidimensional datasets. The choice of operation depends on the analytical goals and the specific dimensions and measures of the dataset.

Output:

```
Command Prompt - mysql -u root -p

mysql> create database Kiran_2;
Query OK, 1 row affected (0.02 sec)

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| kiran        |
| kiran_2      |
| mysql       |
| patient     |
| performance_schema |
| sys         |
+-----+
7 rows in set (0.00 sec)

mysql> use Kiran_2;
Database changed
mysql> show tables;
Empty set (0.00 sec)

mysql> CREATE TABLE SalesFact (
    ->   SaleID SERIAL PRIMARY KEY,
    ->   Date DATE,
    ->   ProductID INT,
    ->   CustomerID INT,
    ->   SalesAmount DECIMAL(10, 2)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> CREATE TABLE Time (
    ->   TimeID SERIAL PRIMARY KEY,
    ->   Date DATE,
    ->   Month INT,
    ->   Quarter INT,
    ->   Year INT
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE Product (
    ->   ProductID SERIAL PRIMARY KEY,
    ->   ProductName VARCHAR(255),
    ->   Category VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE Customer (
    ->   CustomerID SERIAL PRIMARY KEY,
```

```
Command Prompt - mysql -u root -p

mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| kiran        |
| kiran_2      |
| mysql       |
| patient     |
| performance_schema |
| sys         |
+-----+
7 rows in set (0.00 sec)

mysql> use Kiran_2;
Database changed
mysql> show tables;
Empty set (0.00 sec)

mysql> CREATE TABLE SalesFact (
    ->   SaleID SERIAL PRIMARY KEY,
    ->   Date DATE,
    ->   ProductID INT,
    ->   CustomerID INT,
    ->   SalesAmount DECIMAL(10, 2)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> CREATE TABLE Time (
    ->   TimeID SERIAL PRIMARY KEY,
    ->   Date DATE,
    ->   Month INT,
    ->   Quarter INT,
    ->   Year INT
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE Product (
    ->   ProductID SERIAL PRIMARY KEY,
    ->   ProductName VARCHAR(255),
    ->   Category VARCHAR(50)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE Customer (
    ->   CustomerID SERIAL PRIMARY KEY,
    ->   FirstName VARCHAR(50),
    ->   LastName VARCHAR(50),
    ->   Email VARCHAR(100)
    -> );
```

```
Command Prompt - mysql -u root -p
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| kiran       |
| kiran_2     |
| mysql       |
| patient     |
| performance_schema |
| sys        |
+-----+
7 rows in set (0.00 sec)

mysql> use kiran_2;
Database changed
mysql> show tables;
Empty set (0.00 sec)

mysql> CREATE TABLE SalesFact (
  ->   SaleID SERIAL PRIMARY KEY,
  ->   Date DATE,
  ->   ProductID INT,
  ->   CustomerID INT,
  ->   SalesAmount DECIMAL(10, 2)
  -> );
Query OK, 0 rows affected (0.06 sec)

mysql> CREATE TABLE Time (
  ->   TimeID SERIAL PRIMARY KEY,
  ->   Date DATE,
  ->   Month INT,
  ->   Quarter INT,
  ->   Year INT
  -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE Product (
  ->   ProductID SERIAL PRIMARY KEY,
  ->   ProductName VARCHAR(255),
  ->   Category VARCHAR(50)
  -> );
Query OK, 0 rows affected (0.02 sec)

mysql> CREATE TABLE Customer (
  ->   CustomerID SERIAL PRIMARY KEY,
  ->   FirstName VARCHAR(50),
  ->   LastName VARCHAR(50),
  ->   Email VARCHAR(100)
  -> );
```

```
Command Prompt - mysql -u root -p
mysql> show tables;
+-----+
| Tables_in_kiran_2 |
+-----+
| customer           |
| product            |
| salesfact          |
| time               |
+-----+
4 rows in set (0.00 sec)

mysql> select * from customer;
Empty set (0.01 sec)

mysql> INSERT INTO Time (Date, Month, Quarter, Year)
  -> VALUES
  -> ('2023-01-01', 1, 1, 2023),
  -> ('2023-02-01', 2, 1, 2023),
  -> ('2023-03-01', 3, 1, 2023),
  -> ('2023-04-01', 4, 2, 2023);
Query OK, 4 rows affected (0.01 sec)
Records: 4 Duplicates: 0 Warnings: 0

mysql> INSERT INTO Product (ProductName, Category)
  -> VALUES
  -> ('Product A', 'Electronics'),
  -> ('Product B', 'Clothing'),
  -> ('Product C', 'Electronics'),
  -> ('Product D', 'Furniture'),
  -> ('Product E', 'Clothing');
Query OK, 5 rows affected (0.01 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> INSERT INTO Customer (FirstName, LastName, Email)
  -> VALUES
  -> ('Kiran', 'Doe', 'Kirandoe@email.com'),
  -> ('Dip', 'Smith', 'Dipsmith@email.com'),
  -> ('Robert', 'son', 'robertson@email.com'),
  -> ('Emily', 'Wilson', 'emilywilson@email.com');
Query OK, 4 rows affected (0.00 sec)
Records: 4 Duplicates: 0 Warnings: 0

mysql> INSERT INTO SalesFact (Date, ProductID, CustomerID, SalesAmount)
  -> VALUES
  -> ('2023-01-05', 1, 1, 500.00),
  -> ('2023-02-15', 2, 2, 300.00),
  -> ('2023-03-10', 1, 3, 750.00),
  -> ('2023-04-20', 3, 4, 400.00);
Query OK, 4 rows affected (0.00 sec)
```

```
Command Prompt - mysql -u root -p
mysql> select* from customer;
+-----+-----+-----+-----+
| CustomerID | FirstName | LastName | Email |
+-----+-----+-----+-----+
| 1 | Kiran | Doe | Kirandoe@gmail.com |
| 2 | Dip | Smith | Dipsmith@gmail.com |
| 3 | Robert | son | robertson@gmail.com |
| 4 | Emily | Wilson | emilywilson@gmail.com |
+-----+-----+-----+-----+
4 rows in set (0.00 sec)

mysql> INSERT INTO Time (Date, Month, Quarter, Year)
-> VALUES
-> ('2023-05-01', 5, 2, 2023),
-> ('2023-06-01', 6, 2, 2023),
-> ('2023-07-01', 7, 3, 2023),
-> ('2023-08-01', 8, 3, 2023),
-> ('2023-09-01', 9, 3, 2023),
-> ('2023-10-01', 10, 4, 2023);
Query OK, 6 rows affected (0.00 sec)
Records: 6 Duplicates: 0 Warnings: 0

mysql> INSERT INTO Product (ProductName, Category)
-> VALUES
-> ('Product F', 'Electronics'),
-> ('Product G', 'Clothing'),
-> ('Product H', 'Furniture'),
-> ('Product I', 'Electronics'),
-> ('Product J', 'Clothing'),
-> ('Product K', 'Furniture');
Query OK, 6 rows affected (0.01 sec)
Records: 6 Duplicates: 0 Warnings: 0

mysql> INSERT INTO Customer (FirstName, LastName, Email)
-> VALUES
-> ('Sarah', 'Miller', 'sarahmiller@gmail.com'),
-> ('Michael', 'Davis', 'michaeldavis@gmail.com'),
-> ('Jessica', 'Anderson', 'jessicaanderson@gmail.com'),
-> ('William', 'Brown', 'williambrown@gmail.com'),
-> ('Olivia', 'White', 'oliviawhite@gmail.com');
Query OK, 5 rows affected (0.00 sec)
Records: 5 Duplicates: 0 Warnings: 0

mysql> INSERT INTO SalesFact (Date, ProductID, CustomerID, SalesAmount)
-> VALUES
-> ('2023-05-10', 1, 1, 600.00),
-> ('2023-06-15', 2, 2, 350.00),
-> ('2023-07-20', 3, 3, 800.00),
-> ('2023-08-05', 4, 4, 420.00),
-> ('2023-09-10', 5, 5, 550.00);
```

```
Command Prompt - mysql -u root -p
mysql> select * from customer;
+-----+-----+-----+-----+
| CustomerID | FirstName | LastName | Email |
+-----+-----+-----+-----+
| 1 | Kiran | Doe | Kirandoe@gmail.com |
| 2 | Dip | Smith | Dipsmith@gmail.com |
| 3 | Robert | son | robertson@gmail.com |
| 4 | Emily | Wilson | emilywilson@gmail.com |
| 5 | Sarah | Miller | sarahmiller@gmail.com |
| 6 | Michael | Davis | michaeldavis@gmail.com |
| 7 | Jessica | Anderson | jessicaanderson@gmail.com |
| 8 | William | Brown | williambrown@gmail.com |
| 9 | Olivia | White | oliviawhite@gmail.com |
+-----+-----+-----+-----+
9 rows in set (0.00 sec)

mysql> select * from product;
+-----+-----+-----+
| ProductID | ProductName | Category |
+-----+-----+-----+
| 1 | Product A | Electronics |
| 2 | Product B | Clothing |
| 3 | Product C | Electronics |
| 4 | Product D | Furniture |
| 5 | Product E | Clothing |
| 6 | Product F | Electronics |
| 7 | Product G | Clothing |
| 8 | Product H | Furniture |
| 9 | Product I | Electronics |
| 10 | Product J | Clothing |
| 11 | Product K | Furniture |
+-----+-----+-----+
11 rows in set (0.00 sec)

mysql> select * from salefact;
ERROR 1146 (42S02): Table 'kiran_2.salefact' doesn't exist
mysql> select * from salesfact;
+-----+-----+-----+-----+-----+
| SaleID | Date | ProductID | CustomerID | SalesAmount |
+-----+-----+-----+-----+-----+
| 1 | 2023-01-05 | 1 | 1 | 500.00 |
| 2 | 2023-02-15 | 2 | 2 | 300.00 |
| 3 | 2023-03-10 | 1 | 3 | 750.00 |
| 4 | 2023-04-20 | 3 | 4 | 400.00 |
| 5 | 2023-05-10 | 1 | 1 | 600.00 |
| 6 | 2023-06-15 | 2 | 2 | 350.00 |
| 7 | 2023-07-20 | 3 | 3 | 800.00 |
| 8 | 2023-08-05 | 4 | 4 | 420.00 |
| 9 | 2023-09-10 | 5 | 5 | 550.00 |
| 10 | 2023-10-15 | 6 | 1 | 700.00 |
+-----+-----+-----+-----+-----+
10 rows in set (0.00 sec)
```


1. Slicing

Operation: Slice the data to view sales for a specific quarter (e.g., Quarter 2 of 2023).

```
Command Prompt - mysql -u root -p
mysql> SELECT
->   T.Date,
->   P.Category,
->   SUM(SF.SalesAmount) AS TotalSales
-> FROM SalesFact SF
-> JOIN Time T ON SF.Date = T.Date
-> JOIN Product P ON SF.ProductID = P.ProductID
-> WHERE T.Quarter = 2 AND T.Year = 2023
-> GROUP BY T.Date, P.Category
-> ORDER BY T.Date, P.Category;
+-----+-----+-----+
| Date      | Category | TotalSales |
+-----+-----+-----+
| 2023-01-05 | Electronics | 500.00 |
| 2023-02-15 | Clothing   | 300.00 |
| 2023-03-10 | Electronics | 750.00 |
| 2023-04-20 | Electronics | 400.00 |
+-----+-----+-----+
4 rows in set (0.00 sec)
```

2. Dicing

Operation: Create a subcube to view sales for Quarter 2 of 2023 for products in the "Electronics" category.

```
Command Prompt - mysql -u root -p
mysql> SELECT
->   T.Date,
->   P.ProductName,
->   SF.SalesAmount
-> FROM SalesFact SF
-> JOIN Time T ON SF.Date = T.Date
-> JOIN Product P ON SF.ProductID = P.ProductID
-> WHERE T.Quarter = 2 AND T.Year = 2023 AND P.Category = 'Electronics';
+-----+-----+-----+
| Date      | ProductName | SalesAmount |
+-----+-----+-----+
| 2023-01-05 | Product A   | 500.00 |
| 2023-03-10 | Product A   | 750.00 |
| 2023-04-20 | Product C   | 400.00 |
+-----+-----+-----+
3 rows in set (0.00 sec)
```

3. Rolling up

Operation: Aggregate sales from monthly to quarterly level for 2023.

```
Command Prompt - mysql -u root -p
mysql> SELECT
  ->     T.Year,
  ->     T.Quarter,
  ->     P.Category,
  ->     SUM(SF.SalesAmount) AS TotalSales
  -> FROM SalesFact SF
  -> JOIN Time T ON SF.Date = T.Date
  -> JOIN Product P ON SF.ProductID = P.ProductID
  -> WHERE T.Year = 2023
  -> GROUP BY T.Year, T.Quarter, P.Category
  -> ORDER BY T.Year, T.Quarter, P.Category;
+-----+-----+-----+-----+
| Year | Quarter | Category | TotalSales |
+-----+-----+-----+-----+
| 2023 |      2 | Clothing |      300.00 |
| 2023 |      2 | Electronics |     1650.00 |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)

mysql> SELECT
```

4. Drill-Up:

Operation: Navigate from quarterly sales data to monthly sales data for Quarter 2 of 2023.

```
Command Prompt - mysql -u root -p
mysql> SELECT
  ->     T.Date,
  ->     P.Category,
  ->     SUM(SF.SalesAmount) AS TotalSales
  -> FROM SalesFact SF
  -> JOIN Time T ON SF.Date = T.Date
  -> JOIN Product P ON SF.ProductID = P.ProductID
  -> WHERE T.Quarter = 2 AND T.Year = 2023
  -> GROUP BY T.Date, P.Category
  -> ORDER BY T.Date, P.Category;
+-----+-----+-----+
| Date       | Category | TotalSales |
+-----+-----+-----+
| 2023-01-05 | Electronics |      500.00 |
| 2023-02-15 | Clothing   |      300.00 |
| 2023-03-10 | Electronics |      750.00 |
| 2023-04-20 | Electronics |      400.00 |
+-----+-----+-----+
4 rows in set (0.00 sec)
```

5. Pivot:

Operation: Pivot the data to view total sales by product category for each quarter in 2023.

```
Command Prompt - mysql -u root -p
mysql> SELECT
->   P.Category,
->   SUM(CASE WHEN T.Quarter = 1 THEN SF.SalesAmount ELSE 0 END) AS Q1_Sales,
->   SUM(CASE WHEN T.Quarter = 2 THEN SF.SalesAmount ELSE 0 END) AS Q2_Sales,
->   SUM(CASE WHEN T.Quarter = 3 THEN SF.SalesAmount ELSE 0 END) AS Q3_Sales,
->   SUM(CASE WHEN T.Quarter = 4 THEN SF.SalesAmount ELSE 0 END) AS Q4_Sales
-> FROM SalesFact SF
-> JOIN Time T ON SF.Date = T.Date
-> JOIN Product P ON SF.ProductID = P.ProductID
-> WHERE T.Year = 2023
-> GROUP BY P.Category
-> ORDER BY P.Category;
+-----+-----+-----+-----+
| Category | Q1_Sales | Q2_Sales | Q3_Sales | Q4_Sales |
+-----+-----+-----+-----+
| Clothing |      0.00 |    300.00 |      0.00 |      0.00 |
| Electronics |      0.00 |   1650.00 |      0.00 |      0.00 |
+-----+-----+-----+-----+
2 rows in set (0.00 sec)
```

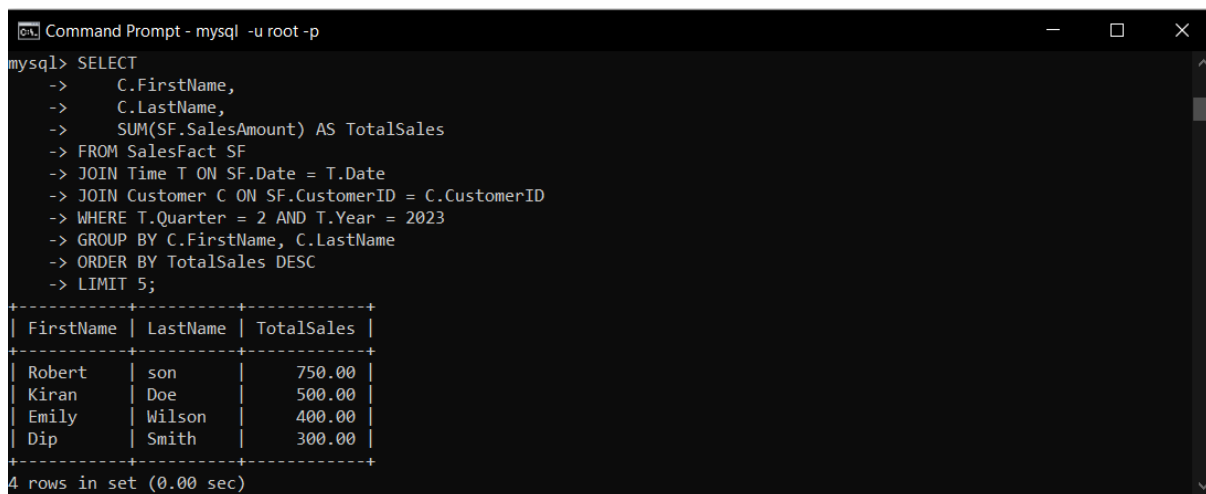
6. Drill-Down:

Operation: Navigate from quarterly sales data to monthly sales data for Quarter 2 of 2023.

```
Command Prompt - mysql -u root -p
mysql> SELECT
->   T.Date,
->   P.Category,
->   SUM(SF.SalesAmount) AS TotalSales
-> FROM SalesFact SF
-> JOIN Time T ON SF.Date = T.Date
-> JOIN Product P ON SF.ProductID = P.ProductID
-> WHERE T.Month BETWEEN 4 AND 6 AND T.Year = 2023
-> GROUP BY T.Date, P.Category
-> ORDER BY T.Date, P.Category;
Empty set (0.00 sec)
```

8. Top-N Analysis:

Operation: Find the top 5 customers with the highest sales in Quarter 2 of 2023.



```
Command Prompt - mysql -u root -p
mysql> SELECT
->     C.FirstName,
->     C.LastName,
->     SUM(SF.SalesAmount) AS TotalSales
-> FROM SalesFact SF
-> JOIN Time T ON SF.Date = T.Date
-> JOIN Customer C ON SF.CustomerID = C.CustomerID
-> WHERE T.Quarter = 2 AND T.Year = 2023
-> GROUP BY C.FirstName, C.LastName
-> ORDER BY TotalSales DESC
-> LIMIT 5;
+-----+-----+-----+
| FirstName | LastName | TotalSales |
+-----+-----+-----+
| Robert    | son      | 750.00     |
| Kiran     | Doe      | 500.00     |
| Emily     | Wilson   | 400.00     |
| Dip       | Smith    | 300.00     |
+-----+-----+-----+
4 rows in set (0.00 sec)
```

Conclusion:

In conclusion, Online Analytical Processing (OLAP) operations are essential tools for dissecting and interpreting multidimensional data. These operations, including slice, dice, roll-up, drill-down, pivot, and Top-N analysis, provide a versatile framework for users to navigate, summarize, and manipulate data according to their analytical needs. Whether it's narrowing the focus of data, drilling deeper for detailed insights, summarizing information for higher-level trends, or even reorienting data perspectives, OLAP operations empower users to derive valuable insights and make informed decisions from complex datasets. Understanding and skillfully employing these OLAP operations is vital for business intelligence and data analysis in various domains.