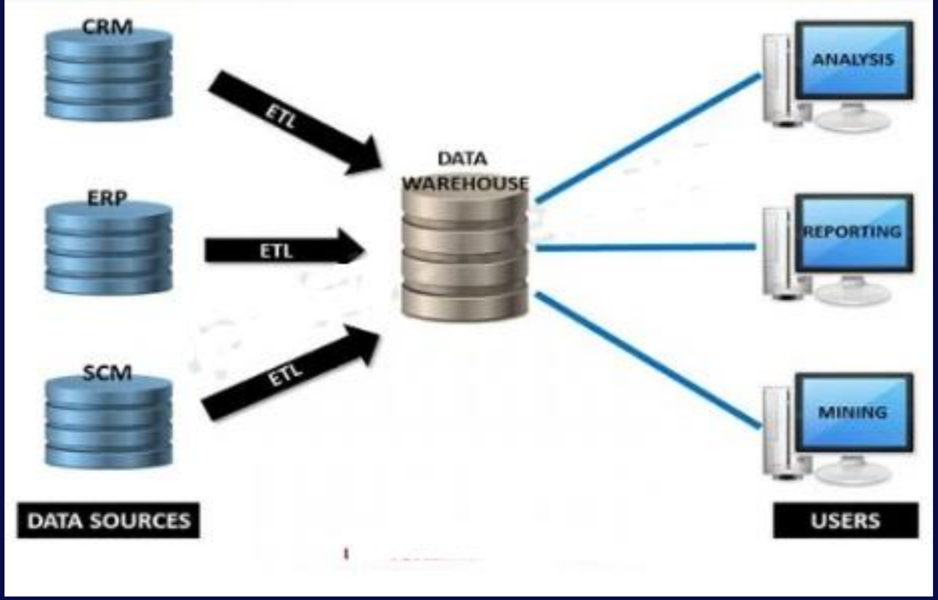
# Chapter 3
# Data Warehousing and Online Analytical Processing

# Content:

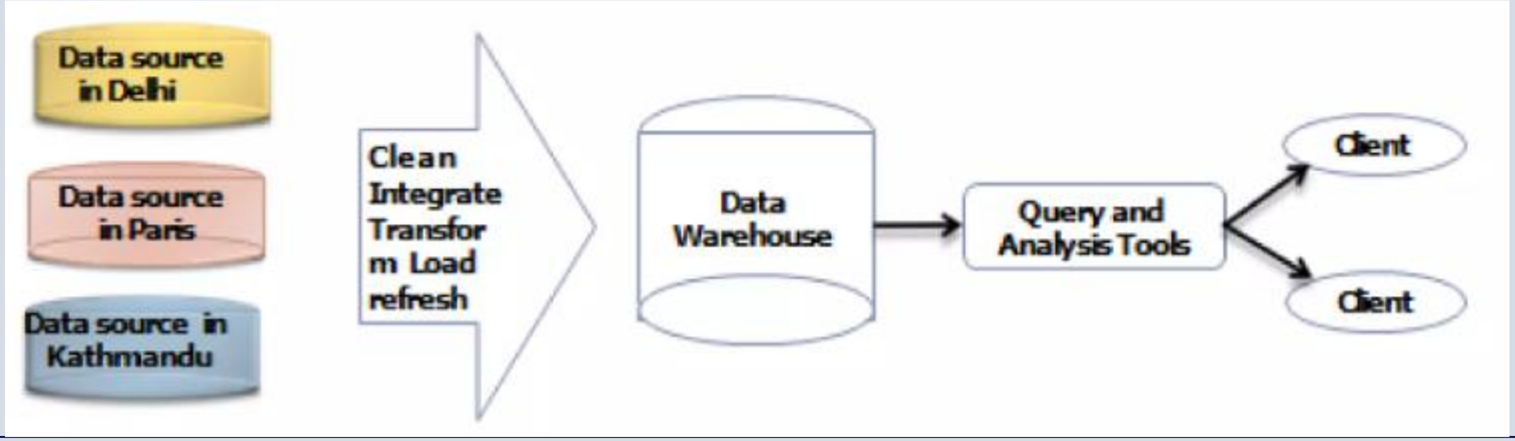
1. Data Warehouse: Basic Concepts and features
2. Characteristics of Data Warehouse
3. Extract, Transform and Load
4. Multitiered Architecture
5. Conceptual Modeling of Data Warehouse
6. Data Warehouse Modeling: Data Cube, Data Mart and OLAP

# 3.1 Data Warehouse Basic Concept



# Data warehouse

- A data warehouse **is a large collection of business data** used to help an organization
- A Data Warehouse (DW**)** **is a repository of information collected from multiple sources, stored under a unified schema.**
- It is **not used for daily operations and transaction processing** but used **for making decisions.**
- A Data Warehouse (DW) is a relational database that is designed for query and analysis rather than transaction processing.
- A Data Warehouse provides integrated, enterprise-wide, historical data and focuses on providing support for decision-makers for data modeling and analysis.



# Definition

- The popular definition of the data warehouse by WH Inmon:

“ A Data Warehouse is a :

**subject-oriented**,

**integrated** ,

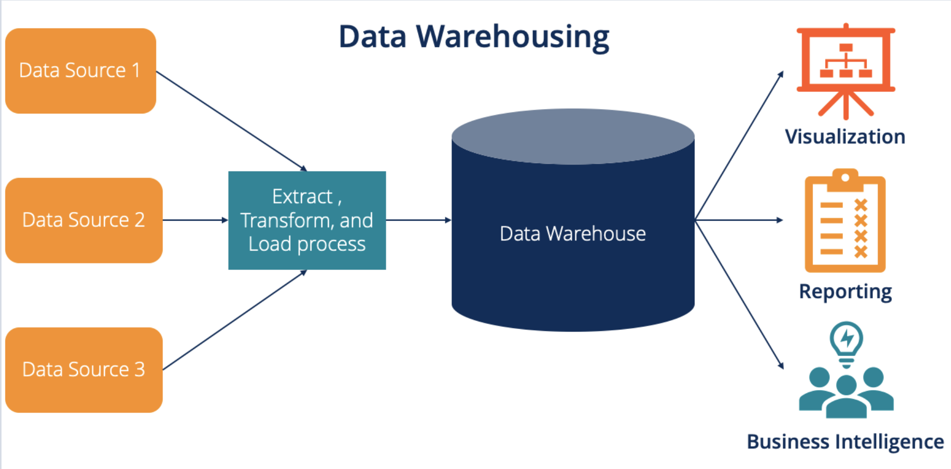
**time-variant**, **and**

**non-volatile**

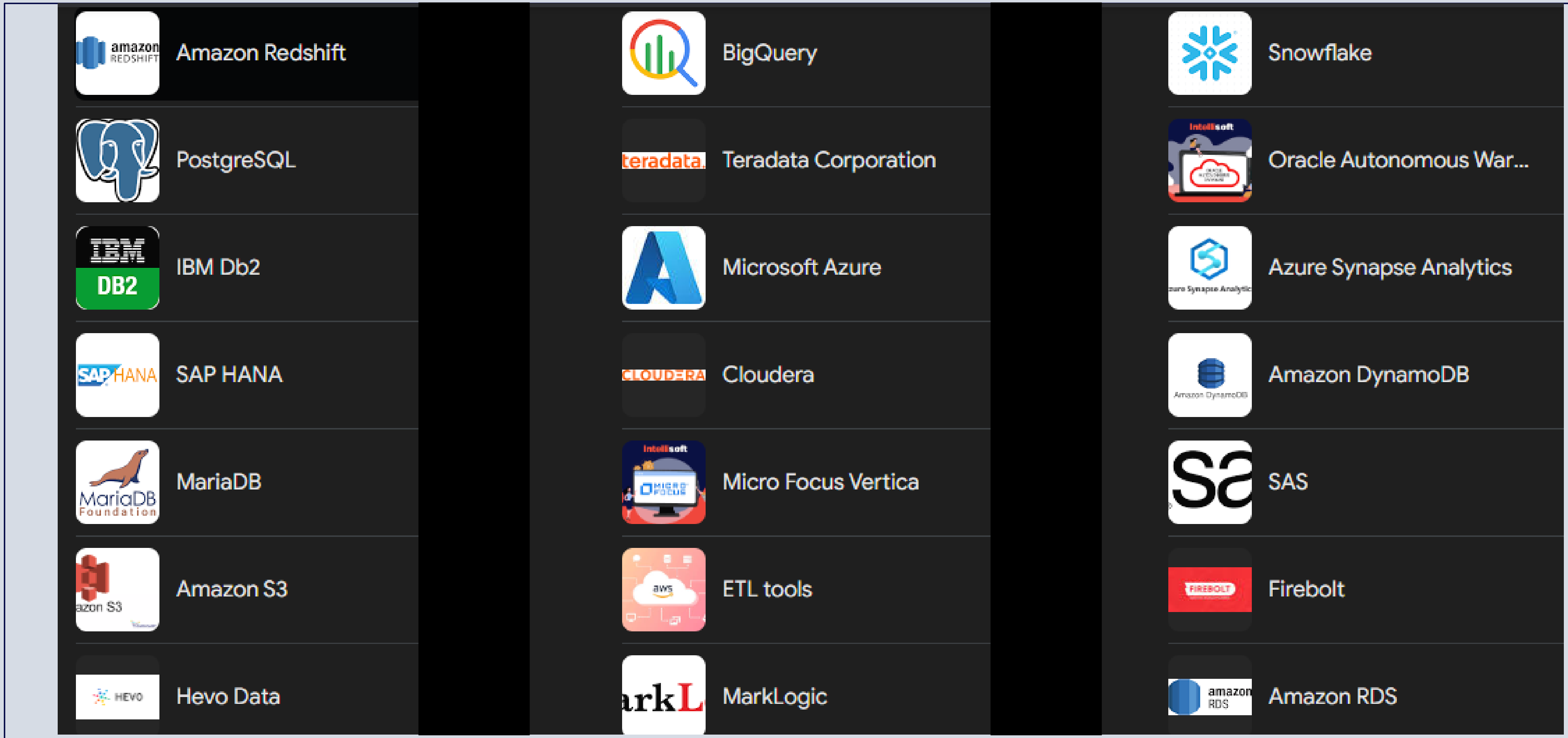
collection of data in support of management’s decision-making process”

# Data Warehousing

- **Data Warehousing** is the process of **collecting, storing, and managing data** from different sources in a **central database** for analysis and reporting.
- It helps businesses make **better decisions** by organizing data in a structured way.
- Data warehousing involves data scraping, collecting, storing, data wrangling, and managing enormous data from various sources.
- It is a centralized repository that offers a single source of truth for consistent and reliable data analysis to support business decision-making.
- This data undergoes comprehensive data cleansing, is structured, transformed, and organized to provide a unified view of your organization's operations and performance.



# Tools for Data warehousing



Amazon Redshift

PostgreSQL

IBM Db2

SAP HANA

MariaDB

Amazon S3

Hevo Data

BigQuery

Teradata Corporation

Microsoft Azure

Cloudera

Micro Focus Vertica

ETL tools

MarkLogic

Snowflake

Oracle Autonomous War...

Azure Synapse Analytics

Amazon DynamoDB

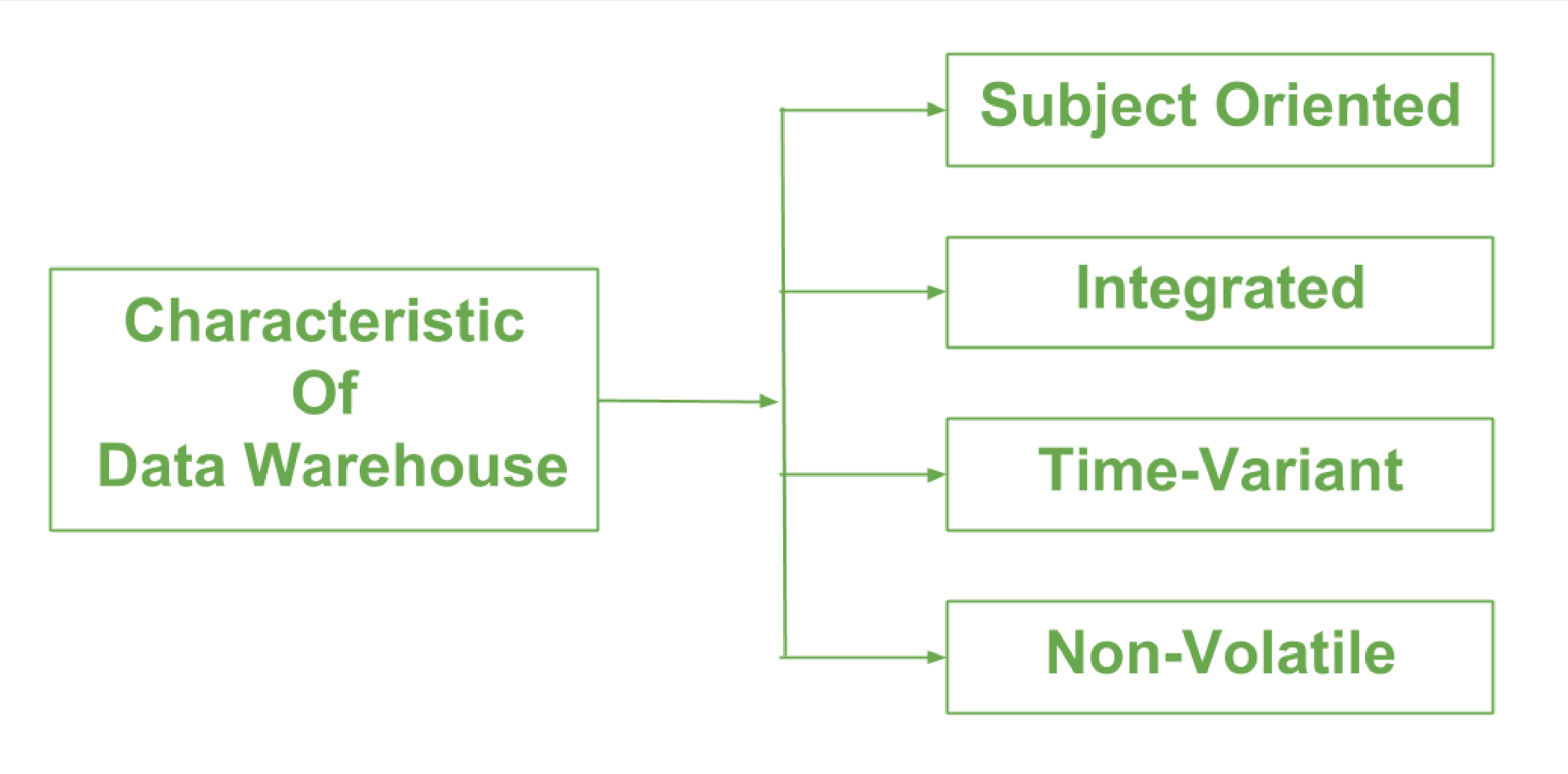
SAS

Firebolt

Amazon RDS

# 3.2 Characteristics of Data Warehouse

# Characteristics of Data Warehouse



Characteristic Of Data Warehouse

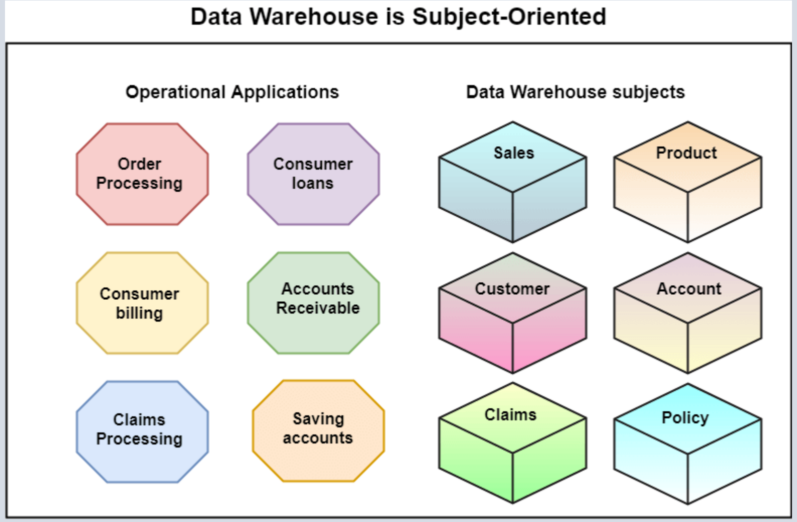
- Subject Oriented
- Integrated
- Time-Variant
- Non-Volatile

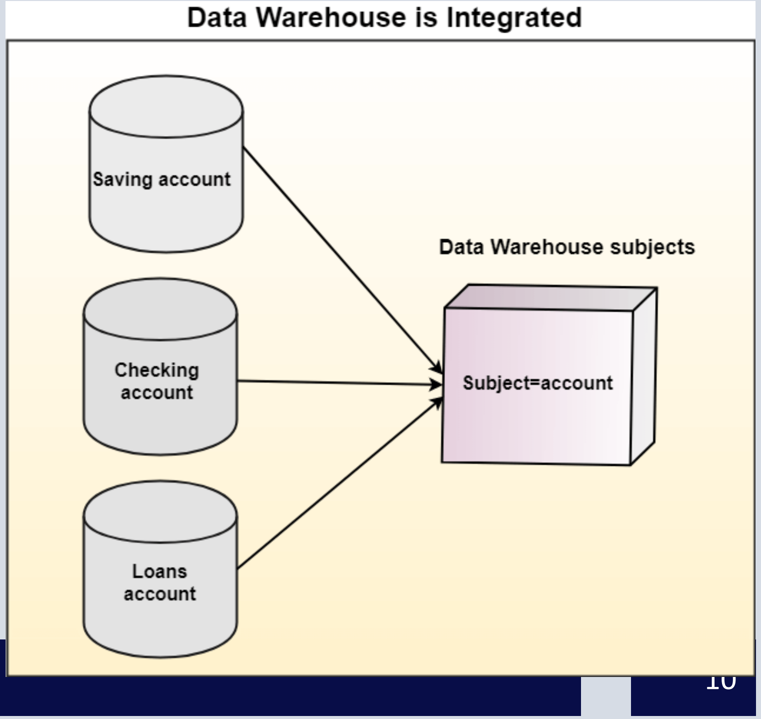
# 1. Subject-oriented

- A data warehouse is a subject-oriented approach.
- Because, it **provides information on a specific topic** rather than information about an organization’s ongoing operations.
- **Provides views on subjects** such as **customer**, **product**, or **sales**, instead of the global organization's ongoing operations.

# 2. Integrated

- A data warehouse **integrates various heterogeneous data sources** like RDBMS, flat files, and online transaction records.
- It **requires** performing **data cleaning** and **integration** during data warehousing to ensure consistency in naming conventions, attributes types, etc., among different data sources.



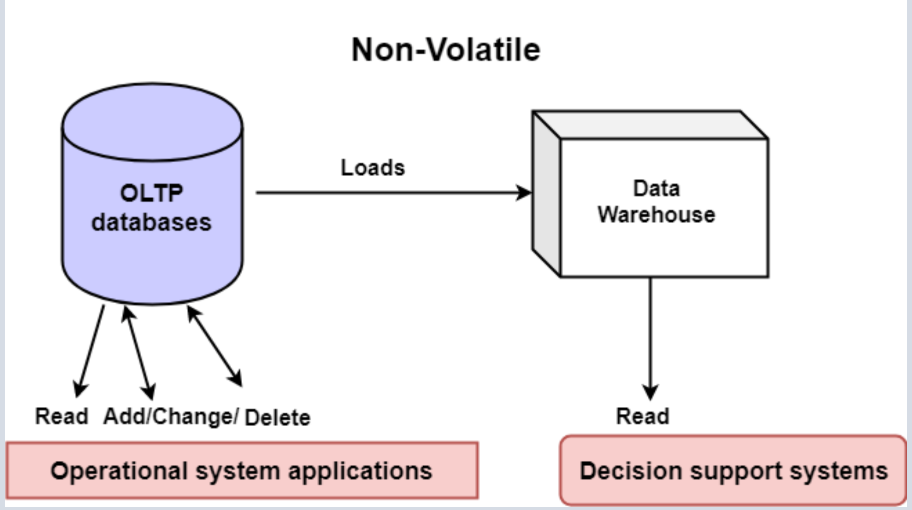


### 3. Time-variant

- The **time horizon** for the data warehouse **is significantly longer** than that of operational systems.
    - ***Operational dataset:*** current value data (60 to 90 days)
    - ***Data warehouse data:*** provide information from a historical perspective (e.g. past 5-10 years)

### 4. Non-volatile

- Data Warehouse is **relatively static in nature.**
- Data Warehouse data is loaded (usually in mass) and accessed, but is not uploaded (in the general sense).
- Data warehouse data is a sophisticated series of snapshots, each taken at one moment in time. The effect created by the series of snapshots is that the data warehouse has a historical sequence of activities and events



# 3.3 Architecture of Data Warehouse

# Data Warehouse Architecture

- A Data Warehouse is a system that combine data from multiple sources, organizes it under a single architecture, and helps organizations make better decisions.
- It simplifies data handling, storage, and reporting, making analysis more efficient.
- Data Warehouse Architecture uses a structured framework to manage and store data effectively.
- There are two common approaches to constructing a data warehouse:
  - **Top-Down Approach:** This method starts with designing the overall data warehouse architecture first and then creating individual data marts.
  - **Bottom-Up Approach:** In this method, data marts are built first to meet specific business needs, and later integrated into a central data warehouse.

# Components of Data warehouse Architecture

- A data warehouse architecture consists of several key components that work together to store, manage, and analyze data.
  1. **Data Source:** These include transactional databases (OLTP), external data sources, flat files, etc.
  2. **Staging Area:** The staging area is a temporary space where raw data from external sources is validated and prepared before entering the data warehouse.
     - **Extract**: Data is pulled from different sources.
     - **Transform:** Data is cleaned and converted into a standardized format.
     - **Load:** Data is stored in the data warehouse.
  3. **Data warehouse:** A central repository where processed data is stored, typically in **fact tables** (quantitative data) and **dimension tables** (descriptive data).
  4. **Data Marts:** Subsets of a data warehouse focused on specific departments (e.g., marketing, finance).
  5. **Metadata:** Data about data, such as source details, transformation rules, and table relationships.
  6. **Data mining and BI Tools:** Used for data mining, reporting, dashboards, and data visualization (e.g., Power BI, Tableau).

# The architecture of a Data Warehouse System

- A typical data warehouse system has three main phases:
  1. **Data Acquisition**
     - Relevant data collection
     - *Recovering:* Transforming into the data warehouse model from existing models
     - *Loading:* Cleaning and loading in the DW
  2. **Storage**
     - *Metadata:* data of data
     - *Data marts*: Small data sets
  3. **Data Extraction**
     - Includes query and Analysis tools like:
       - ❑ Query report, SQL, OLAP tools, Data mining

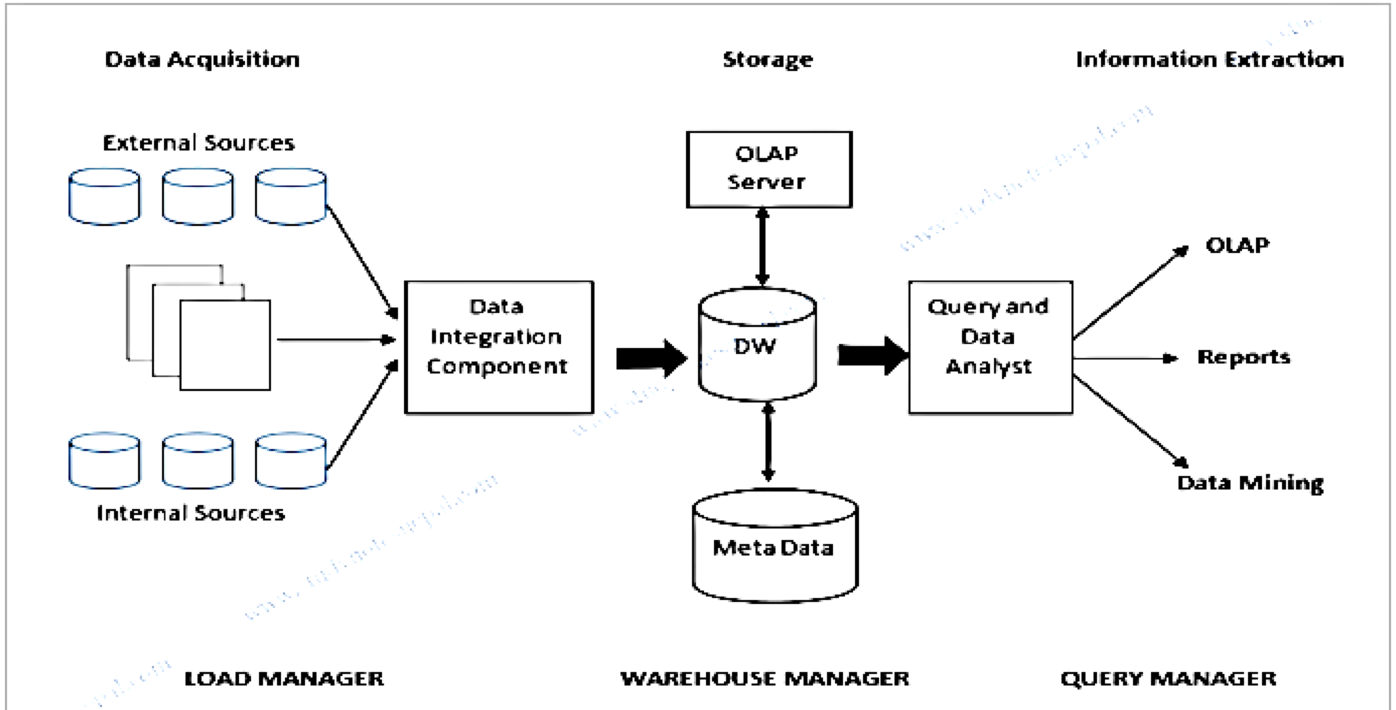
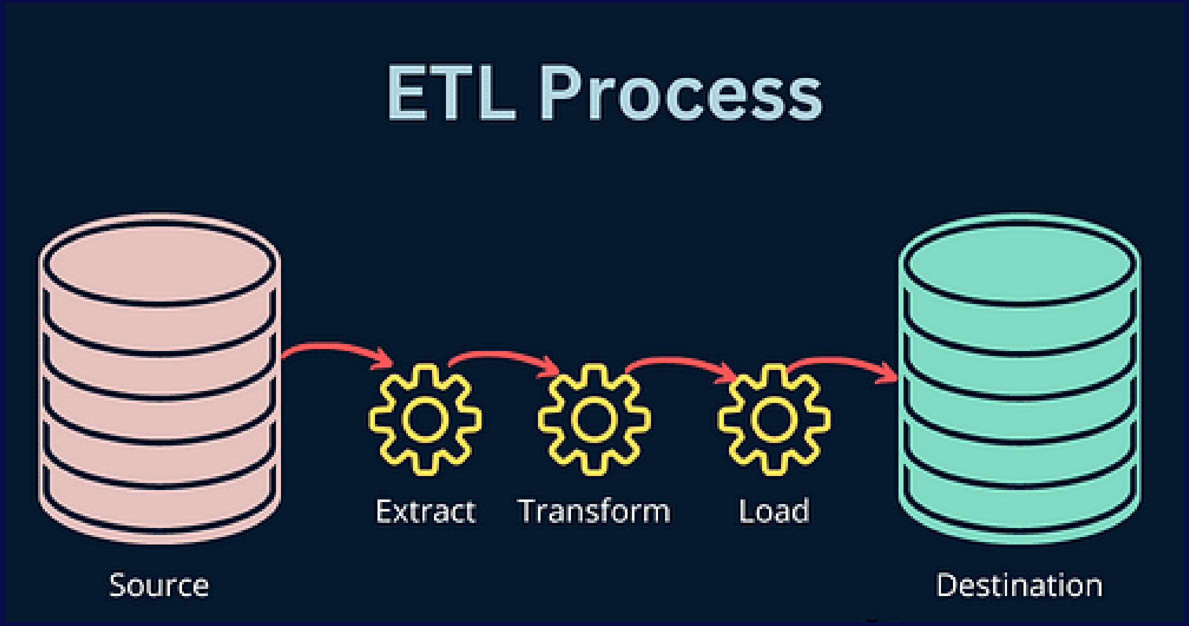


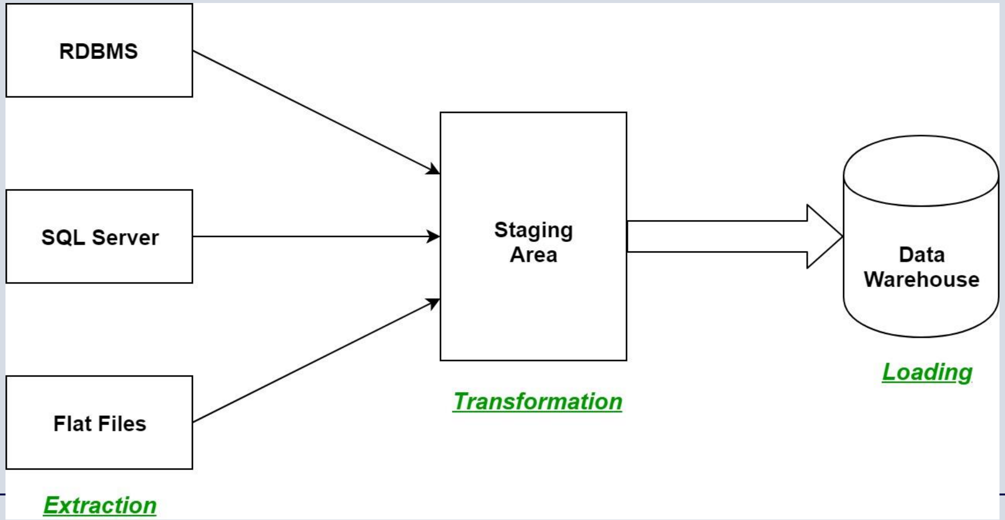
Figure: Data Warehouse Architecture

# 3.4 ETL (Extract, Transform, Load)



# ETL process

- ETL stands for Extract, Transform, Load
- It is **a process used in data warehousing:**
  - **to extract data from various sources,**
  - **transform it into a format suitable for loading into a data warehouse,** and
  - **then load** it into the warehouse.
- The process of ETL can be broken down into the following three stages:
  1. Data Extraction
  2. Data Transformation
  3. Data Loading



# i) Data Extraction

- The first step of the ETL process is extraction.
- In this step, **data from various source systems is extracted** which can be in various formats like relational databases, No SQL, XML, and flat files into the staging area.
- It is important to **extract the data from various source systems and store it into the staging area first** and not directly into the data warehouse because the extracted data is in various formats and can be corrupted also.
- Hence, **loading it directly into the data warehouse may damage it** and rollback will be much more difficult.
- This phase involves retrieving raw data from multiple sources such as:
  - **Databases** (e.g., MySQL, Oracle, SQL Server)
  - **Files** (CSV, XML, JSON, Excel)
  - **APIs** (Web services, cloud applications)
  - **Streaming Data** (IoT, sensors, logs)

# ii) Data Transformation

- The second step of the ETL process is transformation. Raw data is cleaned, enriched, and formatted to match the data warehouse schema.
- In this step, a set of rules or functions are applied on the **extracted data to convert it into a single standard format**.
- **Common Transformations:**
  a) **Data Cleaning:** Handling missing values, duplicate records, incorrect formats.
  b) **Data Integration:** Merging data from multiple sources (e.g., sales + marketing data).
  c) **Data Aggregation:** Summarizing data (e.g., daily sales → monthly sales).
  d) **Data Standardization:** Converting units, date formats, etc.
  e) **Business Rules Application:** Applying specific logic (e.g., calculating customer lifetime value).
- **Example:** Converting inconsistent date formats ( MM/DD/YYYY → YYYY-MM-DD).

# iii) Data Loading

- The third and final step of the ETL process is loading.
- In this step, the **transformed data is finally loaded into the data warehouse.**
- Sometimes the data is updated by loading into the data warehouse very frequently and sometimes it is done after longer but regular intervals.
- The rate and period of loading solely depends on the requirements and varies from system to system.
- **Loading Strategies:**
  - **Full Load:** All data is loaded at once (first-time load).
  - **Incremental Load:** Only new or updated records are added (common in real-time processing).
  - **Delta Load:** Only changes (insert/update/delete) since the last load are applied.

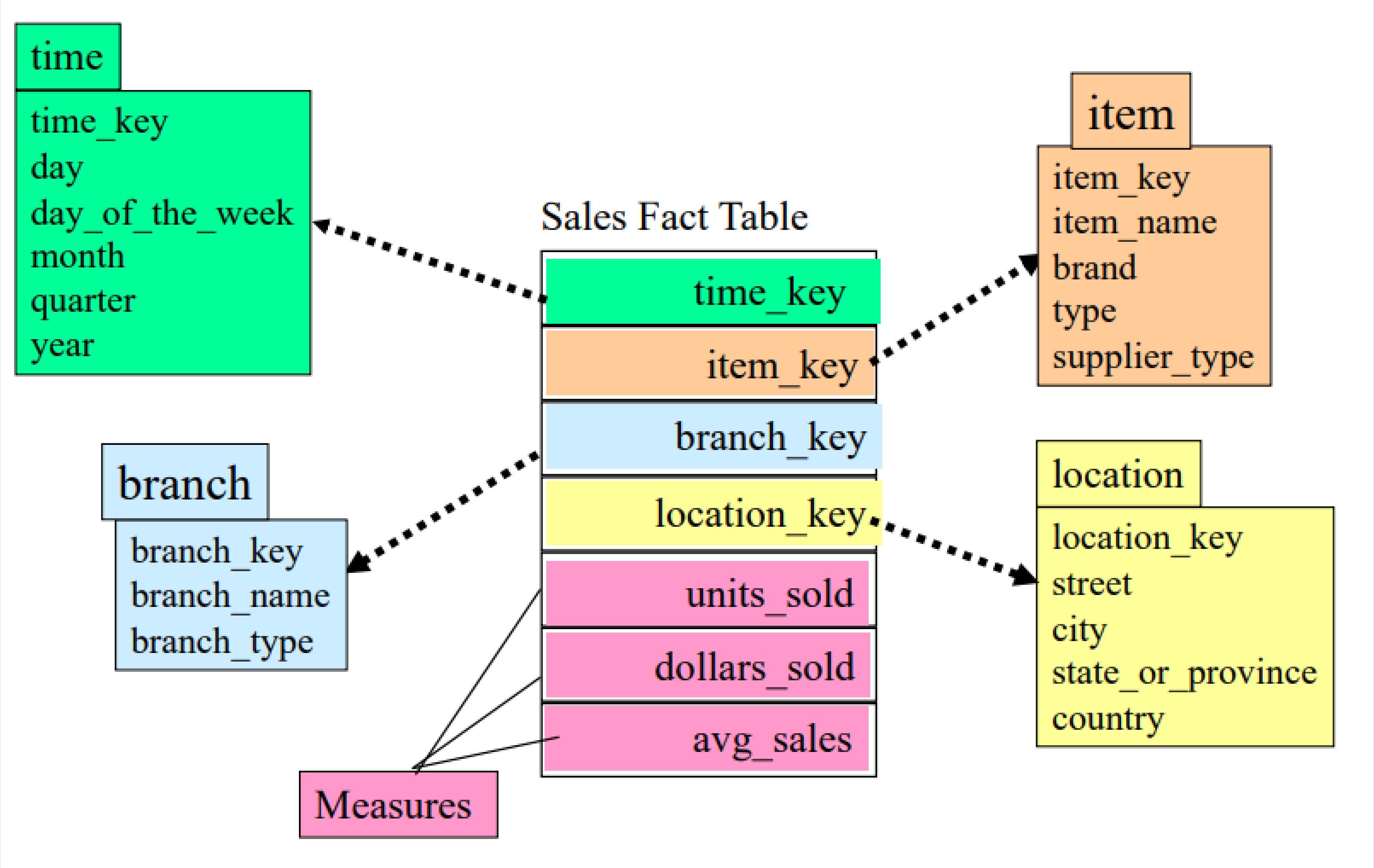
# 3.5 Conceptual Modeling of Data Warehouses

# Conceptual Modeling of Data Warehouse

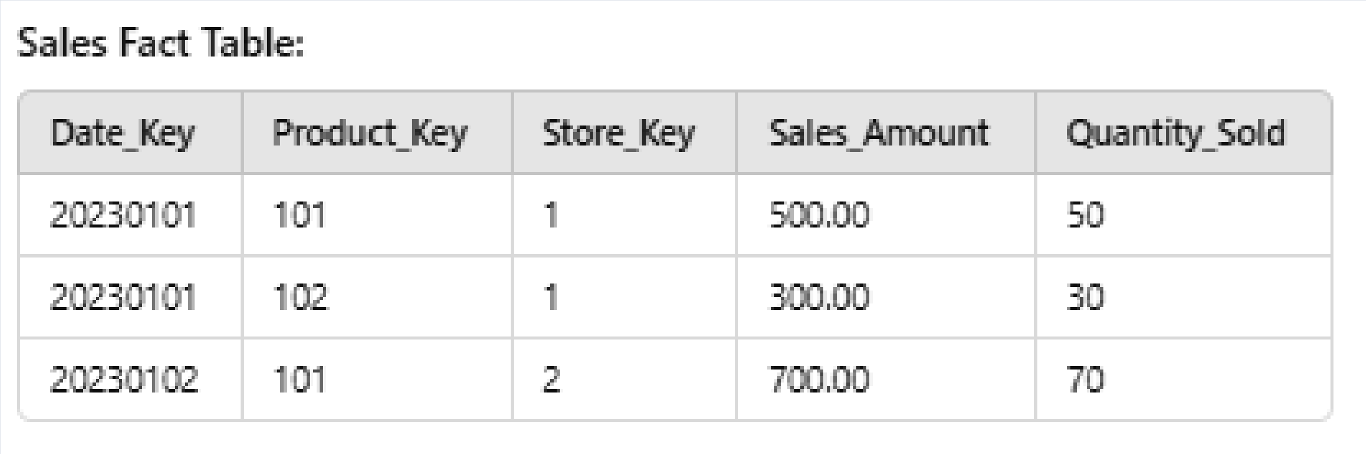
- Modeling data warehouses: dimensions & measures
  - Star schema: A fact table in the middle connected to a set of dimension tables
  - Snowflake schema: A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake
  - Fact constellations: Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation

Star schema:

- A fact table in the middle connected to a set of dimension tables
- A **fact table** is a central table that stores quantitative data for analysis and reporting.
- It contains **facts** (measurable, numerical values) related to a business process.
- It also consists of Foreign keys.



- Example of a Fact Table

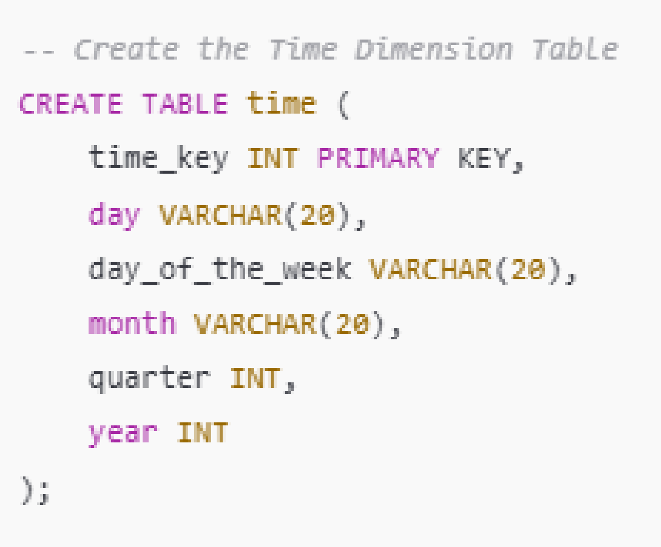


Sales Fact Table:

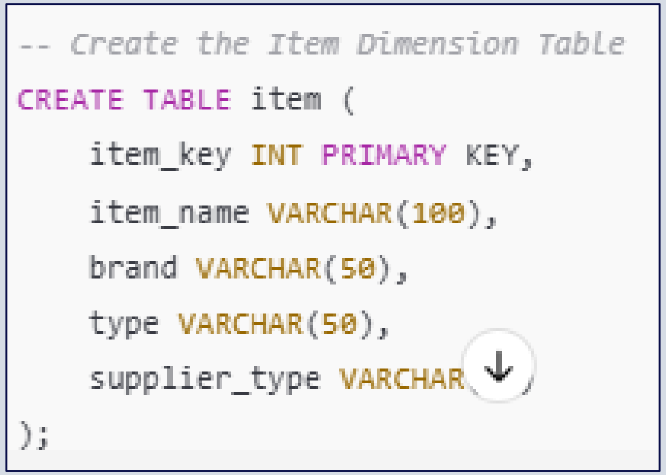
| Date_Key | Product_Key | Store_Key | Sales_Amount | Quantity_Sold |
|---|---|---|---|---|
| 20230101 | 101 | 1 | 500.00 | 50 |
| 20230101 | 102 | 1 | 300.00 | 30 |
| 20230102 | 101 | 2 | 700.00 | 70 |

- **Foreign Keys**:
  - Date_Key links to a **Date Dimension Table**.
  - Product_Key links to a **Product Dimension Table**.
  - Store_Key links to a **Store Dimension Table**.
- **Measures (Facts)**: Sales_Amount and Quantity_Sold.

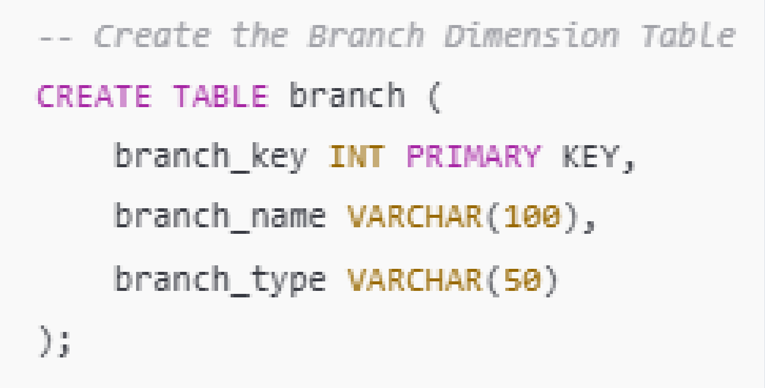
# Code: creating tables



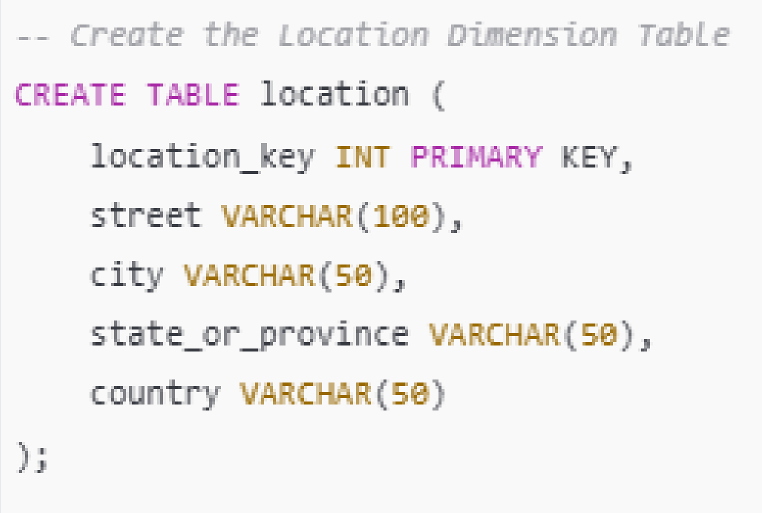
```sql
-- Create the Time Dimension Table
CREATE TABLE time (
    time_key INT PRIMARY KEY,
    day VARCHAR(20),
    day_of_the_week VARCHAR(20),
    month VARCHAR(20),
    quarter INT,
    year INT
);
```



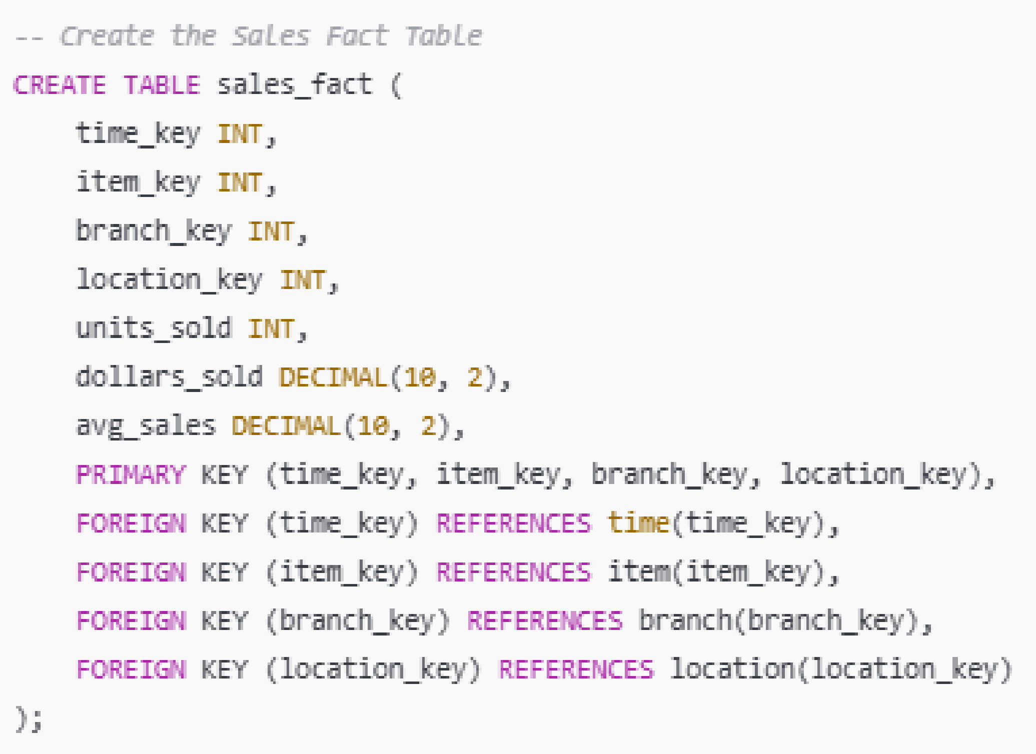
```sql
-- Create the Item Dimension Table
CREATE TABLE item (
    item_key INT PRIMARY KEY,
    item_name VARCHAR(100),
    brand VARCHAR(50),
    type VARCHAR(50),
    supplier_type VARCHAR
);
```



```sql
-- Create the Branch Dimension Table
CREATE TABLE branch (
    branch_key INT PRIMARY KEY,
    branch_name VARCHAR(100),
    branch_type VARCHAR(50)
);
```

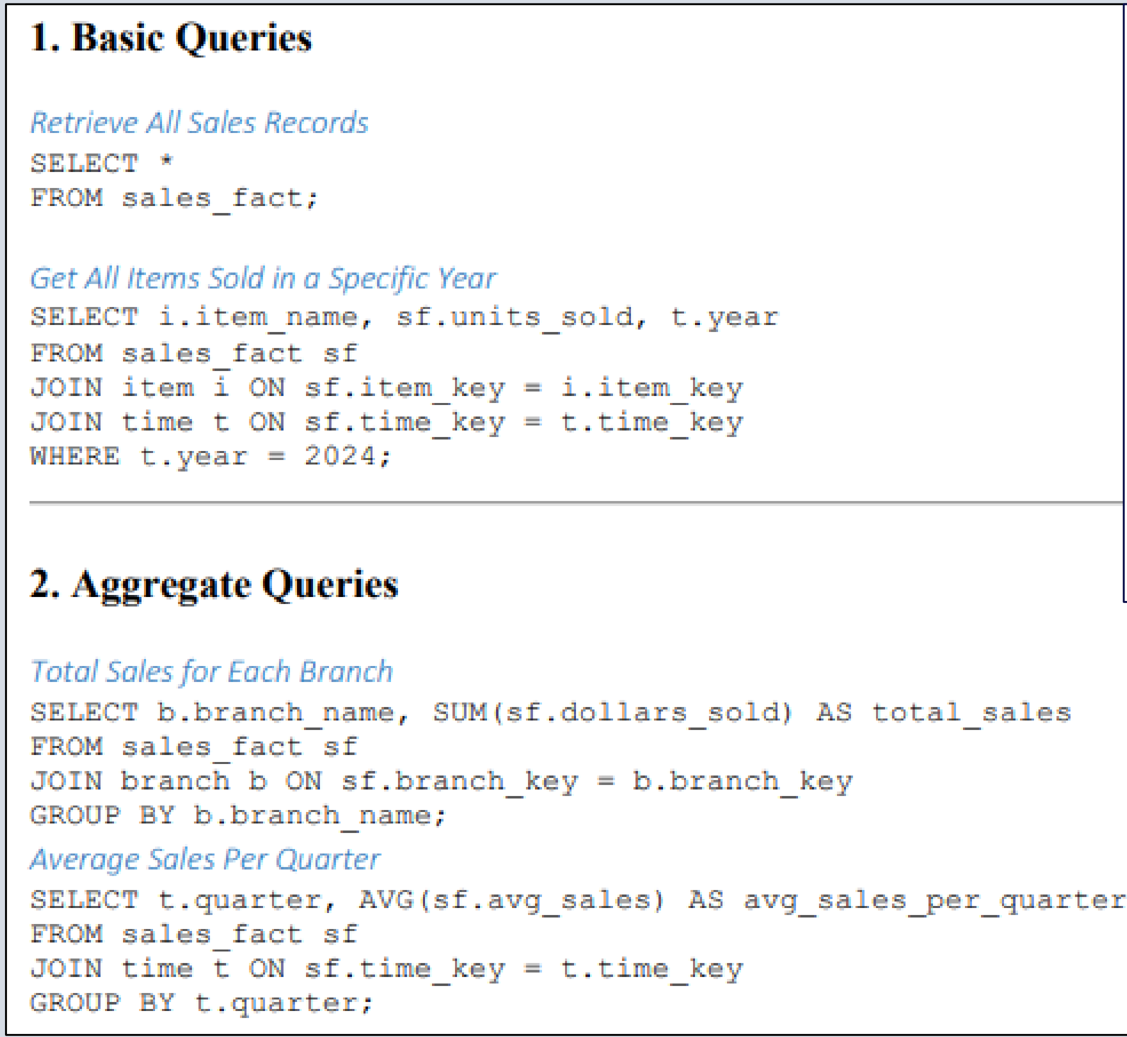


```sql
-- Create the Location Dimension Table
CREATE TABLE location (
    location_key INT PRIMARY KEY,
    street VARCHAR(100),
    city VARCHAR(50),
    state_or_province VARCHAR(50),
    country VARCHAR(50)
);
```



```sql
-- Create the Sales Fact Table
CREATE TABLE sales_fact (
    time_key INT,
    item_key INT,
    branch_key INT,
    location_key INT,
    units_sold INT,
    dollars_sold DECIMAL(10, 2),
    avg_sales DECIMAL(10, 2),
    PRIMARY KEY (time_key, item_key, branch_key, location_key),
    FOREIGN KEY (time_key) REFERENCES time(time_key),
    FOREIGN KEY (item_key) REFERENCES item(item_key),
    FOREIGN KEY (branch_key) REFERENCES branch(branch_key),
    FOREIGN KEY (location_key) REFERENCES location(location_key)
);
```

# Basic Queries



## 1. Basic Queries

*Retrieve All Sales Records*

```
SELECT *
FROM sales_fact;
```

*Get All Items Sold in a Specific Year*

```
SELECT i.item_name, sf.units_sold, t.year
FROM sales_fact sf
JOIN item i ON sf.item_key = i.item_key
JOIN time t ON sf.time_key = t.time_key
WHERE t.year = 2024;
```

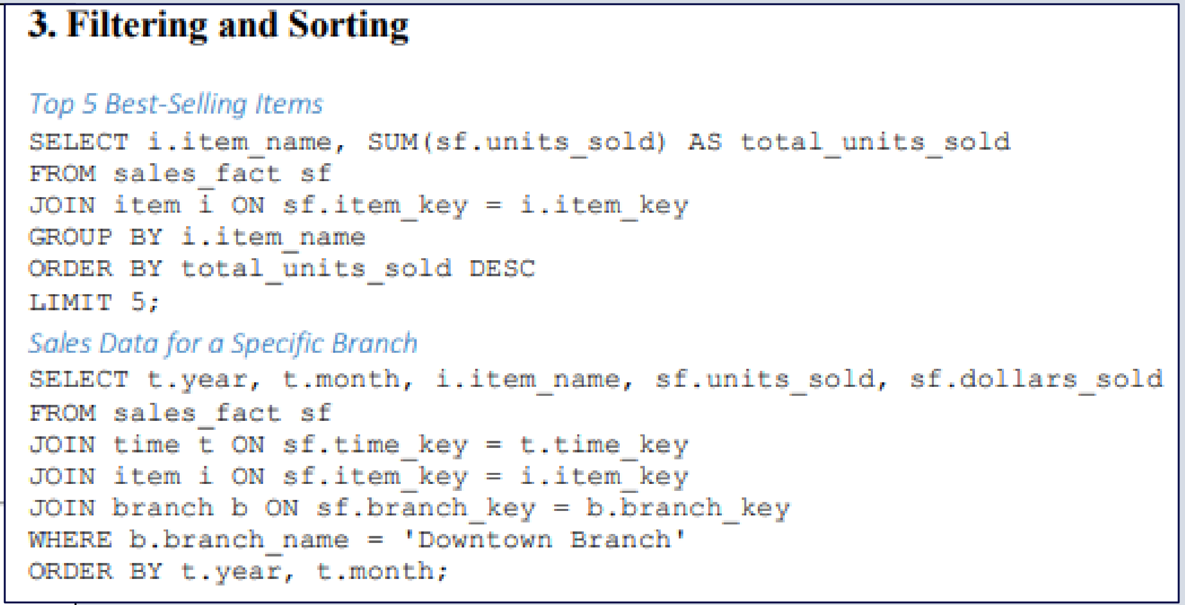
## 2. Aggregate Queries

*Total Sales for Each Branch*

```
SELECT b.branch_name, SUM(sf.dollars_sold) AS total_sales
FROM sales_fact sf
JOIN branch b ON sf.branch_key = b.branch_key
GROUP BY b.branch_name;
```

*Average Sales Per Quarter*

```
SELECT t.quarter, AVG(sf.avg_sales) AS avg_sales_per_quarter
FROM sales_fact sf
JOIN time t ON sf.time_key = t.time_key
GROUP BY t.quarter;
```



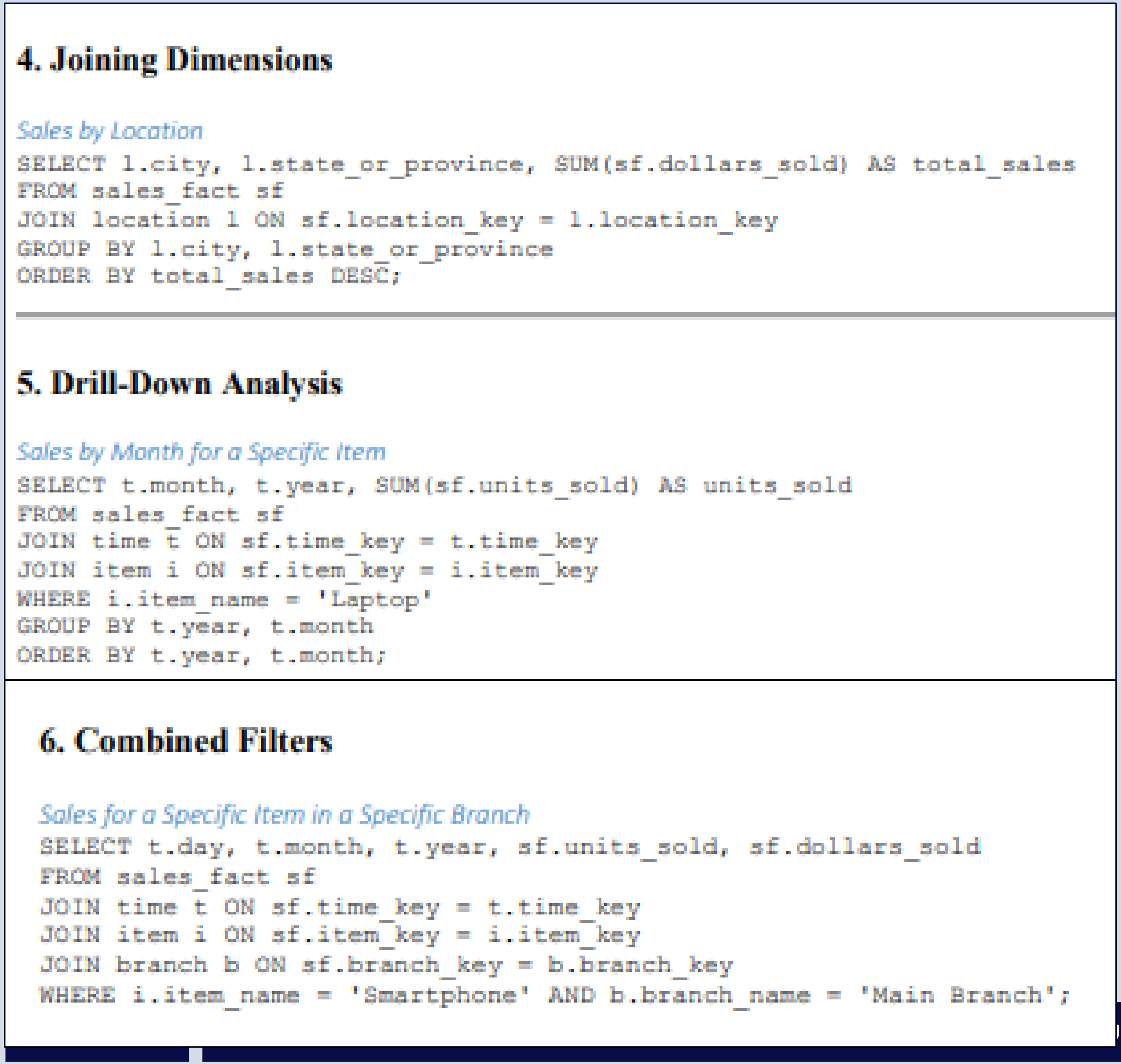
## 3. Filtering and Sorting

*Top 5 Best-Selling Items*

```
SELECT i.item_name, SUM(sf.units_sold) AS total_units_sold
FROM sales_fact sf
JOIN item i ON sf.item_key = i.item_key
GROUP BY i.item_name
ORDER BY total_units_sold DESC
LIMIT 5;
```

*Sales Data for a Specific Branch*

```
SELECT t.year, t.month, i.item_name, sf.units_sold, sf.dollars_sold
FROM sales_fact sf
JOIN time t ON sf.time_key = t.time_key
JOIN item i ON sf.item_key = i.item_key
JOIN branch b ON sf.branch_key = b.branch_key
WHERE b.branch_name = 'Downtown Branch'
ORDER BY t.year, t.month;
```



## 4. Joining Dimensions

*Sales by Location*

```
SELECT l.city, l.state_or_province, SUM(sf.dollars_sold) AS total_sales
FROM sales_fact sf
JOIN location l ON sf.location_key = l.location_key
GROUP BY l.city, l.state_or_province
ORDER BY total_sales DESC;
```

## 5. Drill-Down Analysis

*Sales by Month for a Specific Item*

```
SELECT t.month, t.year, SUM(sf.units_sold) AS units_sold
FROM sales_fact sf
JOIN time t ON sf.time_key = t.time_key
JOIN item i ON sf.item_key = i.item_key
WHERE i.item_name = 'Laptop'
GROUP BY t.year, t.month
ORDER BY t.year, t.month;
```

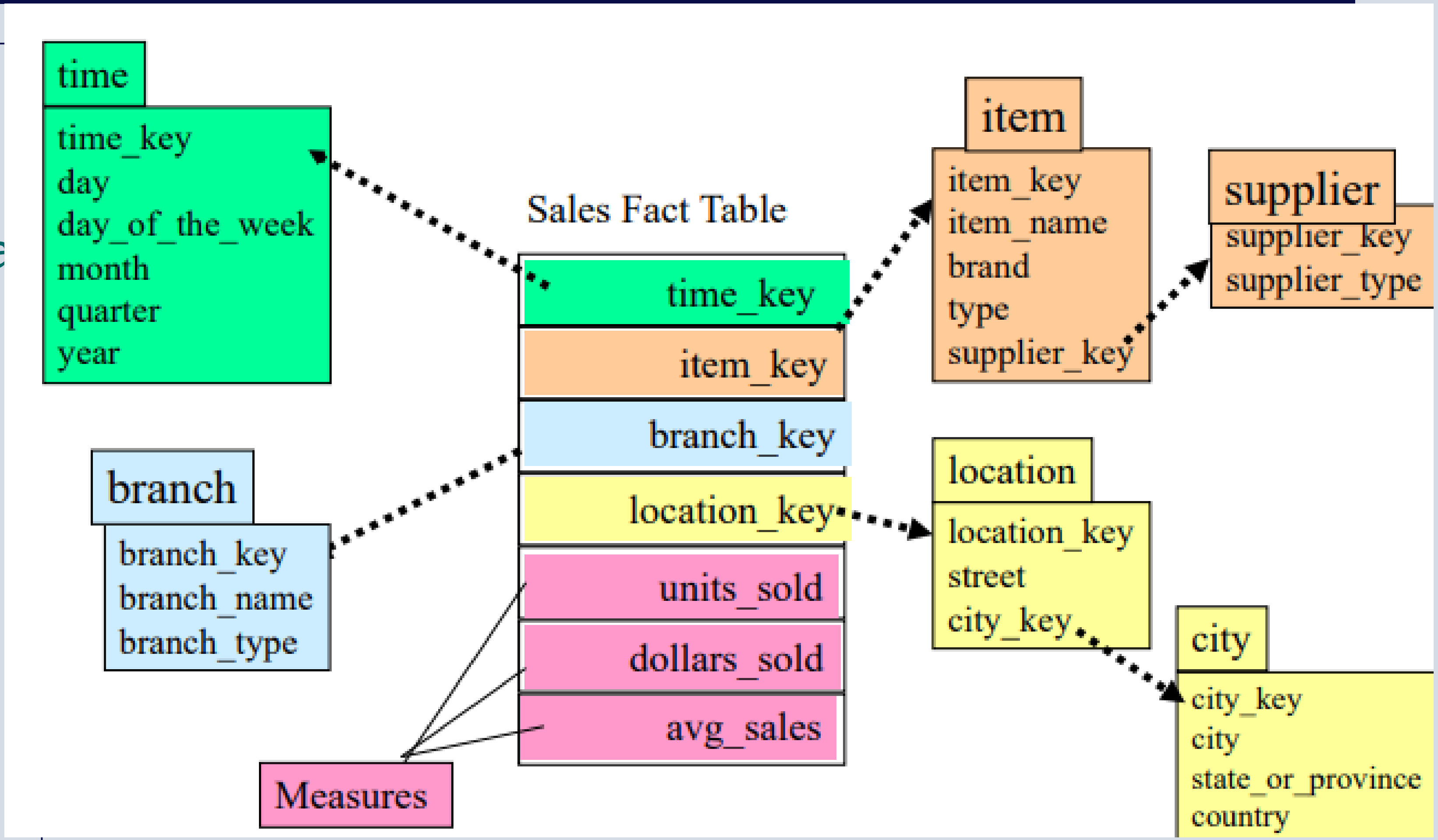
## 6. Combined Filters

*Sales for a Specific Item in a Specific Branch*

```
SELECT t.day, t.month, t.year, sf.units_sold, sf.dollars_sold
FROM sales_fact sf
JOIN time t ON sf.time_key = t.time_key
JOIN item i ON sf.item_key = i.item_key
JOIN branch b ON sf.branch_key = b.branch_key
WHERE i.item_name = 'Smartphone' AND b.branch_name = 'Main Branch';
```

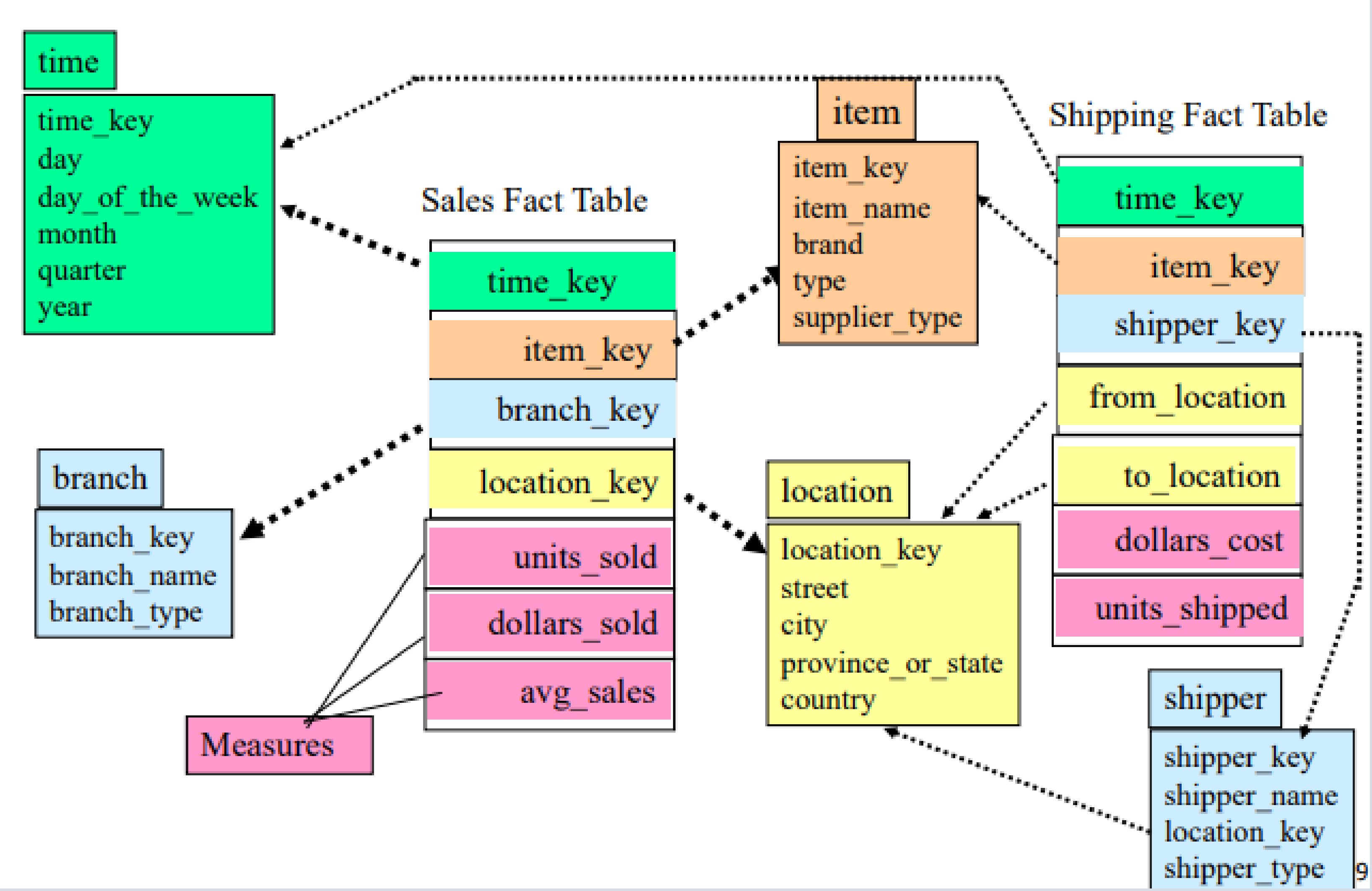
## Snowflake schema:

- A refinement of star schema where some dimensional hierarchy is normalized into a set of smaller dimension tables, forming a shape similar to snowflake



## Fact Constellations:

- Multiple fact tables share dimension tables, viewed as a collection of stars, therefore called galaxy schema or fact constellation



# code

# 3.6 Data Warehouse Modeling

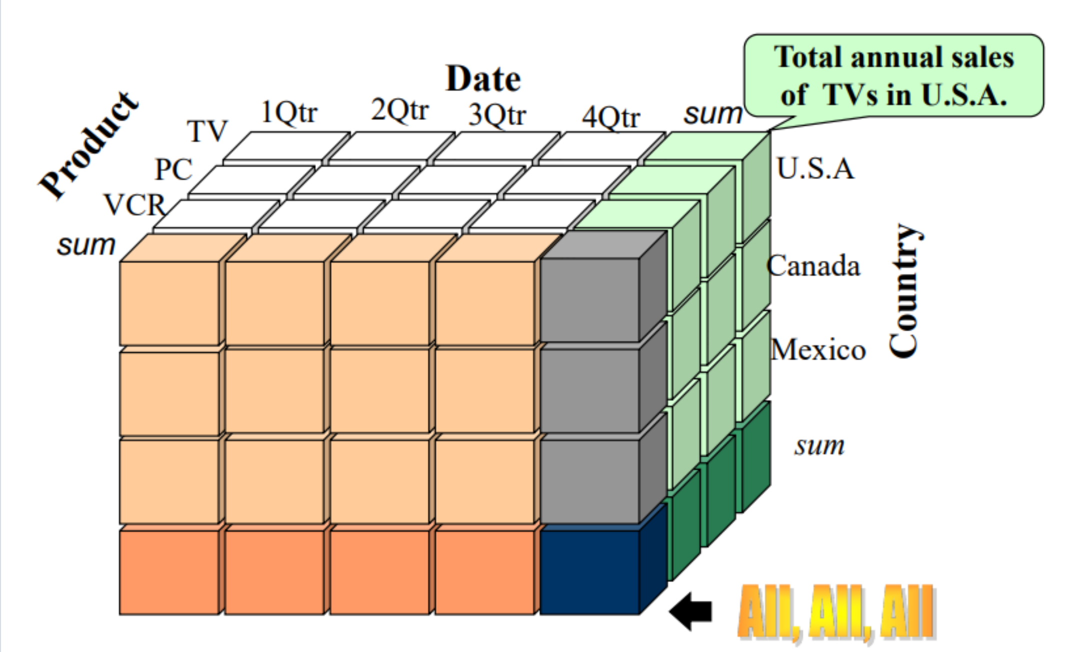
Data cube, Data Mart and OLAP

# 3.6.1 Multidimensional View and Data cube

- A multidimensional data model **views data in the form of a data cube.**
- A data cube is a fundamental concept in OLAP (Online Analytical Processing) and multidimensional database systems.
- It **represents data in a multidimensional format**, allowing for efficient and flexible analysis along multiple dimensions.
- A data cube allows data to be modeled and viewed in multiple dimensions. It is defined by **dimensions** and **facts**.
  - **Dimensions** represent the categorical attributes or perspectives along which data can be analyzed. Examples of dimensions include time, geography, product, customer, and sales channel.
  - **Facts**, also known as measures, represent the numerical or quantitative data that are being analyzed. Examples of facts include sales revenue, quantity sold, profit, and expenses.

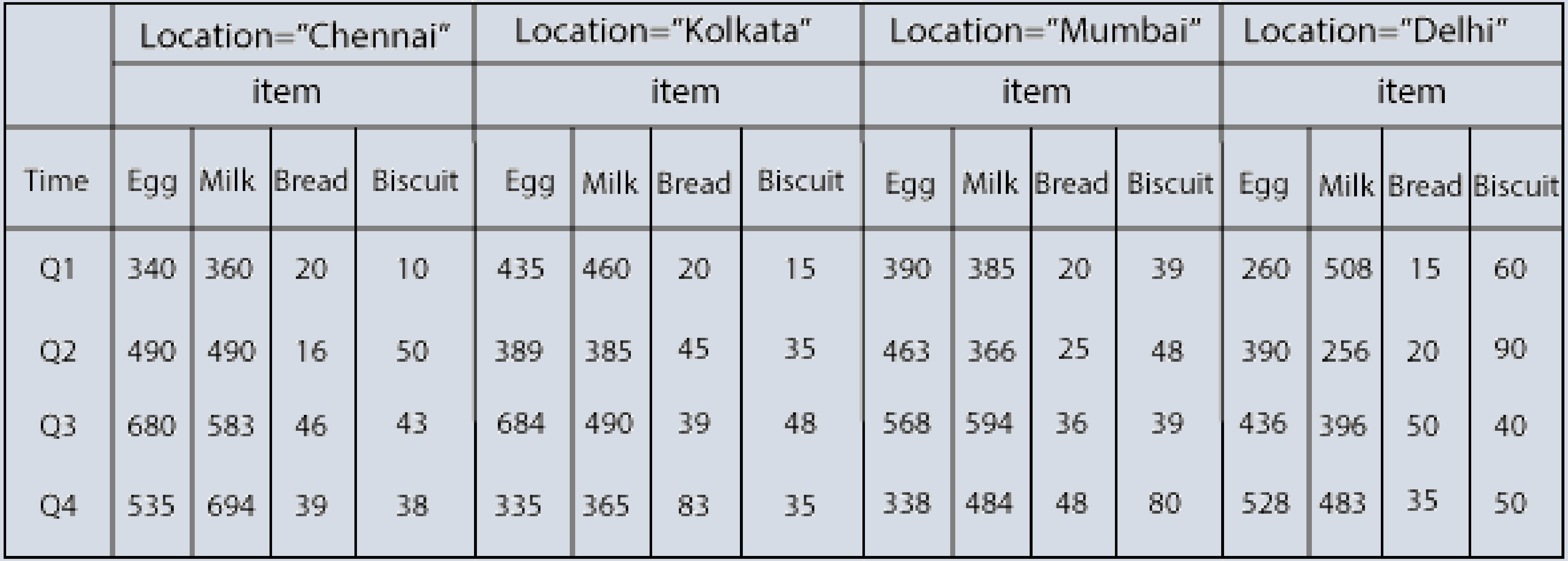
# Data Cube

- A data cube is a fundamental concept which **represents data in a multidimensional format**, allowing for efficient and flexible analysis along multiple dimensions.
- It is defined by **dimensions** and **facts**.
  - **Dimensions** represent the categorical attributes or perspectives along which data can be analyzed. Examples of dimensions include time, geography, product, customer, and sales channel.
  - **Facts**, also known as measures, represent the numerical or quantitative data that are being analyzed. Examples of facts include sales revenue, quantity sold, profit, and expenses.

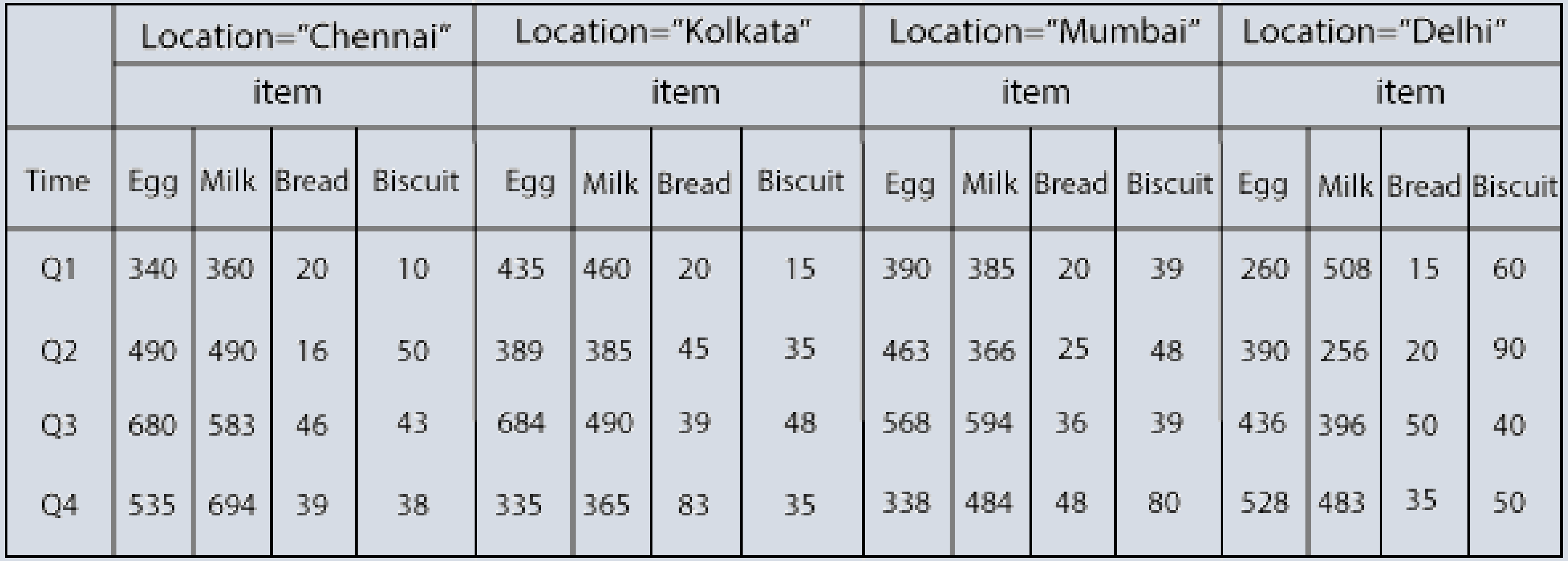


# Data Cube Implementation

- For example, suppose the data according to time and item, as well as the location is considered for the cities Chennai, Kolkata, Mumbai, and Delhi. These 3D data are shown in the table. The 3D data of the table are represented as a series of 2D tables.

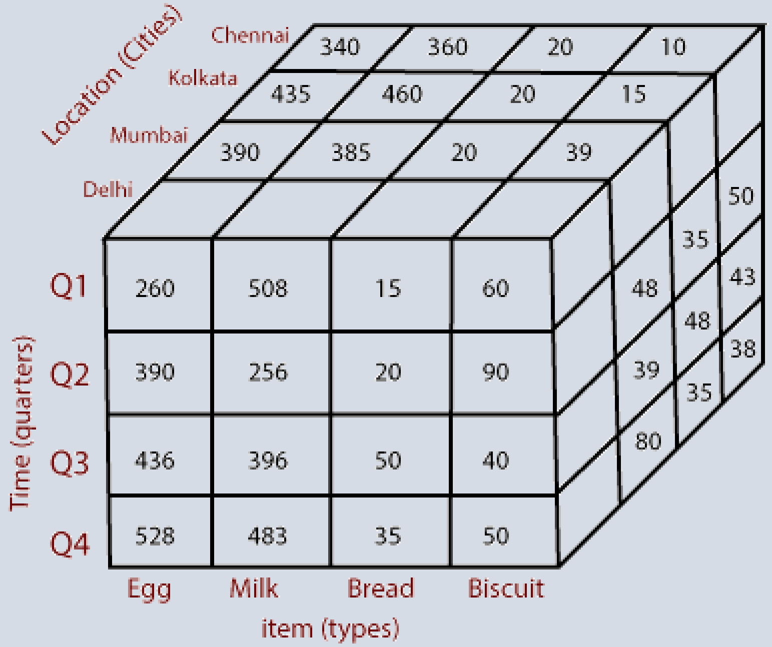


| | Location="Chennai" | | | | Location="Kolkata" | | | | Location="Mumbai" | | | | Location="Delhi" | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | item | | | | item | | | | item | | | | item | | | |
| Time | Egg | Milk | Bread | Biscuit | Egg | Milk | Bread | Biscuit | Egg | Milk | Bread | Biscuit | Egg | Milk | Bread | Biscuit |
| Q1 | 340 | 360 | 20 | 10 | 435 | 460 | 20 | 15 | 390 | 385 | 20 | 39 | 260 | 508 | 15 | 60 |
| Q2 | 490 | 490 | 16 | 50 | 389 | 385 | 45 | 35 | 463 | 366 | 25 | 48 | 390 | 256 | 20 | 90 |
| Q3 | 680 | 583 | 46 | 43 | 684 | 490 | 39 | 48 | 568 | 594 | 36 | 39 | 436 | 396 | 50 | 40 |
| Q4 | 535 | 694 | 39 | 38 | 335 | 365 | 83 | 35 | 338 | 484 | 48 | 80 | 528 | 483 | 35 | 50 |



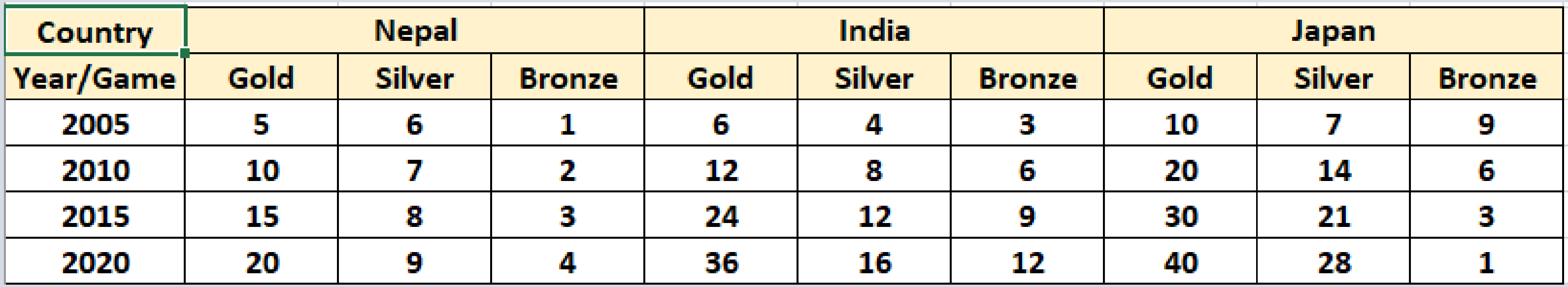
|  | Location="Chennai" |  |  |  | Location="Kolkata" |  |  |  | Location="Mumbai" |  |  |  | Location="Delhi" |  |  |  |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|  | item |  |  |  | item |  |  |  | item |  |  |  | item |  |  |  |
| Time | Egg | Milk | Bread | Biscuit | Egg | Milk | Bread | Biscuit | Egg | Milk | Bread | Biscuit | Egg | Milk | Bread | Biscuit |
| Q1 | 340 | 360 | 20 | 10 | 435 | 460 | 20 | 15 | 390 | 385 | 20 | 39 | 260 | 508 | 15 | 60 |
| Q2 | 490 | 490 | 16 | 50 | 389 | 385 | 45 | 35 | 463 | 366 | 25 | 48 | 390 | 256 | 20 | 90 |
| Q3 | 680 | 583 | 46 | 43 | 684 | 490 | 39 | 48 | 568 | 594 | 36 | 39 | 436 | 396 | 50 | 40 |
| Q4 | 535 | 694 | 39 | 38 | 335 | 365 | 83 | 35 | 338 | 484 | 48 | 80 | 528 | 483 | 35 | 50 |

- Conceptually, it may also be represented by the same data in the form of a 3D data cube, as shown in fig:



# Assignment:

- Implement a data cube for the below dataset:



| Country | Nepal | | | India | | | Japan | | |
|---|---|---|---|---|---|---|---|---|---|
| Year/Game | Gold | Silver | Bronze | Gold | Silver | Bronze | Gold | Silver | Bronze |
| 2005 | 5 | 6 | 1 | 6 | 4 | 3 | 10 | 7 | 9 |
| 2010 | 10 | 7 | 2 | 12 | 8 | 6 | 20 | 14 | 6 |
| 2015 | 15 | 8 | 3 | 24 | 12 | 9 | 30 | 21 | 3 |
| 2020 | 20 | 9 | 4 | 36 | 16 | 12 | 40 | 28 | 1 |

# 3.6.2 Data marts

- A **data mart** is a **small, focused part of a data warehouse** that stores data for a specific department or team, like Sales, Marketing, or Finance. It helps them quickly find and analyze the information they need without searching through a large data warehouse.
- Imagine a **large supermarket** that collects data on **sales, customers, employees, and inventory**. Instead of giving every department access to all the data, separate **data marts** can be created:
  - **Sales Data Mart** – Contains only sales data for sales managers.
  - **Inventory Data Mart** – Stores stock details for warehouse teams.
  - **HR Data Mart** – Includes employee details for the HR department.
- Each department gets only the data they need, making reports faster and easier.
- **Benefits of Data marts:**
  1. **Faster Query Performance** – Queries run quickly since they work on a smaller dataset.
  2. **Cost-Effective** – Requires fewer resources compared to a full data warehouse.
  3. **Easy to Maintain** – Less complex and easier to manage for specific departments.
  4. **Better Security** – Limits access to only relevant users, reducing security risks.

# Types of Data marts

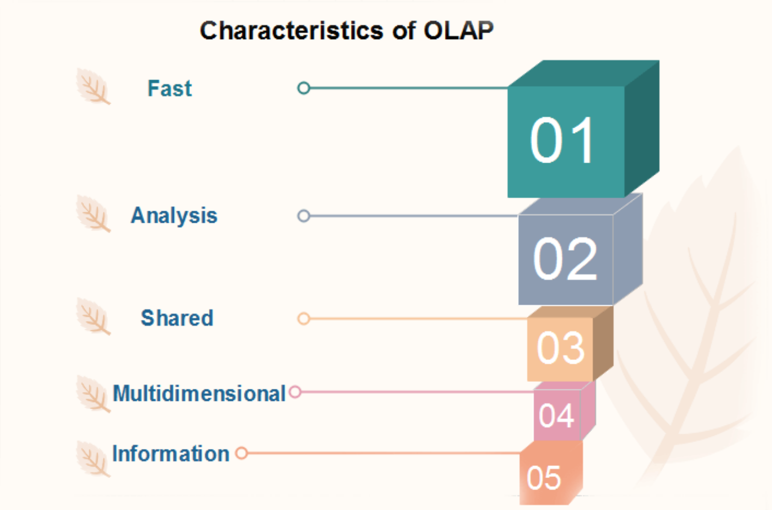
1. **Dependent Data Mart**
   - Extracts data **directly from a central data warehouse**.
   - Ensures consistency across the organization.
   - Example: A **Sales Data Mart** created from an enterprise data warehouse.
2. **Independent Data Mart**
   - Built directly from **operational systems** or external sources without using a central data warehouse.
   - Suitable for small teams or departments that need quick analysis.
   - Example: A **Marketing Data Mart** pulling data from CRM and online ad platforms.
3. **Hybrid Data Mart**
   - Combines data from both **a data warehouse and external sources**.
   - Provides flexibility by integrating additional data sources when needed.
   - Example: A **Customer Insights Data Mart** combining internal sales data and external customer feedback.

# 3.6.3 Online Analytical Processing (OLAP)

- **OLAP (Online Analytical Processing)** is a technology used in a data warehouse that helps users **analyze large amounts of data quickly**.
- It allows businesses to **view data from different angles** and make better decisions.
- OLAP systems organize data into a multidimensional data model, typically in the form of **data cubes.**
- It is a computing method that organizes data into multidimensional structures to facilitate users to get information and analyze data from multiple viewpoints.
- **Why is OLAP Used?**
  1. **Fast Analysis** – OLAP helps users quickly find trends and patterns in large data sets.
  2. **Multi-Dimensional View** – You can look at data in different ways, such as **sales by region, product, and time**.
  3. **Better Decision-Making** – Helps businesses make **informed** choices based on data insights.

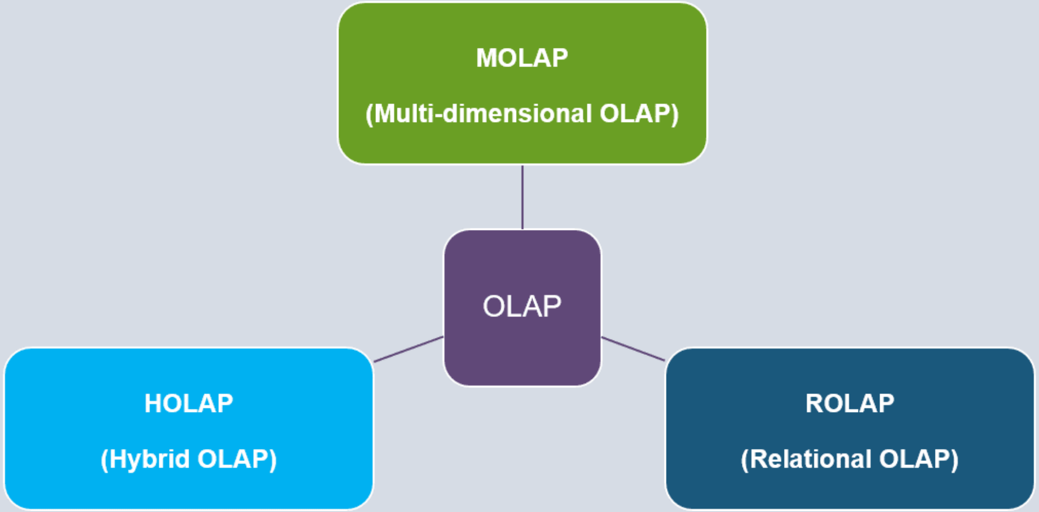
# FASMI Characteristics of OLAP systems

1. **Fast:** Provides most feedback to the client very fast.
2. **Analytic:** An OLAP system must provide rich analytic functionality to cope with any relevant queries for the application and the user.
3. **Shared:** An OLAP allows concurrent access of data by multiple users.
4. **Multidimensional** – The system should support a multidimensional analysis of the data..
5. **Information:** The system should be able to hold all the data needed by the applications.



# Types of OLAP

1. ROLAP
2. MOLAP
3. HOLAP



## 1. ROLAP

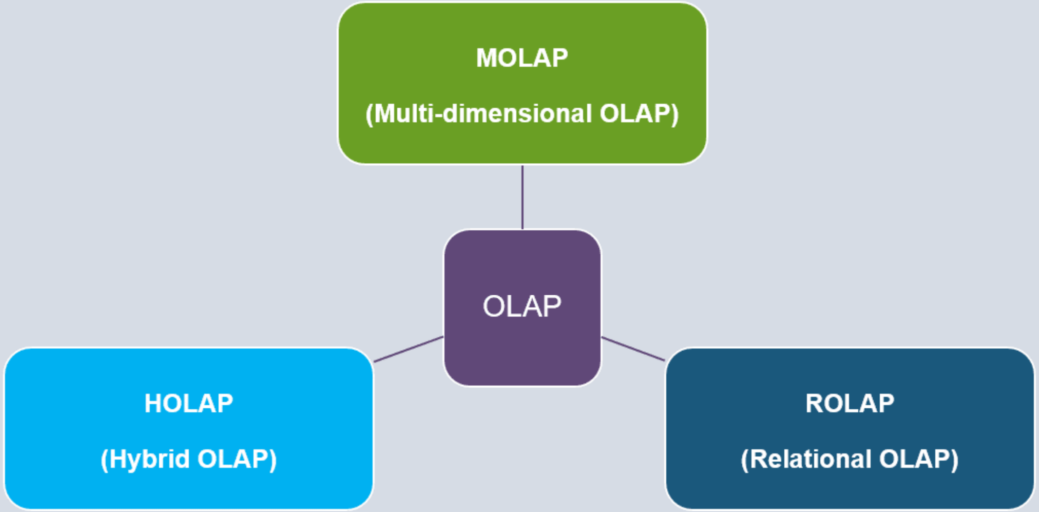
- stands for **Relational Online Analytical Processing.**
- Based on Star Schema
- It **stores data in the form of rows and columns**. ROLAP does not pre-compute data; it can be accessed through SQL queries on demand.
- The ROLAP is based on the premise that **data need not to be stored multidimensionally.**
- It **can deal with large datasets,** but the larger is the dataset more is the processing time. Thus, performance becomes an issue with rising data volumes and concurrences.
- Suitable for large datasets but can be slower than MOLAP.

## 2. MOLAP

- MOLAP is an acronym for **Multidimensional Online Analytical Processing**.
- In MOLAP**, data is pre-aggregated, summarized, and stored in the form of a multidimensional array.**
- It **enables users to model data and visualize it from multiple viewpoints**.
- Since all the complex calculations are done in advance, users can easily perform slice and dice operations on their data with fast response times.
- However, traditional MOLAP is less scalable than ROLAP, as a limited amount of data can be stored in a multidimensional cube.
- Best for small to medium-sized datasets.

# 3. HOLAP

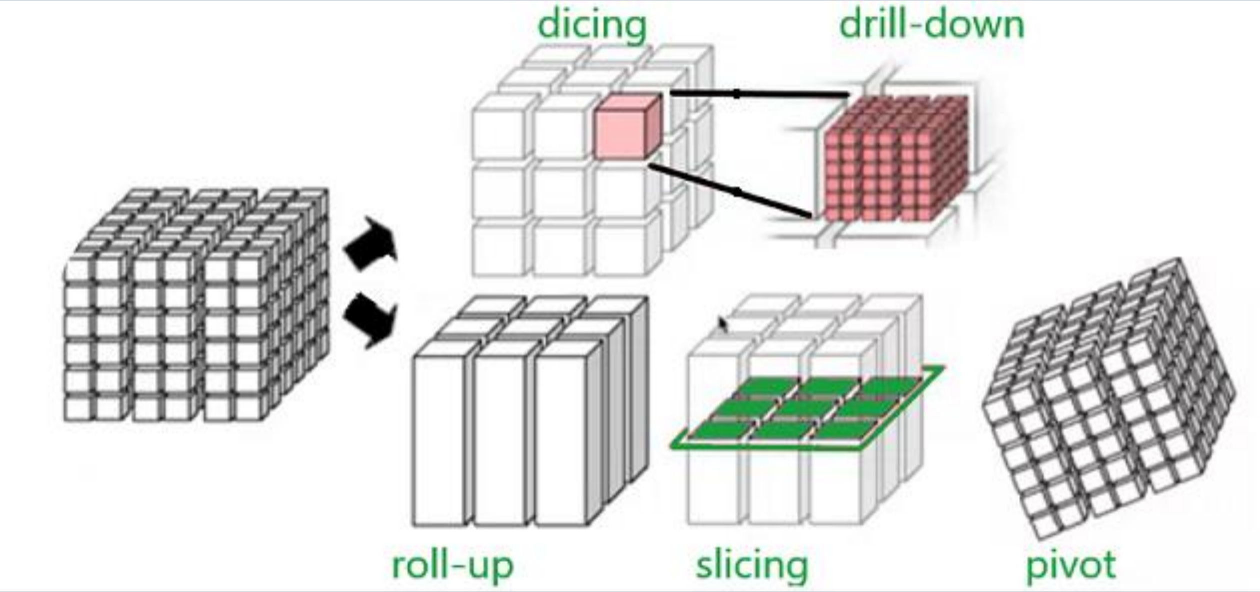
- **Hybrid OLAP**
- is a **combination of both MOLAP and ROLAP** features.
- It **uses both relational and multidimensional structures** to store data, and which one should be used to access data depends on the processing application.
- Thus, HOLAP **provides a mid-way approach** to both the methods described above.



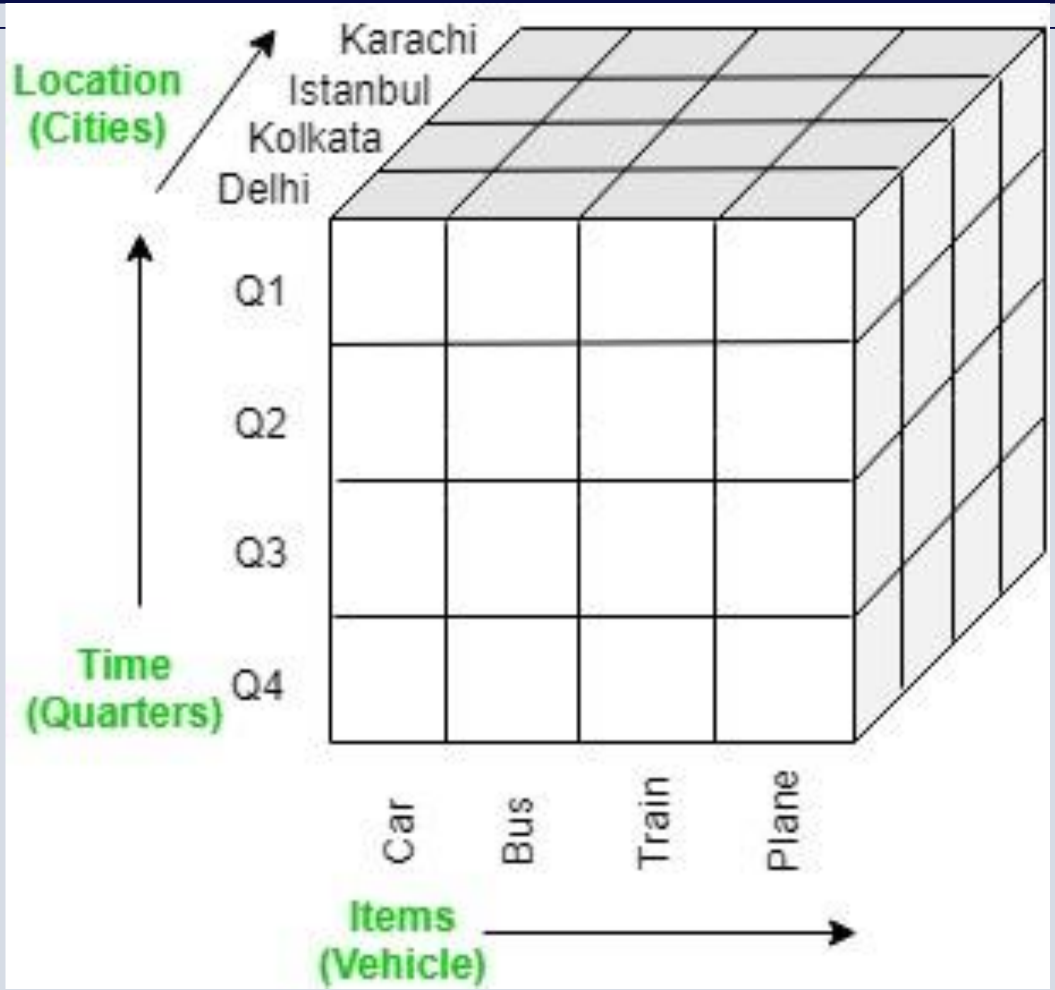
# 3.6.4 OLAP / Data-Cube Operations

There are five basic analytical operations that can be performed on an OLAP cube:

1. Drill Down
2. Roll Up
3. Dice
4. Slice
5. Pivot



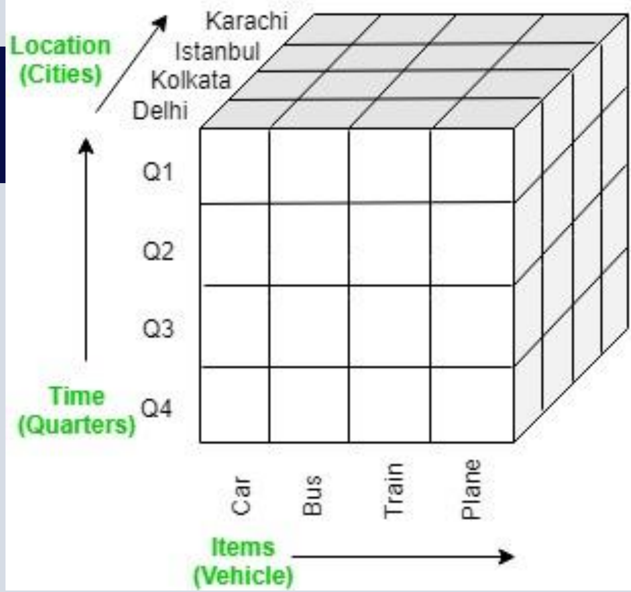
Consider below data cube:

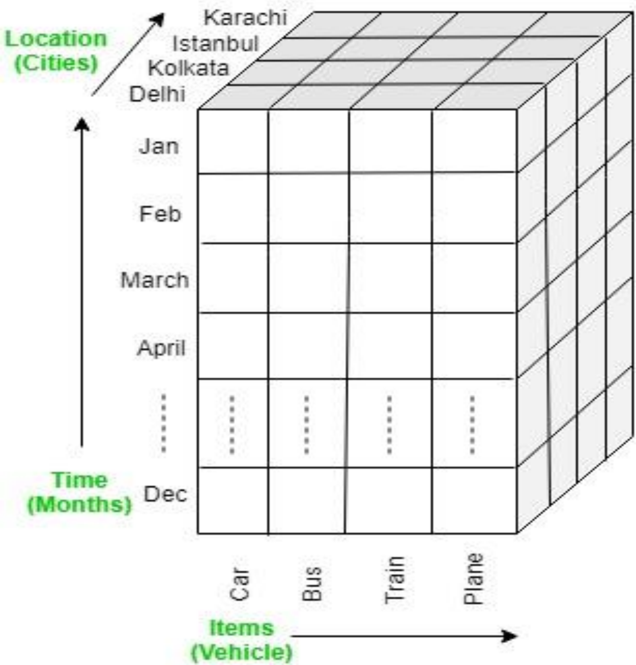


# OLAP / Data-Cube Operations

## 1. Drill down:

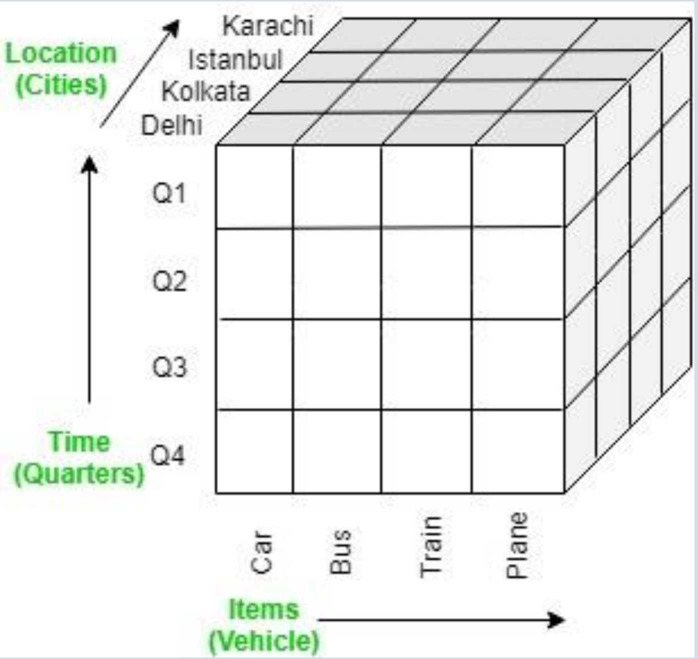
- In drill-down operation, the less detailed data is converted into highly detailed data. It can be done by:
  - Moving down in the concept hierarchy
  - Adding a new dimension
- In the cube given in overview section, the drill down operation is performed by moving down in the concept hierarchy of *Time* dimension (Quarter -> Month).

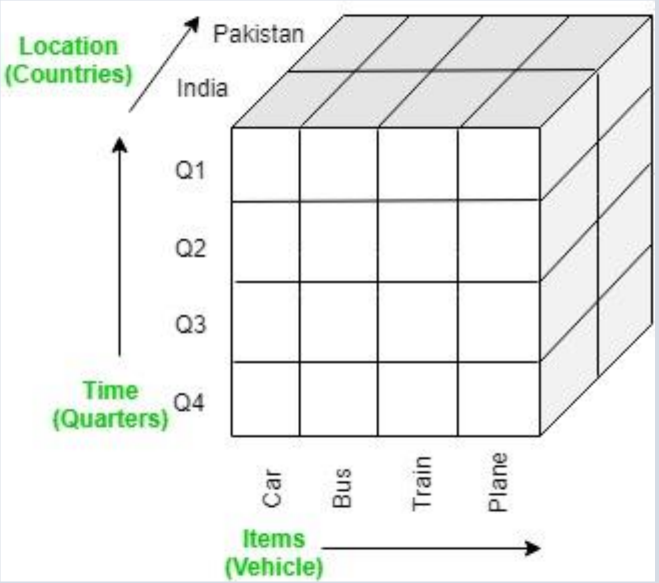




## 2. Roll up:

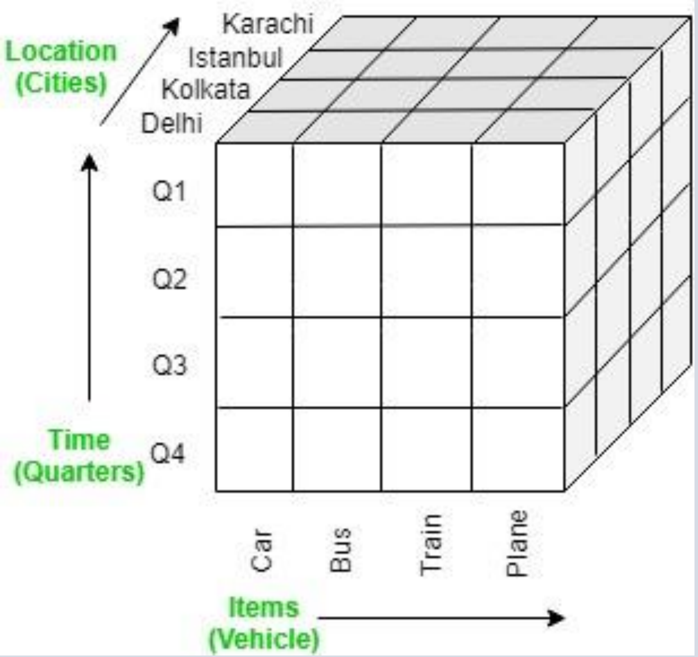
- It is just opposite of the drill-down operation.
- It performs aggregation on the OLAP cube.
- It can be done by:
  - Climbing up in the concept hierarchy
  - Reducing the dimensions
- In the cube given in the overview section, the roll-up operation is performed by climbing up in the concept hierarchy of *Location* dimension (City -> Country).

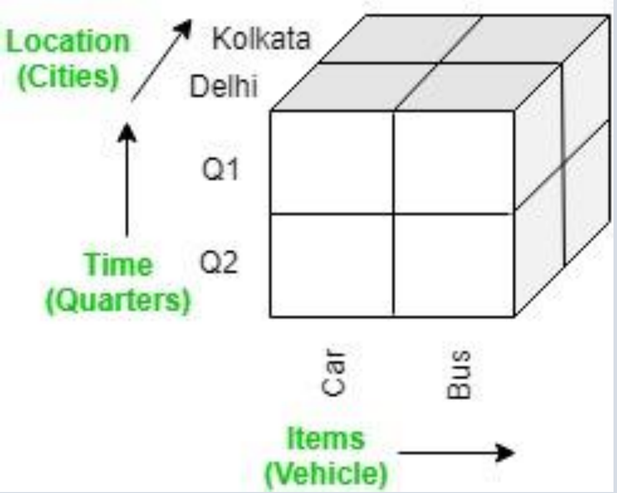




## 3. Dice:

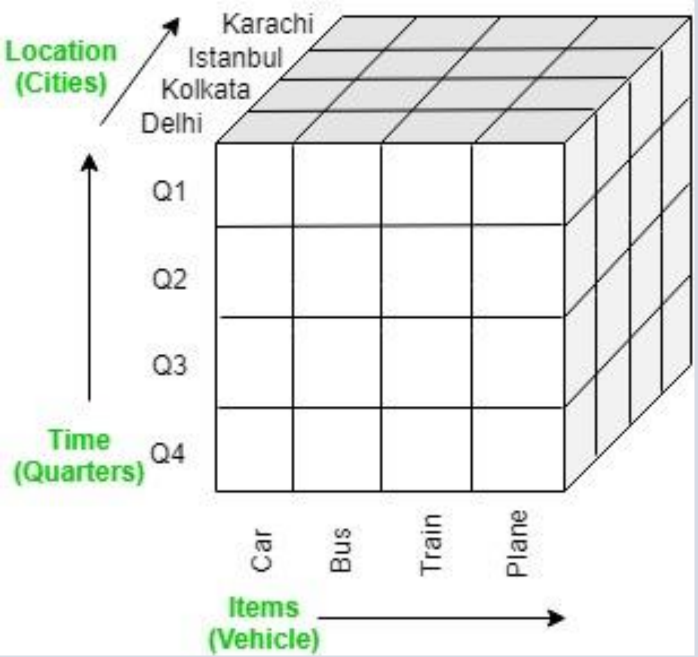
- It selects a sub-cube from the OLAP cube by selecting two or more dimensions.
- In the cube given in the overview section, a sub-cube is selected by selecting following dimensions with criteria:
  - Location = “Delhi” or “Kolkata”
  - Time = “Q1” or “Q2”
  - Item = “Car” or “Bus”

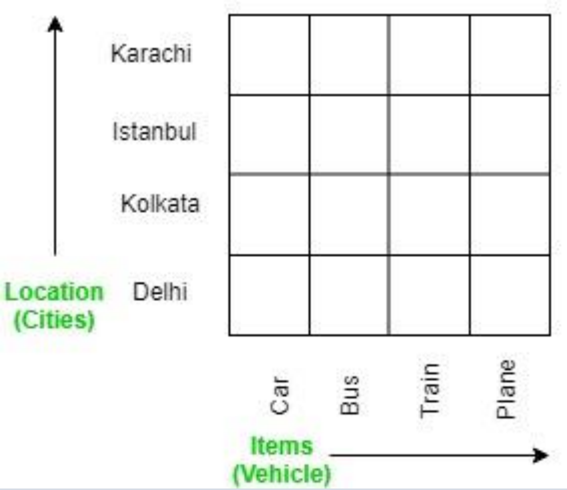


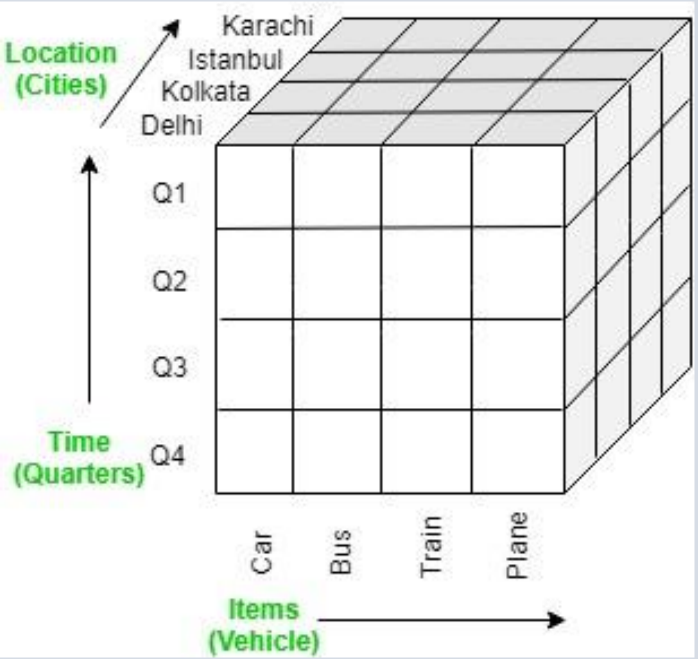


# 4. Slice:

- It selects a single dimension from the OLAP cube which results in a new sub-cube creation.
- In the cube given in the overview section, Slice is performed on the dimension Time = “Q1”.

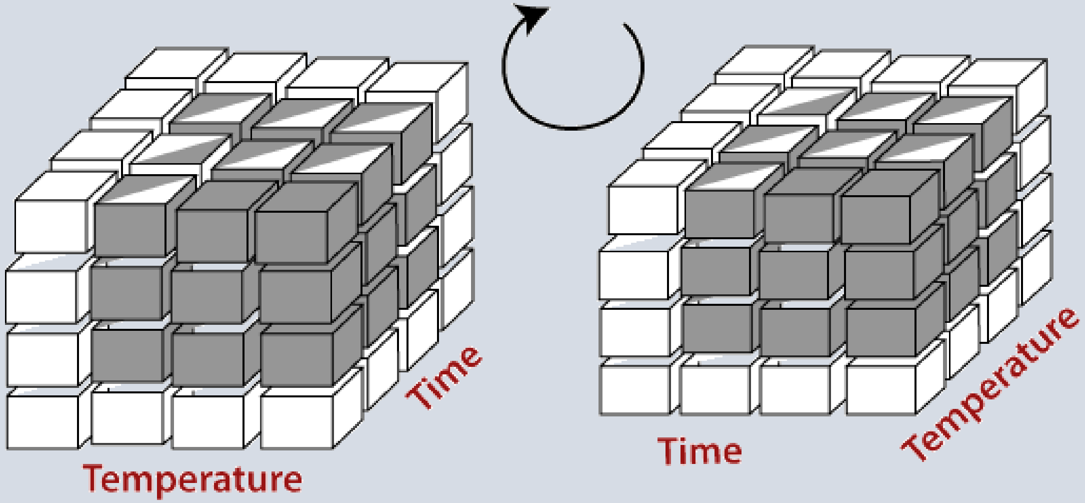


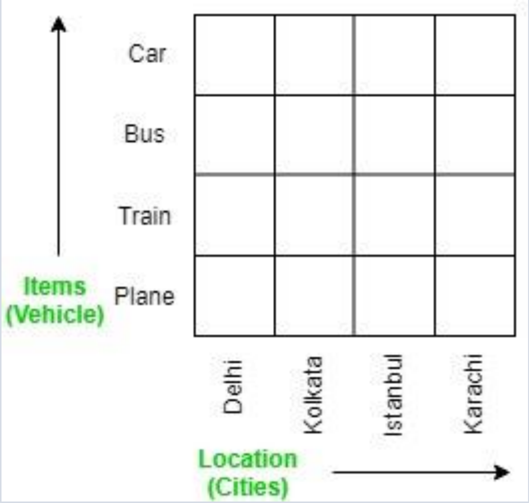




## 5. Pivot:

- It is also known as *rotation* operation as it rotates the current view to get a new view of the representation.
- In the sub-cube obtained after the slice operation, performing pivot operation gives a new view of it.





# Applications of OLAP

1. **Business Reporting**:
   - Sales forecasting, financial reporting, market analysis.
2. **Customer Relationship Management (CRM)**:
   - Analyzing customer behavior, segmenting customers, and identifying trends.
3. **Supply Chain Management**:
   - Inventory analysis, demand forecasting, and logistics planning.
4. **Healthcare**:
   - Patient data analysis, resource management, and treatment outcome evaluation.
5. **Retail**:
   - Basket analysis, price optimization, and promotional effectiveness.

| BASIS FOR COMPARISON | OLTP | OLAP |
|---|---|---|
| **Basic** | It is an online transactional system and manages database modification. | It is an online data retrieving and data analysis system. |
| **Focus** | Insert, Update, Delete information from the database. | Extract data for analyzing that helps in decision making. |
| **Data** | OLTP and its transactions are the original source of data. | Different OLTPs database becomes the source of data for OLAP. |
| **Transaction** | OLTP has short transactions. | OLAP has long transactions. |
| **Time** | The processing time of a transaction is comparatively less in OLTP. | The processing time of a transaction is comparatively more in OLAP. |
| **Queries** | Simpler queries. | Complex queries. |
| **Normalization** | Tables in OLTP database are normalized (3NF). | Tables in OLAP database are not normalized. |
| **Integrity** | OLTP database must maintain data integrity constraint. | OLAP database does not get frequently modified. Hence, data integrity is not affected. |

# Exercise

# Exercise:

1. How data mining differs with data warehousing?
2. Why is data warehouse called nonvolatile?
3. Define Data warehouse and explain its characteristics.
4. Define data mart and Meta data.
5. Explain Load manager.
6. Explain the ETL process. Distinguish between OLAP and OLTP.
7. Define warehouse manager. Write the functions of Warehouse manager.
8. What are the different types of OLAP? Explain each of them.
9. Explain the different types of OLAP operations.
10. Describe the architecture of a data warehouse.
11. Describe different types of schema for data warehouse modeling.

End of chapter