JAIN
DEEMED-TO-BE UNIVERSITY
SCHOOL OF
COMPUTER
SCIENCE AND IT

Department of CS & IT
Module 2

Data Warehousing and Data mining

23MCAG203

by
Dr. S.K. Manju bargavi
Professor
Jain(Deemed-to-be University)

key components of a data warehouse

- A typical data warehouse has four main components: a central database, ETL (extract, transform, load) tools, metadata, and access tools. All of these components are engineered for speed so that you can get results quickly and analyze data on the fly.

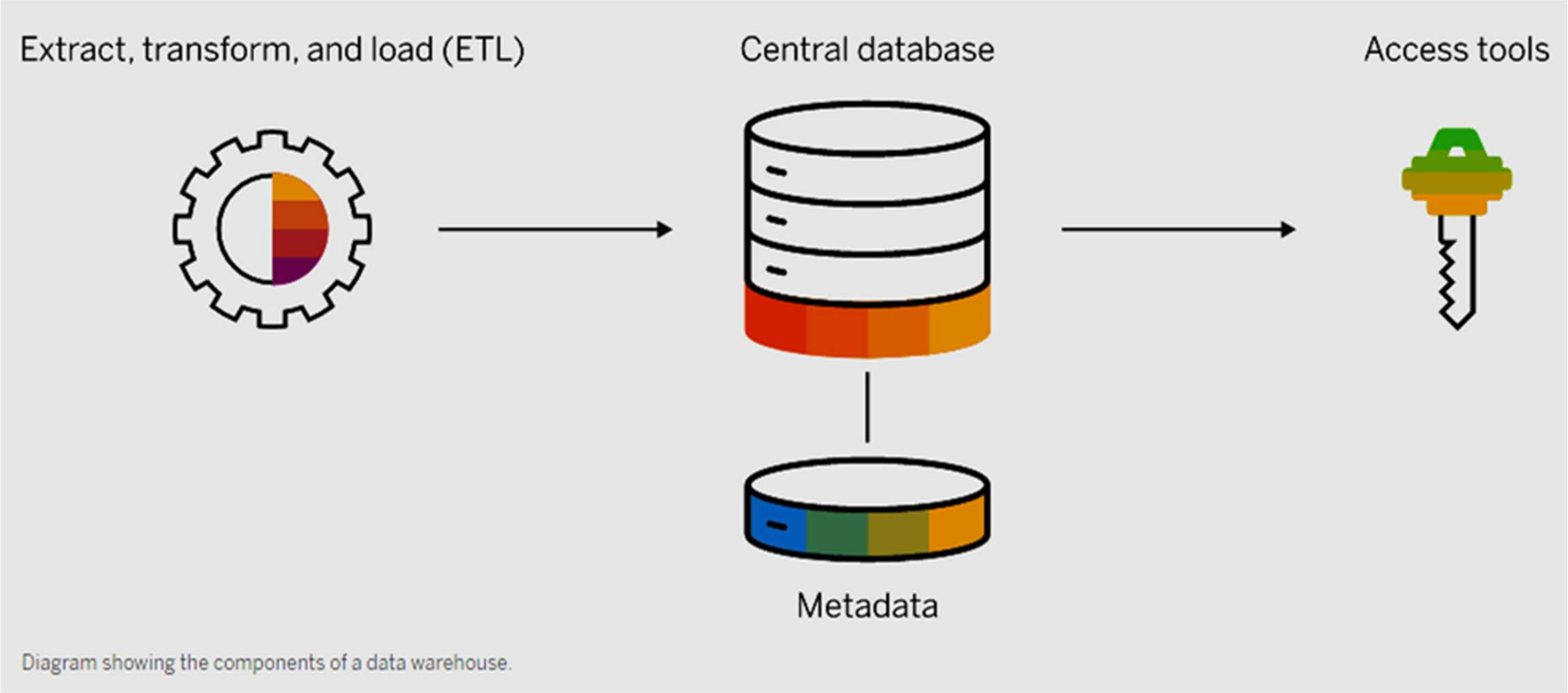


Diagram showing the components of a data warehouse.

KEY COMPONENTS OF A DATA WAREHOUSE?

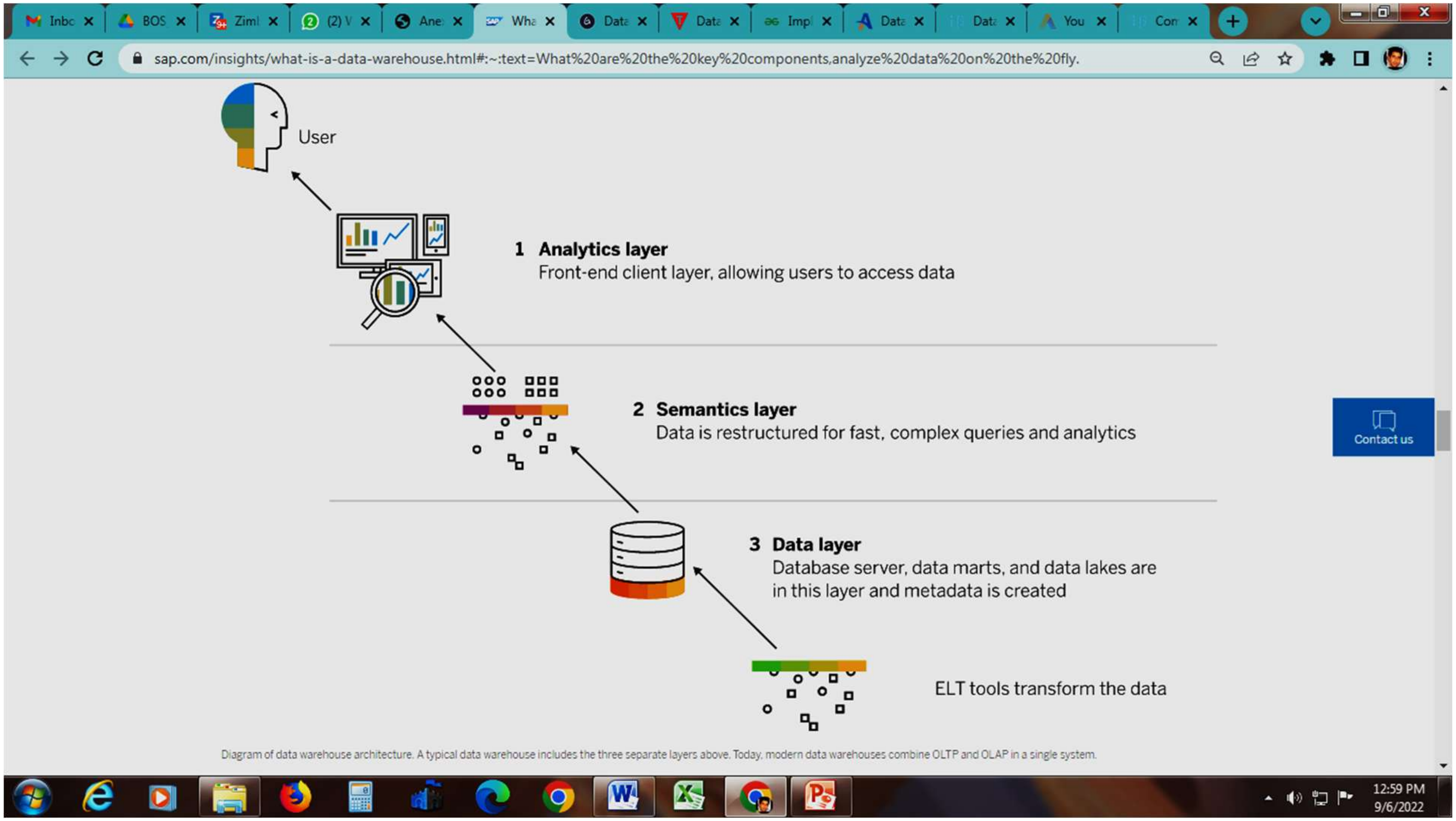
1. **Central database:** A database serves as the foundation of your data warehouse. Traditionally, these have been standard relational databases running on premise or in the cloud. But because of Big Data, the need for true, real-time performance, and a drastic reduction in the cost of RAM, in-memory databases are rapidly gaining in popularity.
2. **Data integration:** Data is pulled from source systems and modified to align the information for rapid analytical consumption using a variety of data integration approaches such as ETL (extract, transform, load) and ELT as well as real-time data replication, bulk-load processing, data transformation, and data quality and enrichment services.

key components of a data warehouse

3. **Metadata:** Metadata is data about your data. It specifies the source, usage, values, and other features of the data sets in your data warehouse. There is business metadata, which adds context to your data, and technical metadata, which describes how to access data – including where it resides and how it is structured.

3. **Data warehouse access tools:** Access tools allow users to interact with the data in your data warehouse. Examples of access tools include: query and reporting tools, application development tools, data mining tools, and OLAP tools.

Data warehouse architecture



Inbo
BOS
Ziml
(2) V
Ane
Wha
Data
Data
Impl
Data
Data
You
Con
sap.com/insights/what-is-a-data-warehouse.html#:~:text=What%20are%20the%20key%20components,analyze%20data%20on%20the%20fly.
User
Contact us
12:59 PM
9/6/2022

1 **Analytics layer**
Front-end client layer, allowing users to access data

2 **Semantics layer**
Data is restructured for fast, complex queries and analytics

3 **Data layer**
Database server, data marts, and data lakes are in this layer and metadata is created

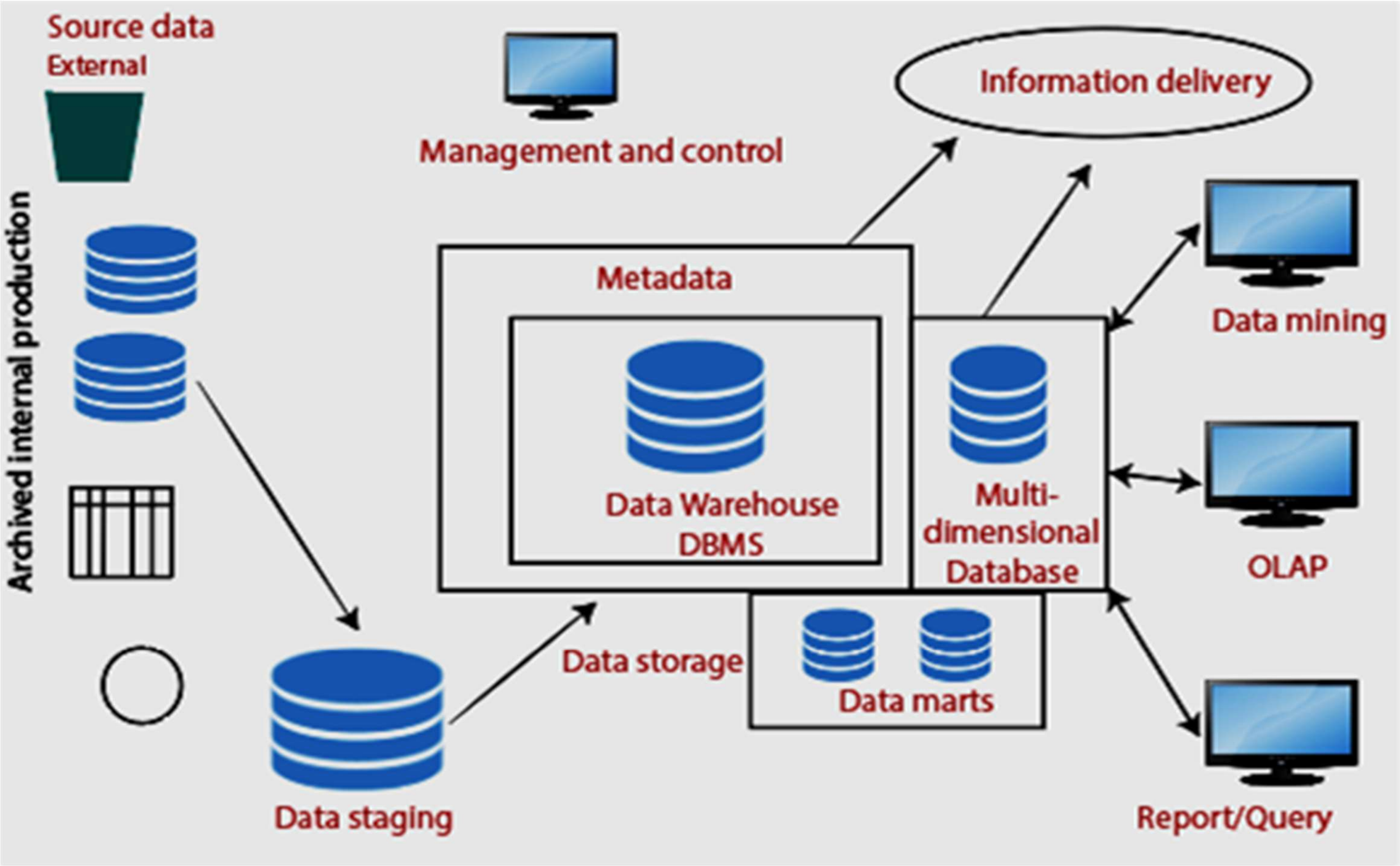
ELT tools transform the data

Diagram of data warehouse architecture. A typical data warehouse includes the three separate layers above. Today, modern data warehouses combine OLTP and OLAP in a single system.

Data warehouse architecture

- **1. Data layer:** Data is extracted from your sources and then transformed and loaded into the bottom tier using ETL tools. The bottom tier consists of your database server, data marts, and data lakes. Metadata is created in this tier – and data integration tools, like data virtualization, are used to seamlessly combine and aggregate data.
- **2. Semantics layer:** In the middle tier, online analytical processing (OLAP) and online transactional processing (OLTP) servers restructure the data for fast, complex queries and analytics.
- **3. Analytics layer:** The top tier is the front-end client layer. It holds the data warehouse access tools that let users interact with data, create dashboards and reports, monitor KPIs, mine and analyze data, build apps, and more. This tier often includes a workbench or sandbox area for data exploration and new data model development.

Components /Building Blocks of Data Warehouse



Source Data Component

- Source data coming into the data warehouses may be grouped into four broad categories:

- **Production Data:** This type of data comes from the different operating systems of the enterprise. Based on the data requirements in the data warehouse, we choose segments of the data from the various operational modes.
- **Internal Data:** In each organization, the client keeps their "**private**" spreadsheets, reports, customer profiles, and sometimes even department databases. This is the internal data, part of which could be useful in a data warehouse.

Source Data Component

- **Archived Data:** Operational systems are mainly intended to run the current business. In every operational system, we periodically take the old data and store it in achieved files.
- **External Data:** Most executives depend on information from external sources for a large percentage of the information they use. They use statistics associating to their industry produced by the external department.

Architecture

- May have one or more tiers
  - Determined by warehouse, data acquisition (back end), and client (front end)
    - One tier, where all run on same platform, is rare
    - Two tier usually combines DSS engine (client) with warehouse
      - More economical
    - Three tier separates these functional parts

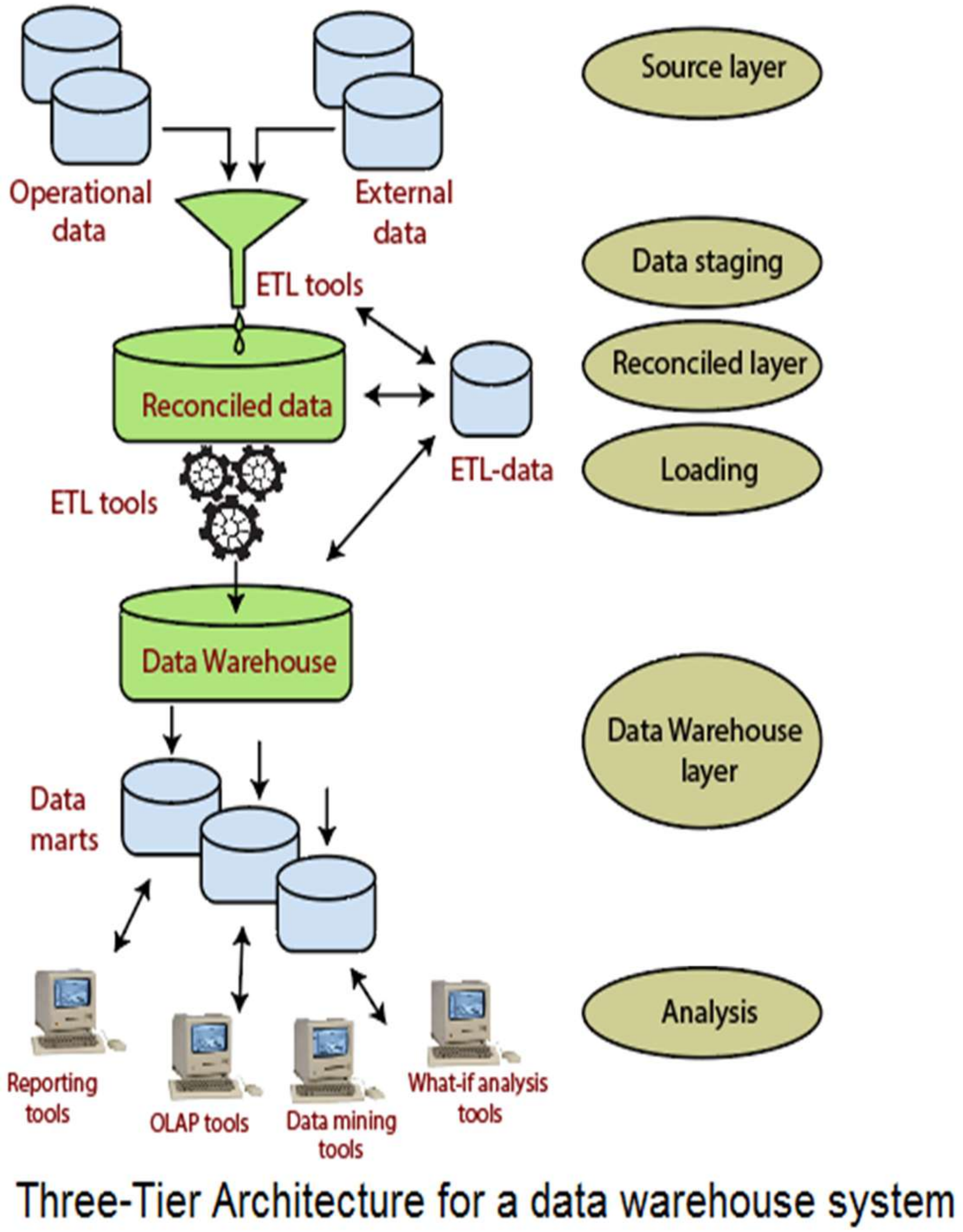


Figure 1.2: Architecture of a 3 Tier Data Warehouse

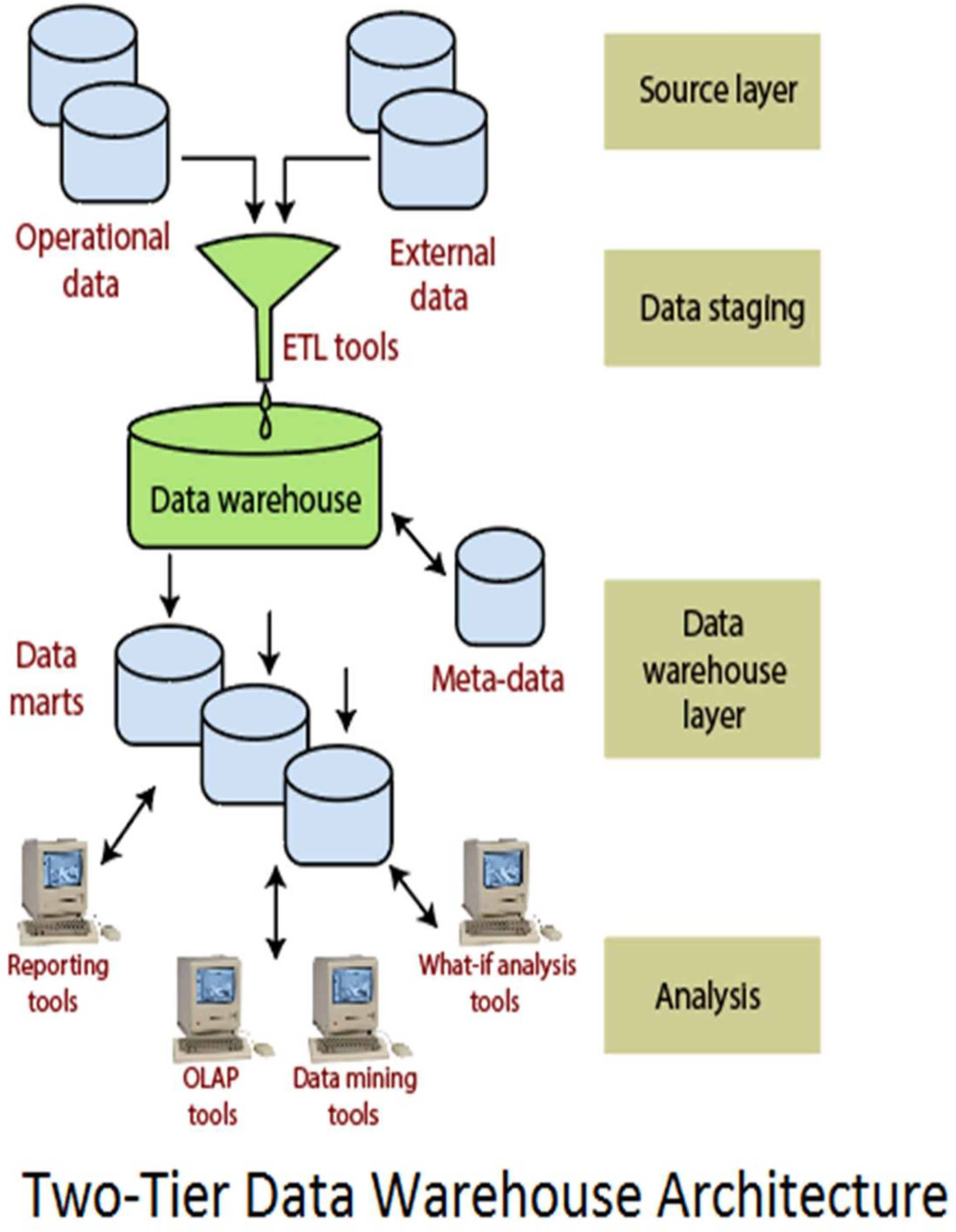
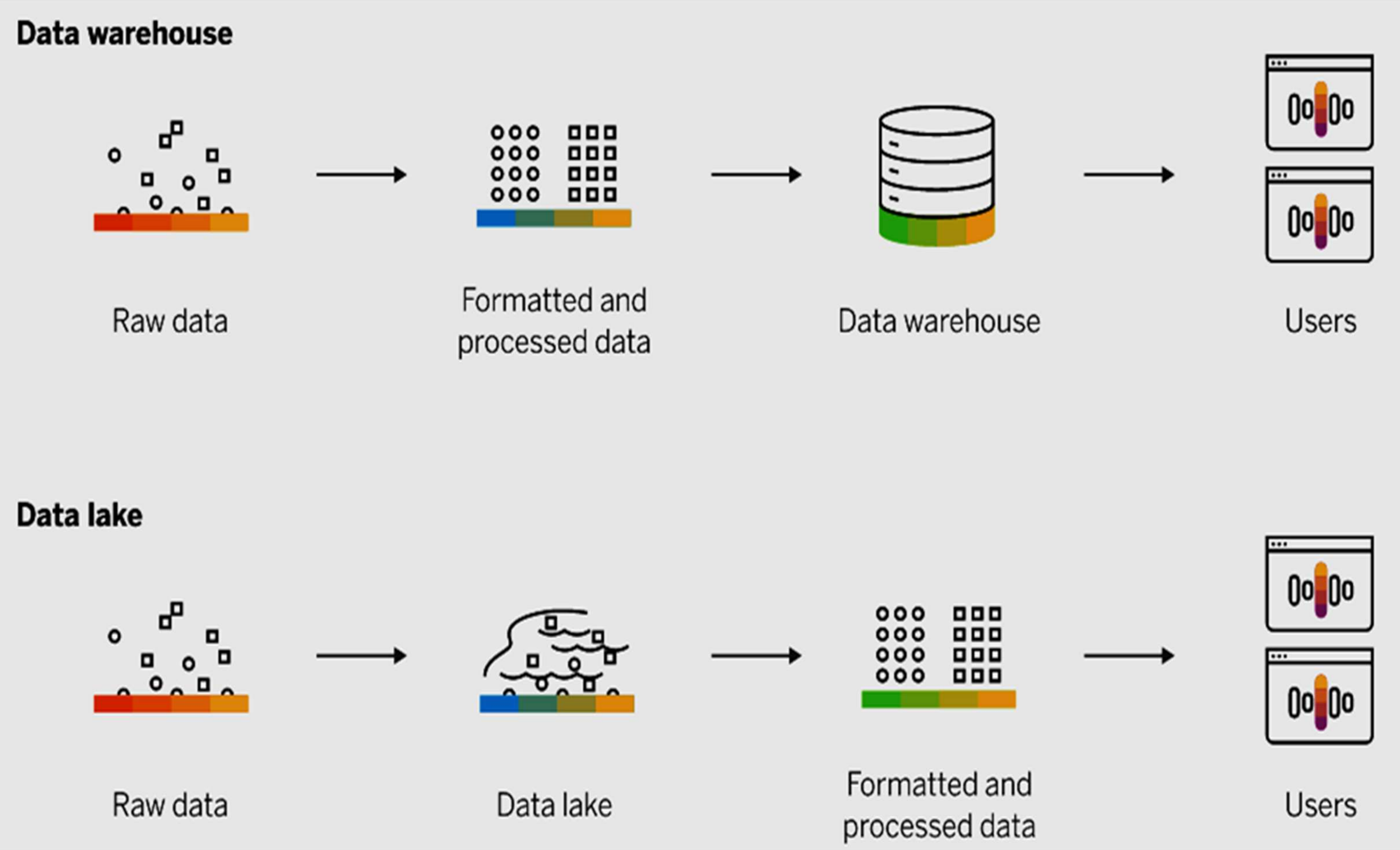


Figure 1.3: Architecture of a 2 Tier Data Warehouse

Data warehouse vs. data lake

- Both data warehouses and data lakes are used for storing Big Data, but they are very different storage systems. A data warehouse stores data that has been formatted for a specific purpose, whereas a data lake stores data in its raw, unprocessed state – the purpose of which has not yet been defined.
- Data warehouses and lakes often complement each other. For example, when raw data stored in a lake is needed to answer a business question, it can be extracted, cleaned, transformed, and used in a data warehouse for analysis. The volume of data, database performance, and storage pricing play important role in helping you choose the right storage solution.

Data warehouse vs. data lake



Federated Data Warehouse

- A federated data warehouse is the integration of heterogeneous business intelligence systems set to provide analytical capabilities across the different function of an organization. It’s a realistic method to achieve the “single version of the truth” across the organization considering the political and implementation challenges. It aims to integrate the key business metrics, measures and dimensions. But it doesn’t aspire to create a single platform to carry out all the functional analysis.

Federated Data Warehouse

- The foundation of federated DW is the common business model of the organization. Common business model is a continuously evolving semantic understanding of the business in consent with all business units.
- This common business model is the initiation point for an iterative process of building the different business intelligent subsystem based on a common staging area.

The Need of Federated Data Warehouse

- **Mergers & acquisitions –** The way business has adopted the inorganic growth path has put many architects in unenviable positions. Merger & acquisitions have become routine events in today’s business scenario. Each acquisition brings in a big information bank along with it but the biggest challenge is to integrate this information with the existing BI system. It doesn’t make sense to abandon a fully functioning data warehouse infrastructure, so the DW teams are forced to adopt a federated BI architecture. It’s certainly not an easy task to completely imbibe the new system with the existing BI architecture. The best way to handle it is to federate and integrate the two systems.

The Need of Federated Data Warehouse

- **Cross-functional requirements –** Cross function analysis has become an everyday need for most of the enterprises. A federated DW stores the information of most of the functions, if not all, that an organization deals with. This information could be stored in one or more than one BI systems. FDW support the cross functional analysis using the common dimension across the different systems. Common dimension is an outcome of the common business model which defines the collective nature of business functionality. A federated DW is an active cooperation of multiple business intelligence systems where each system can talk to other BI system and fulfill cross-functional requirements.

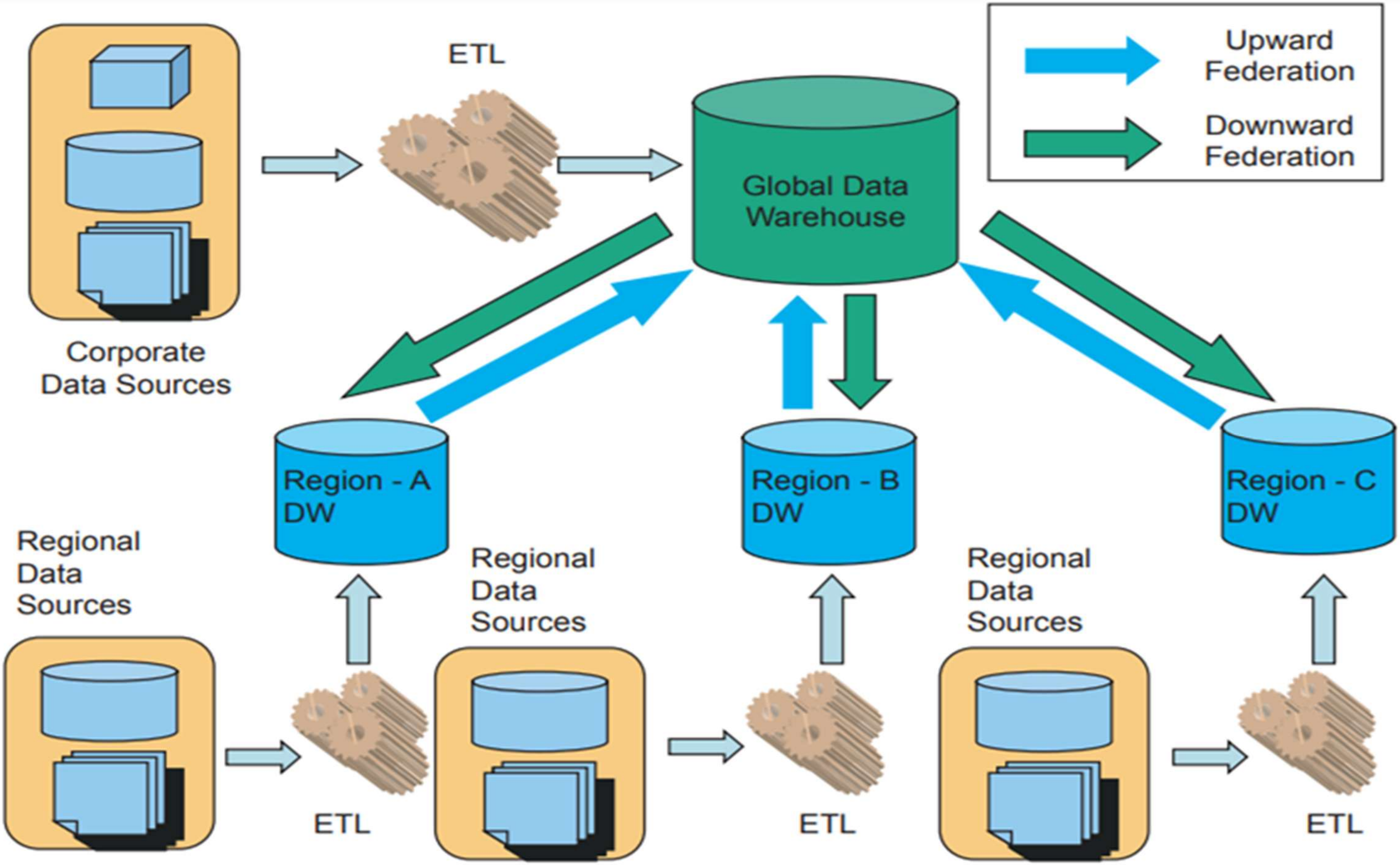
The Need of Federated Data Warehouse

- **A speedy cost effective solution –** The build time involved in federated data warehouse is enormously less when compared with enterprise wide data warehouse. One of the main reasons is that the federated DW tries to integrate the existing system by providing a common framework. It does not aspire to build a uniform foundation which is a tedious and lengthy process. The incremental nature of the federated DW reduces the long waiting period associated with most of the big DW, eventually reducing the cost as well.

The Need of Federated Data Warehouse

- **Ease of implementation –** When building an enterprise wide data warehouses, architects have experienced a lot of politics and vested interests trying to mold the crucial decisions. In the real world scenario these factors cannot be sidelined and at times, influence the implementation in a larger manner. Federated DW approaches the problem in a more pragmatic manner and the idea to use the old BI systems by integrating them with the newer system prevents the major point of conflict.

Regional federation



Regional federation

- A big organization has regional data warehouses for regional analytical requirements and a global data warehouse for the corporate requirements. The difference between the two systems (regional & global) is based on the nature of the data that is contained in each system. The global data warehouse stores data which is mostly summarized for the purpose of corporate analytics and reference data.
- Reference data would contain confirmed dimensions and corporate level data like currency conversions etc. Regional warehouses store data based on the regional analytical requirements which generally has more detailed information.

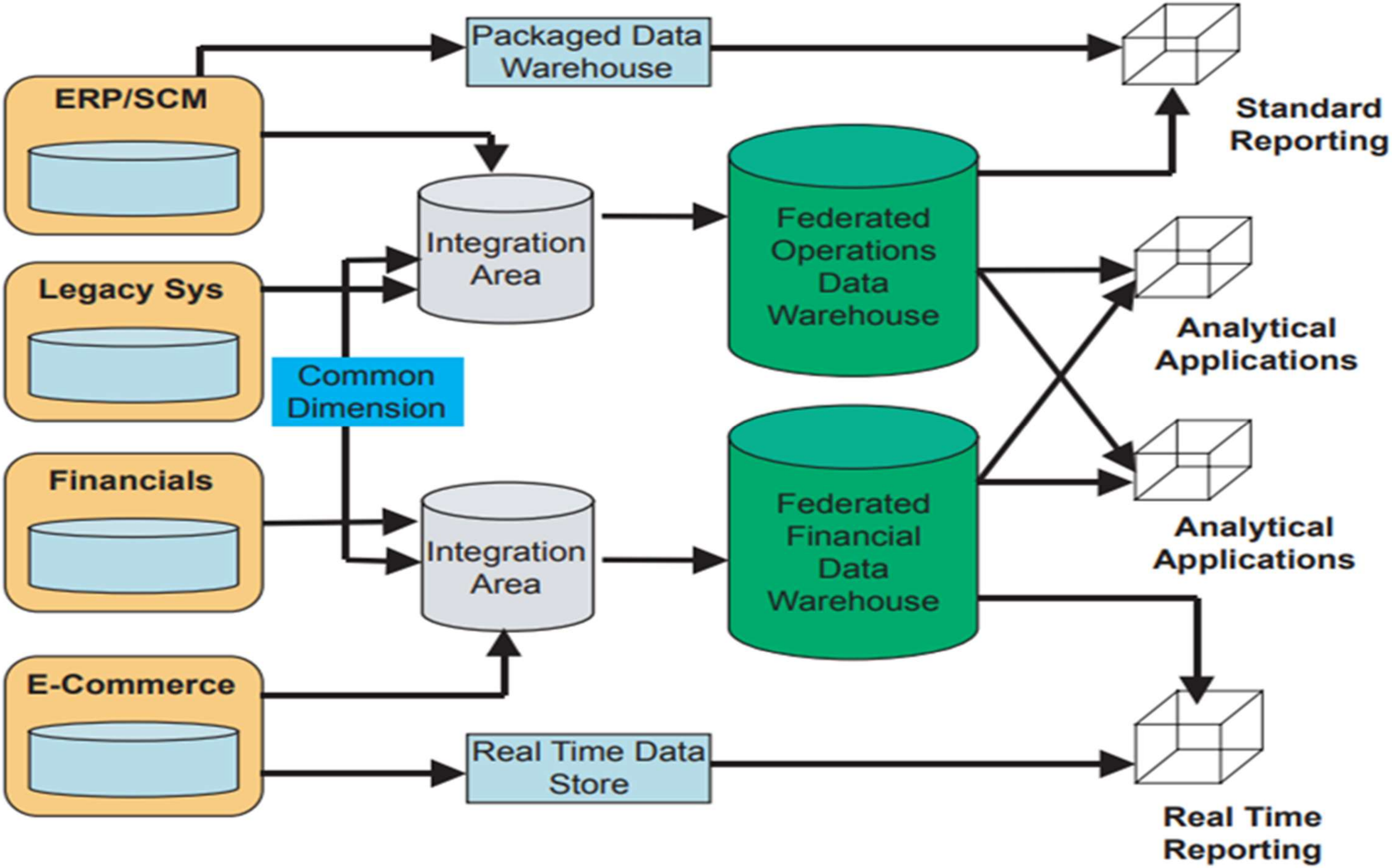
Regional federation

- Reference data provide the integration platform for regional and global warehouse. The data flow from regional data warehouses to the global data warehouse is defined as upward federation and the data flow from global data warehouse to regional data warehouses as downward federation. For data consistency and integrity, reference data should be made common across the various data warehouses. Uniform & consistent definition of reference data across the participating data warehouses ensures the ‘single version of truth’ across the federated architecture.

Data movement

- **During Upward Federation:**
- Upward federation would include the movement of fact data from regional data warehouses to the global data warehouse. If required, this data can be aggregated during movement.
- **During Downward Federation:**
- • Reference data will flow from global to the regional level data warehouses. This flow would be strictly downward to ensure consistency and integrity of reference data.
- • Transactional data that is available in corporate transactional systems (such as corporate ERP) will be sourced at the global level, cleansed and transformed and then moved to the respective regions.
- • Summary data – The global summarization will happen in the global data warehouse and will be moved to the regional data warehouses. This can be useful in the analyzing how a particular Region is performing against the rest of the company.

functional federation



Functional federation

- A functional federated data warehouse will be the candidate when an organization has built data warehouses which are either subject specific, packaged solutions or built for a specific enterprise application.

- The federated data warehouse architecture is the “big umbrella” that provides the foundation and environment to facilitate and enable business analysis and decision support in this heterogeneous environment.

Functional federation

- A functional federated data warehouse has room for all the components of a contemporary BI application of a large and complex business entity. Typically it should contain the following components:
- Packaged data warehouses (DWs) and data marts (DMs)
- Custom built data warehouses and data marts
- Real time data store & real time data reporting
- Custom built analytical applications
- Online analytical processing (OLAP) tools
- Extraction, transformation and load (ETL) tools
- Cross functional reporting systems.

Functional federation

- The following steps should be taken when architecting a federated data warehouse:
  - Define the federated data warehouse goals and business requirements
  - Define team structure & assign roles and responsibilities
  - Document existing data warehouses & BI systems.

The information to be documented is - Business areas addressed by the existing systems, target users, reporting and analytical capabilities provided, the data sourcing strategy and cleansing/transformations methodology while moving the data into these systems

- An analysis needs to be done to identify whether it is a case for federated architecture
- Define the common business model which should provide the basis for common dimensions
- Define an integration strategy for federation. Identify inter-dependency between existing and new to-be developed data warehouses
- Perform the integration of existing data warehouses into the new architecture in small phases, giving first priority to the areas which are critical for business.

Key Success Factors

- In order to develop a successful federated data warehouse, certain criteria have to be fulfilled:
- • The various component data warehouses should share conformed dimensions. The conformed dimensions represent the dimensions having identical business meaning, identical structure and identical data. However, to keep them physically separated or not, can be decided based on implementation complexities
- • There should well defined, documented and integrated business rules which would be used across the component data warehouses in the whole architecture

Key Success Factors

- The component data warehouses should have confirmed facts or measures. Conformed facts have identical business meaning and exactly same data values for same set of dimensions. Identical data value for a confirmed fact will also ensure “single version of truth” which is essentially one of the prime design objectives of a federated data warehouse
- A single ETL tool should be used in the whole federated data warehouse. Usage of a single ETL tool would also ensure common metadata

Conclusion

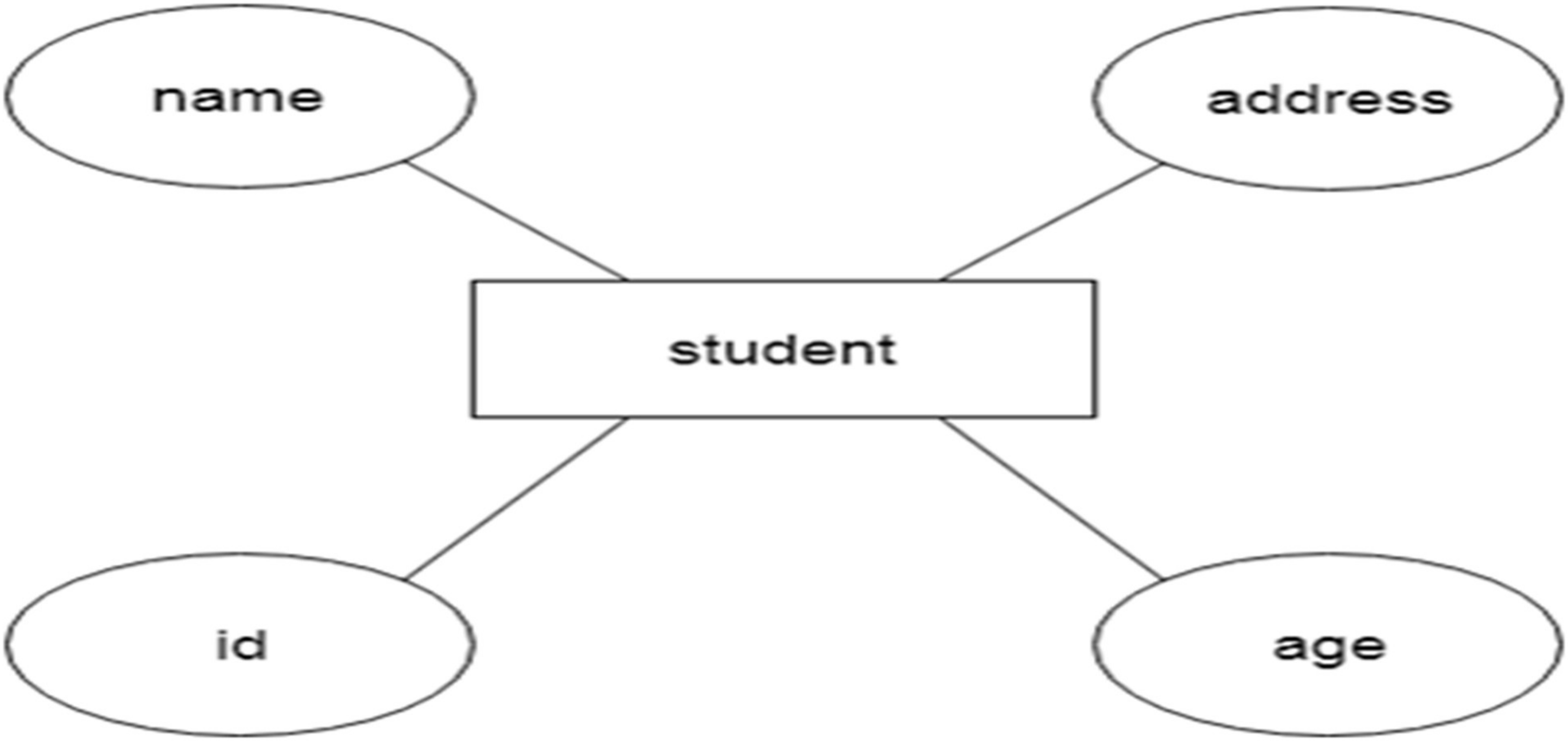
- A Federated Data Warehouse offers a very practical solution for implementing and delivering value added solution to a business.
- The iterative nature of a Federated Date Warehouse reduces the extended delays normally associated with data warehousing solutions.
- In conclusion, a Federated Data Warehouse, with its lower costs and higher reliability, provides an excellent value proposition to organizations.

ER model

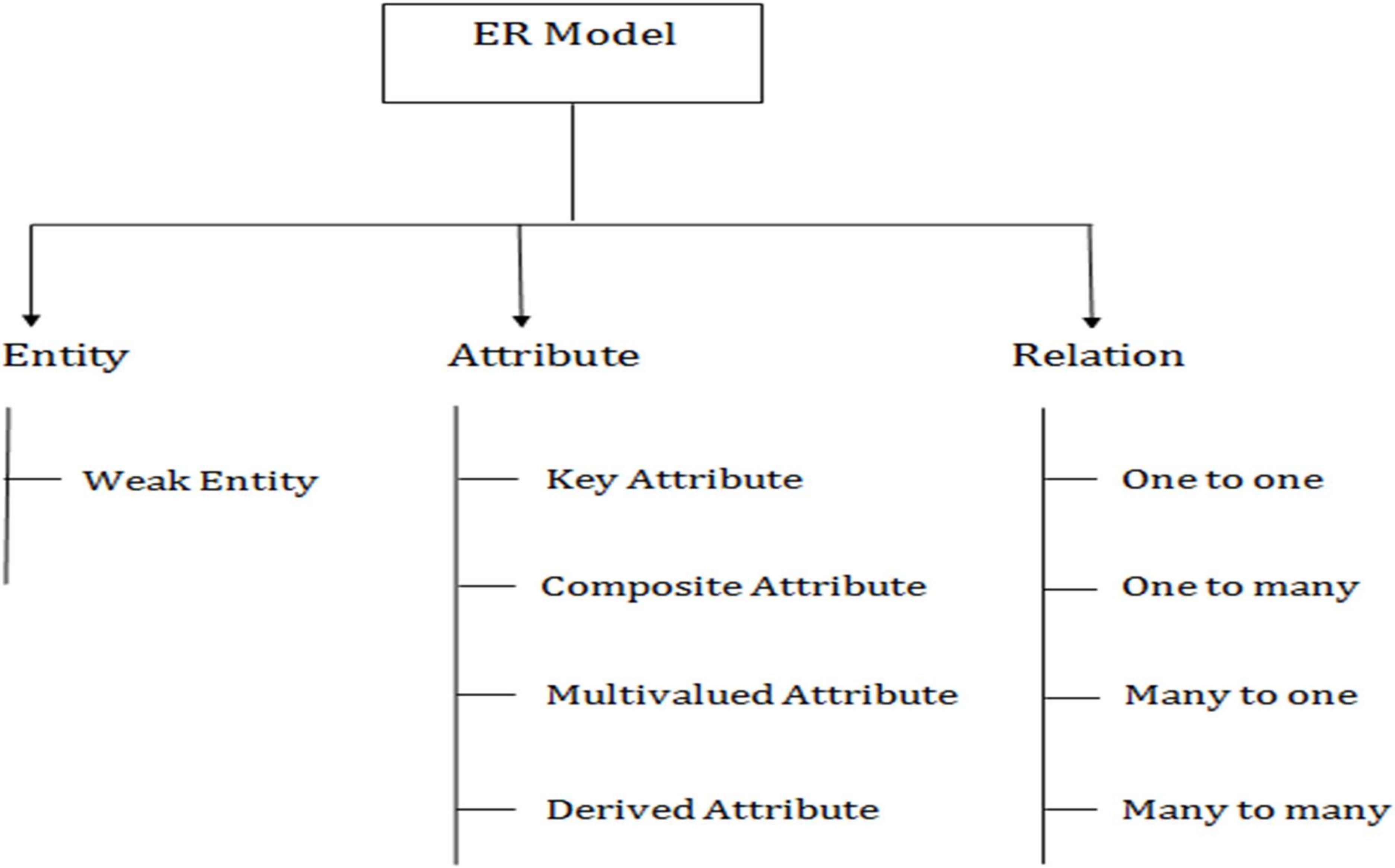
- ER model stands for an Entity-Relationship model. It is a high-level data model. This model is used to define the data elements and relationship for a specified system.
- It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.
- In ER modeling, the database structure is portrayed as a diagram called an entity-relationship diagram.

ER model

- Suppose we design a school database. In this database, the student will be an entity with attributes like address, name, id, age, etc. The address can be another entity with attributes like city, street name, pin code, etc and there will be a relationship between them.



Component of ER Diagram

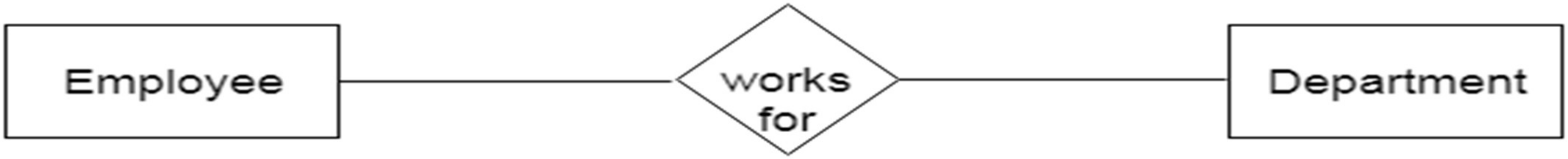


ER model

- ER model stands for an Entity-Relationship model. It is a high-level data model. This model is used to define the data elements and relationship for a specified system.
- It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.
- In ER modeling, the database structure is portrayed as a diagram called an entity-relationship diagram.

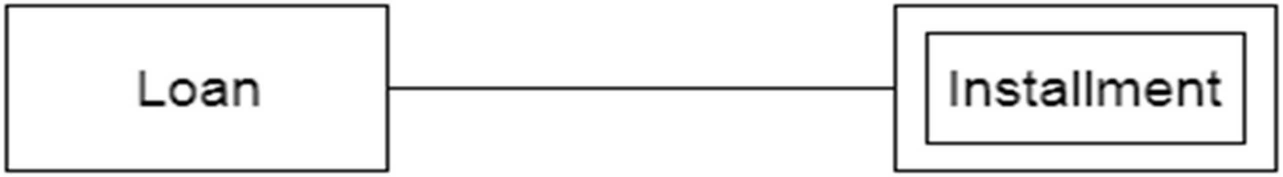
Entity

- Entity:
- An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.
- Consider an organization as an example- manager, product, employee, department etc. can be taken as an entity.



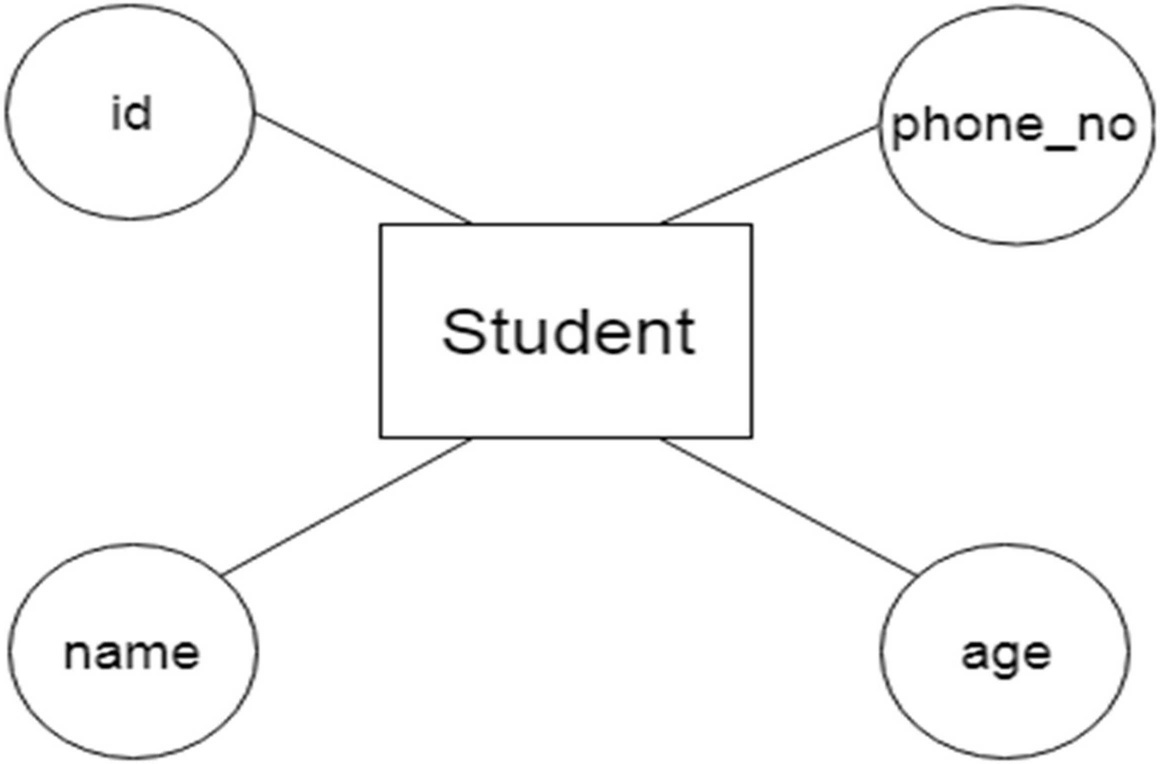
a. Weak Entity

- An entity that depends on another entity called a weak entity. The weak entity doesn't contain any key attribute of its own. The weak entity is represented by a double rectangle.



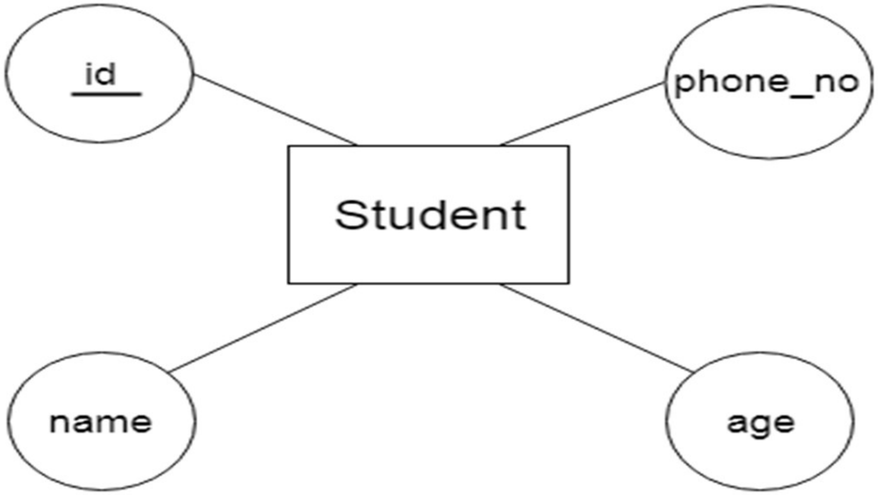
Attribute

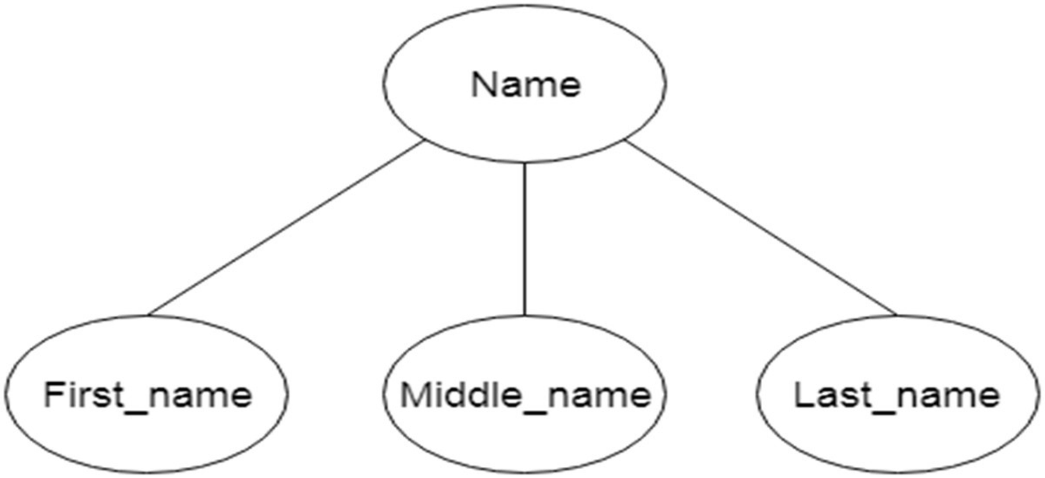
- Attribute
- The attribute is used to describe the property of an entity. Eclipse is used to represent an attribute.
- For example, id, age, contact number, name, etc. can be attributes of a student.



Attribute

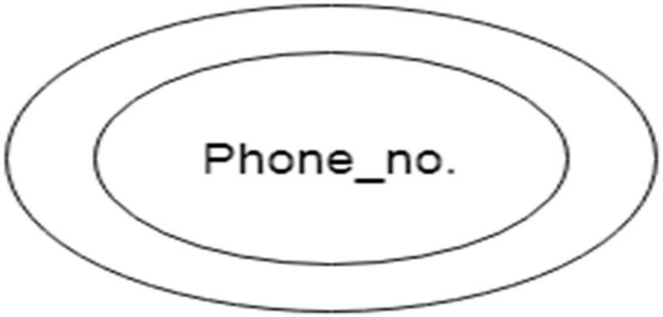
- **a. Key Attribute**
- The key attribute is used to represent the main characteristics of an entity. It represents a primary key. The key attribute is represented by an ellipse with the text underlined.
- **b. Composite Attribute**
- An attribute that composed of many other attributes is known as a composite attribute. The composite attribute is represented by an ellipse, and those ellipses are connected with an ellipse.

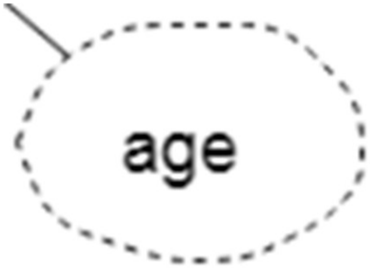




Attribute

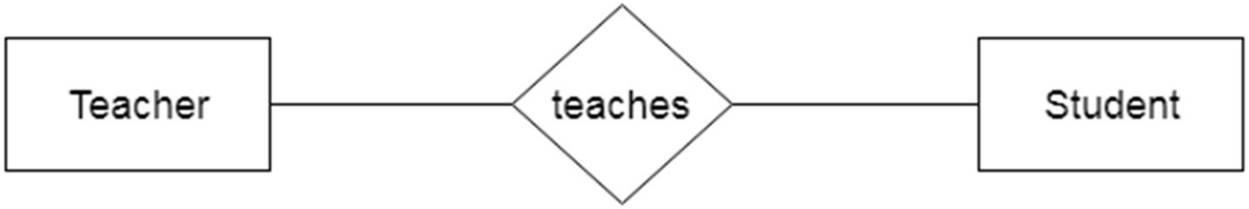
- **c. Multivalued Attribute**
- An attribute can have more than one value. These attributes are known as a multivalued attribute. The double oval is used to represent multivalued attribute. **For example,** a student can have more than one phone number.
- **d. Derived Attribute**
- An attribute that can be derived from other attribute is known as a derived attribute. It can be represented by a dashed ellipse.
- **For example,** A person's age changes over time and can be derived from another attribute like Date of birth.



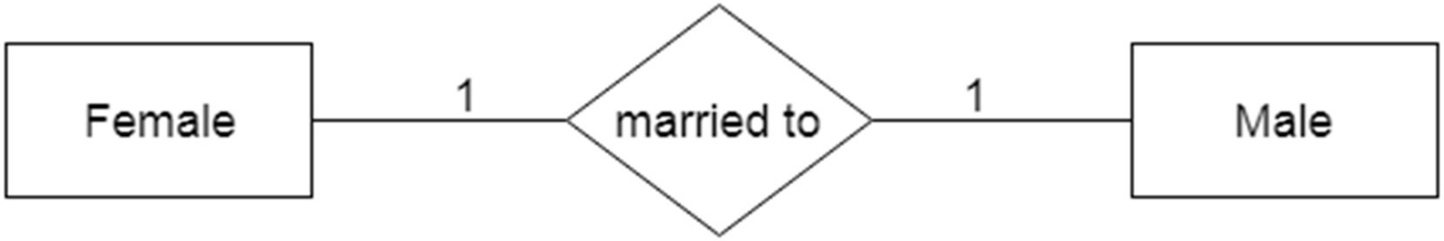


Relationship

- A relationship is used to describe the relation between entities. Diamond or rhombus is used to represent the relationship.

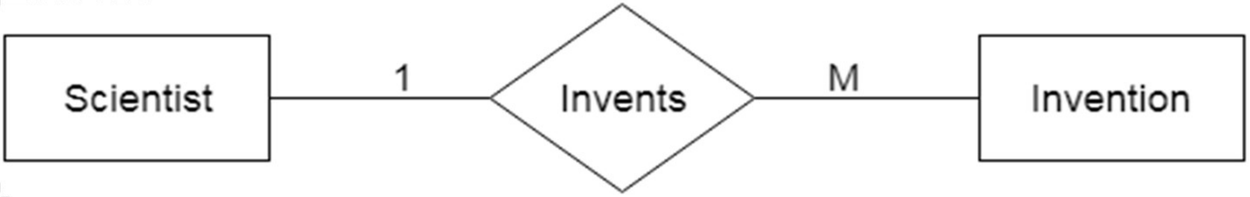


- When only one instance of an entity is associated with the relationship, then it is known as one to one relationship.
- **For example,** A female can marry to one male, and a male can marry to one female.

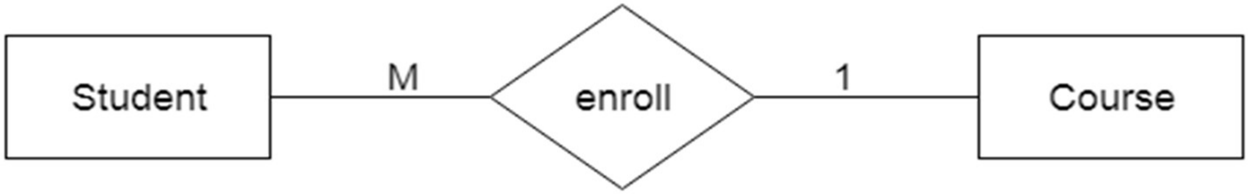


Relationship

- **b. One-to-many relationship**
- When only one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then this is known as a one-to-many relationship. **For example,** Scientist can invent many inventions, but the invention is done by the only specific scientist

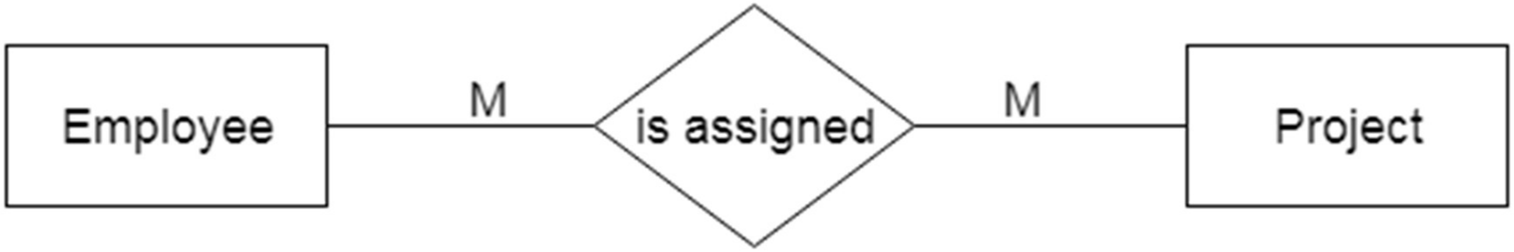


- **c. Many-to-one relationship**
- When more than one instance of the entity on the left, and only one instance of an entity on the right associates with the relationship then it is known as a many-to-one relationship. **For example,** Student enrolls for only one course, but a course can have many students.



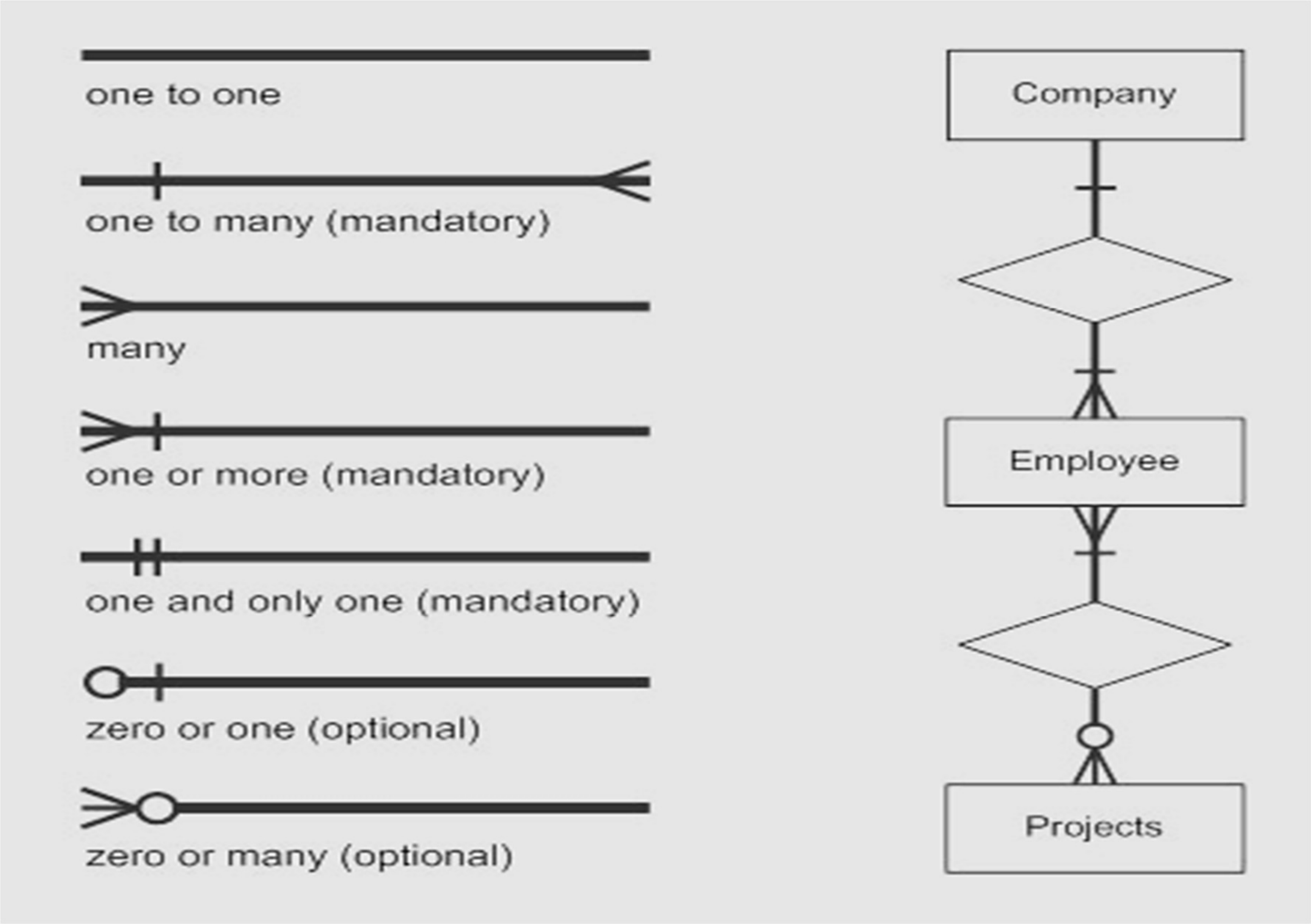
Relationship

- **d. Many-to-many relationship**
- When more than one instance of the entity on the left, and more than one instance of an entity on the right associates with the relationship then it is known as a many-to-many relationship.
- **For example,** Employee can assign by many projects and project can have many employees.



Notation of ER diagram

- Database can be represented using the notations. In ER diagram, many notations are used to express the cardinality.



Dimensional Modeling

- **Dimensional Modeling (DM)** is a data structure technique optimized for data storage in a Data warehouse. The purpose of dimensional modeling is to optimize the database for faster retrieval of data. The concept of Dimensional Modelling was developed by Ralph Kimball and consists of “fact” and “dimension” tables.
- A dimensional model in data warehouse is designed to read, summarize, analyze numeric information like values, balances, counts, weights, etc. in a data warehouse. In contrast, relation models are optimized for addition, updating and deletion of data in a real-time Online Transaction System.

Dimensional Modeling

- These dimensional and relational models have their unique way of data storage that has specific advantages.
- For instance, in the relational model, normalization and ER models reduce redundancy in data. On the contrary, dimensional model in data warehouse arranges data in such a way that it is easier to retrieve information and generate reports.
- Hence, Dimensional models are used in data warehouse systems and not a good fit for relational systems.

Objectives of Dimensional Modeling

- The purposes of dimensional modeling are:
- To produce database architecture that is easy for end-clients to understand and write queries.
- To maximize the efficiency of queries. It achieves these goals by minimizing the number of tables and relationships between them.

Advantages of Dimensional Modeling

- **Dimensional modeling is simple:** Dimensional modeling methods make it possible for warehouse designers to create database schemas that business customers can easily hold and comprehend. There is no need for vast training on how to read diagrams, and there is no complicated relationship between different data elements.
- **Dimensional modeling promotes data quality:** The star schema enable warehouse administrators to enforce referential integrity checks on the data warehouse. Since the fact information key is a concatenation of the essentials of its associated dimensions, a factual record is actively loaded if the corresponding dimensions records are duly described and also exist in the database.

Advantages of Dimensional Modeling

- By enforcing foreign key constraints as a form of referential integrity check, data warehouse DBAs add a line of defense against corrupted warehouses data.

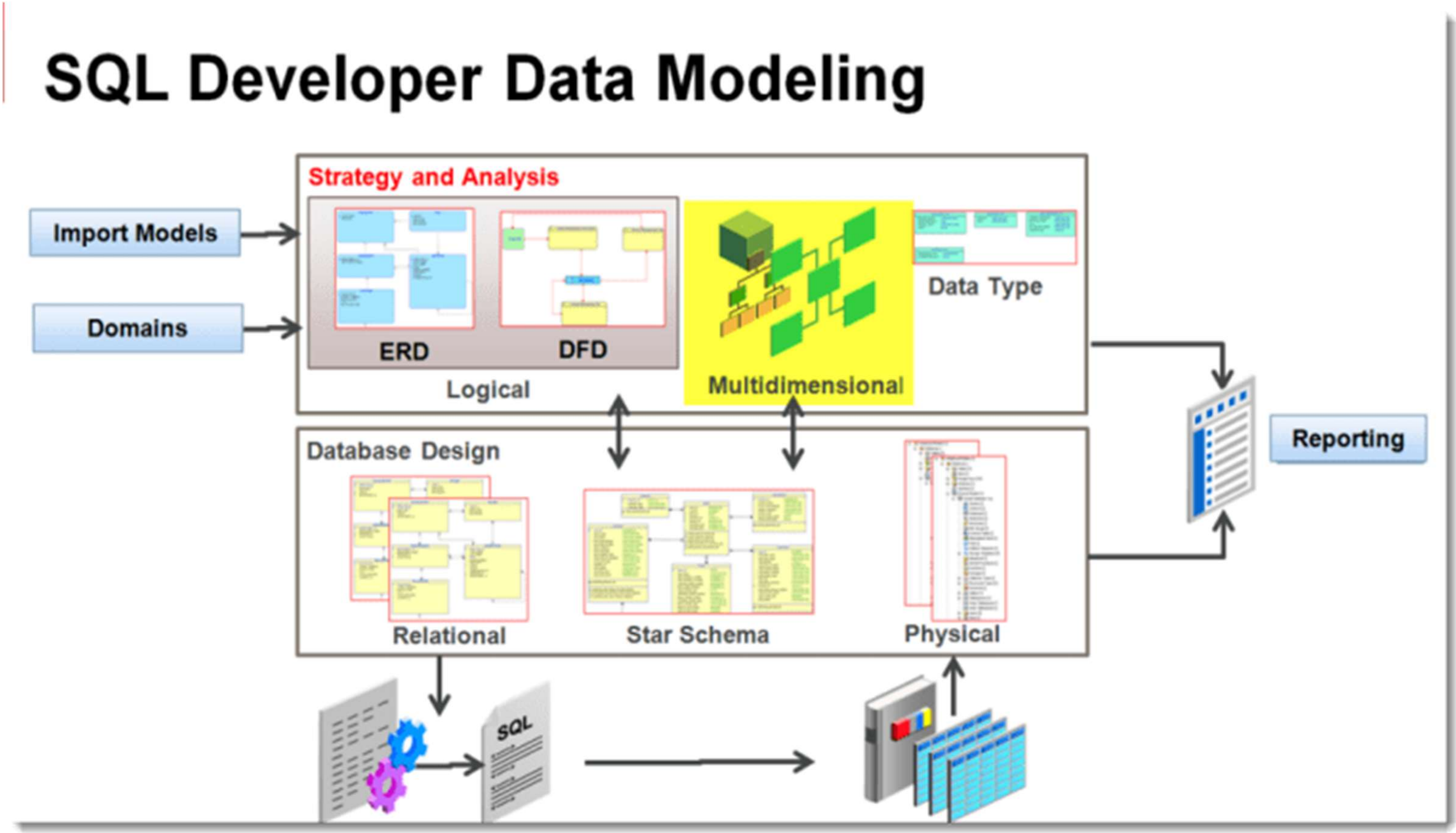
- **Performance optimization is possible through aggregates:** As the size of the data warehouse increases, performance optimization develops into a pressing concern. Customers who have to wait for hours to get a response to a query will quickly become discouraged with the warehouses. Aggregates are one of the easiest methods by which query performance can be optimized.

Disadvantages of Dimensional Modeling

- To maintain the integrity of fact and dimensions, loading the data warehouses with a record from various operational systems is complicated.

- It is severe to modify the data warehouse operation if the organization adopting the dimensional technique changes the method in which it does business.

Objectives of Dimensional Modeling



SQL Developer Data Modeling

Elements of Dimensional Modeling

- **Fact :** It is a collection of associated data items, consisting of measures and context data. It typically represents business items or business transactions.
- **Dimensions :** It is a collection of data which describe one business dimension. Dimensions decide the contextual background for the facts, and they are the framework over which OLAP is performed.
- **Measure :** It is a numeric attribute of a fact, representing the performance or behavior of the business relative to the dimensions.

Elements of Dimensional Modeling

- Considering the relational context, there are two basic models which are used in dimensional modeling:

- **Star Model & Snowflake Model:**

- The star model is the underlying structure for a dimensional model. It has one broad central table (fact table) and a set of smaller tables (dimensions) arranged in a radial design around the primary table. The snowflake model is the conclusion of decomposing one or more of the dimensions.

Elements of Dimensional Modeling

- **Fact :** Facts are the measurements/metrics or facts from your business process. For a Sales business process, a measurement would be quarterly sales number.
- **Dimensions :** Dimension provides the context surrounding a business process event. In simple terms, they give who, what, where of a fact. In the Sales business process, for the fact quarterly sales number, dimensions would be
- Who – Customer Names/ Where – Location /What – Product Name
- In other words, a dimension is a window to view information in the facts.
- **Attributes:** The Attributes are the various characteristics of the dimension in dimensional data modeling. In the Location dimension, the attributes can be State, Country, Zipcode etc. Attributes are used to search, filter, or classify facts. Dimension Tables contain Attributes
- .

Elements of Dimensional Modeling

- **Fact Table :** A fact table is a primary table in dimension modelling.
- A Fact Table contains Measurements/facts, Foreign key to dimension table
- **Dimension Table:** A dimension table contains dimensions of a fact.
- They are joined to fact table via a foreign key.
- Dimension tables are de-normalized tables.
- The Dimension Attributes are the various columns in a dimension table
- Dimensions offers descriptive characteristics of the facts with the help of their attributes
- No set limit set for given for number of dimensions
- The dimension can also contain one or more hierarchical relationships

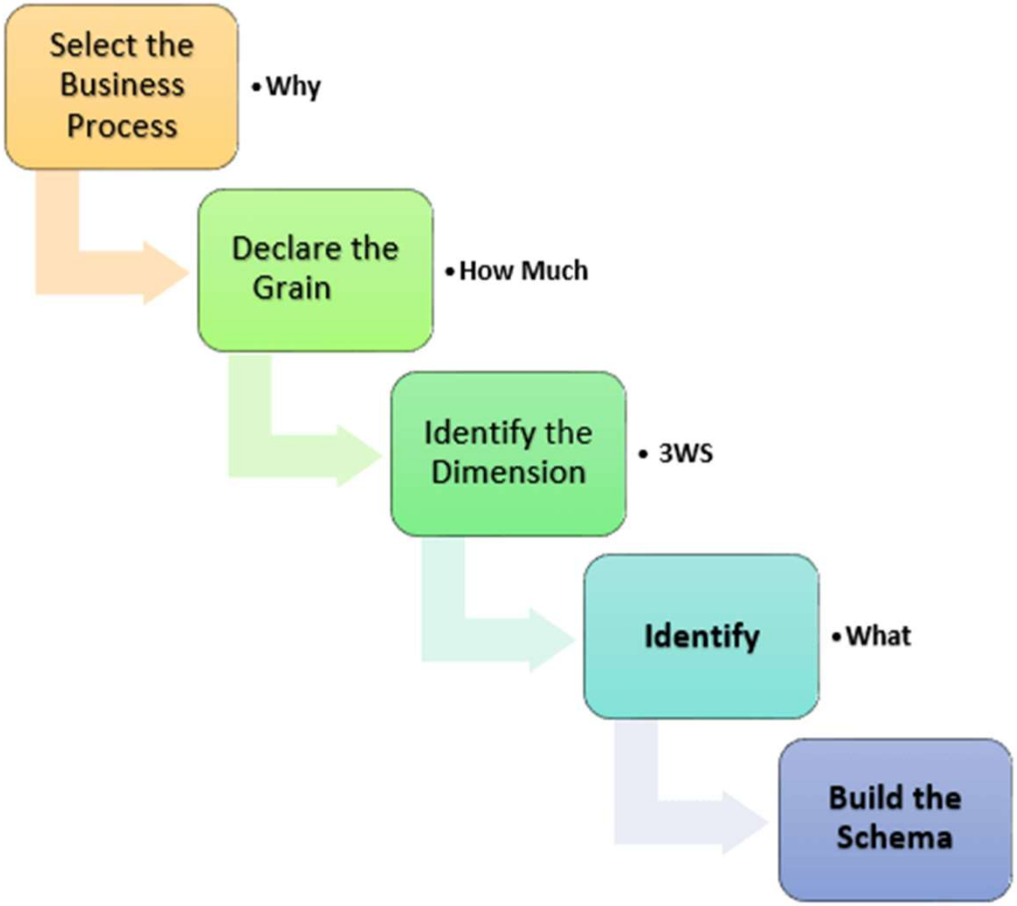
Steps of Dimensional Modelling

- The accuracy in creating your Dimensional modeling determines the success of your data warehouse implementation. Here are the steps to create Dimension Model:

1. Identify Business Process
2. Identify Grain (level of detail)
3. Identify Dimensions
4. Identify Facts
5. Build Star

- The model should describe the Why, How much, When/Where/Who and What of your business process

Steps of Dimensional Modelling



Select the Business Process
- Why

Declare the Grain
- How Much

Identify the Dimension
- 3WS

Identify
- What

Build the Schema

Steps of Dimensional Modelling

- **Step 1) Identify the Business Process**
- Identifying the actual business process a data warehouse should cover. This could be Marketing, Sales, HR, etc. as per the data analysis needs of the organization. The selection of the Business process also depends on the quality of data available for that process. It is the most important step of the Data Modelling process, and a failure here would have cascading and irreparable defects.
- To describe the business process, you can use plain text or use basic Business Process Modelling Notation (BPMN) or Unified Modelling Language (UML).

Steps of Dimensional Modelling

- **Step 2) Identify the Grain**
- The Grain describes the level of detail for the business problem/solution. It is the process of identifying the lowest level of information for any table in your data warehouse. If a table contains sales data for every day, then it should be daily granularity. If a table contains total sales data for each month, then it has monthly granularity.
- During this stage, you answer questions like
- Do we need to store all the available products or just a few types of products? This decision is based on the business processes selected for Data warehouse. Do we store the product sale information on a monthly, weekly, daily or hourly basis? This decision depends on the nature of reports requested by executives. How do the above two choices affect the database size?

Steps of Dimensional Modelling

- **Step 3) Identify the Dimensions**
- Dimensions are nouns like date, store, inventory, etc. These dimensions are where all the data should be stored. For example, the date dimension may contain data like a year, month and weekday.
- **Example of Dimensions:** The CEO at an MNC wants to find the sales for specific products in different locations on a daily basis.
- Dimensions: Product, Location and Time
- Attributes: For Product: Product key (Foreign Key), Name, Type, Specifications
- Hierarchies: For Location: Country, State, City, Street Address, Name

Steps of Dimensional Modelling

- **Step 4) Identify the Fact**
- This step is co-associated with the business users of the system because this is where they get access to data stored in the data warehouse. Most of the fact table rows are numerical values like price or cost per unit, etc.
- **Example of Facts:**
- The CEO at an MNC wants to find the sales for specific products in different locations on a daily basis.
- The fact here is Sum of Sales by product by location by time.

Steps of Dimensional Modelling

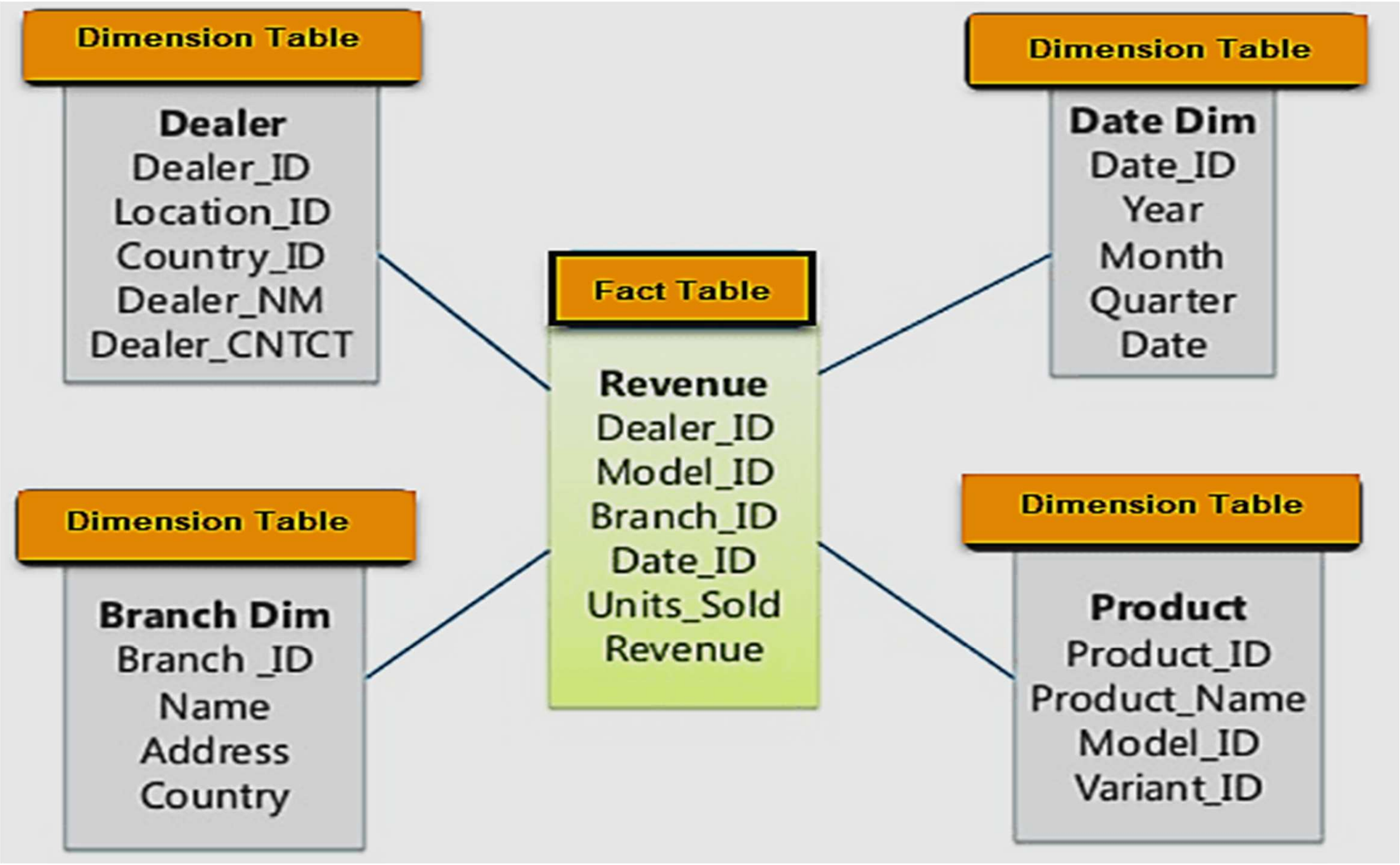
- **Step 5) Build Schema**
- In this step, you implement the Dimension Model. A schema is nothing but the database structure (arrangement of tables). There are two popular schemas
- **Star Schema:** The star schema architecture is easy to design. It is called a star schema because diagram resembles a star, with points radiating from a center. The center of the star consists of the fact table, and the points of the star is dimension tables.
- The fact tables in a star schema which is third normal form whereas dimensional tables are de-normalized.
- **Snowflake Schema:** The snowflake schema is an extension of the star schema. In a snowflake schema, each dimension are normalized and connected to more dimension tables.

Star Schema

- **Star Schema** in data warehouse, in which the center of the star can have one fact table and a number of associated dimension tables. It is known as star schema as its structure resembles a star. The Star Schema data model is the simplest type of Data Warehouse schema. It is also known as Star Join Schema and is optimized for querying large data sets.

- In the following Star Schema example, the fact table is at the center which contains keys to every dimension table like Dealer_ID, Model ID, Date_ID, Product_ID, Branch_ID & other attributes like Units sold and revenue.

Star Schema



Dimension Table

Dealer
Dealer_ID
Location_ID
Country_ID
Dealer_NM
Dealer_CNTCT

Dimension Table

Date Dim
Date_ID
Year
Month
Quarter
Date

Fact Table

Revenue
Dealer_ID
Model_ID
Branch_ID
Date_ID
Units_Sold
Revenue

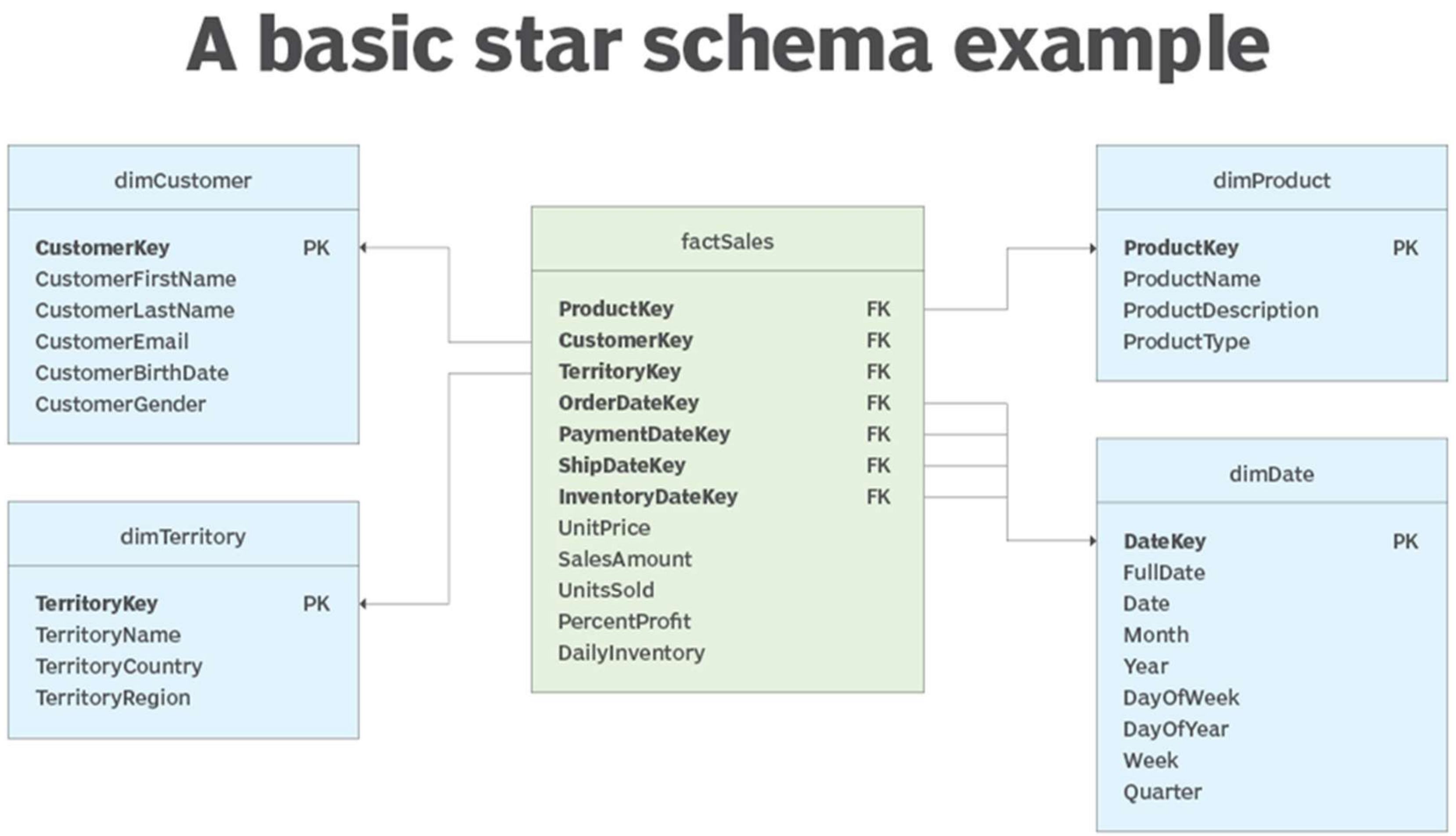
Dimension Table

Branch Dim
Branch _ID
Name
Address
Country

Dimension Table

Product
Product_ID
Product_Name
Model_ID
Variant_ID

Example2



A basic star schema example

dimCustomer	
CustomerKey	PK
CustomerFirstName	
CustomerLastName	
CustomerEmail	
CustomerBirthDate	
CustomerGender	

dimTerritory	
TerritoryKey	PK
TerritoryName	
TerritoryCountry	
TerritoryRegion	

factSales	
ProductKey	FK
CustomerKey	FK
TerritoryKey	FK
OrderDateKey	FK
PaymentDateKey	FK
ShipDateKey	FK
InventoryDateKey	FK
UnitPrice	
SalesAmount	
UnitsSold	
PercentProfit	
DailyInventory	

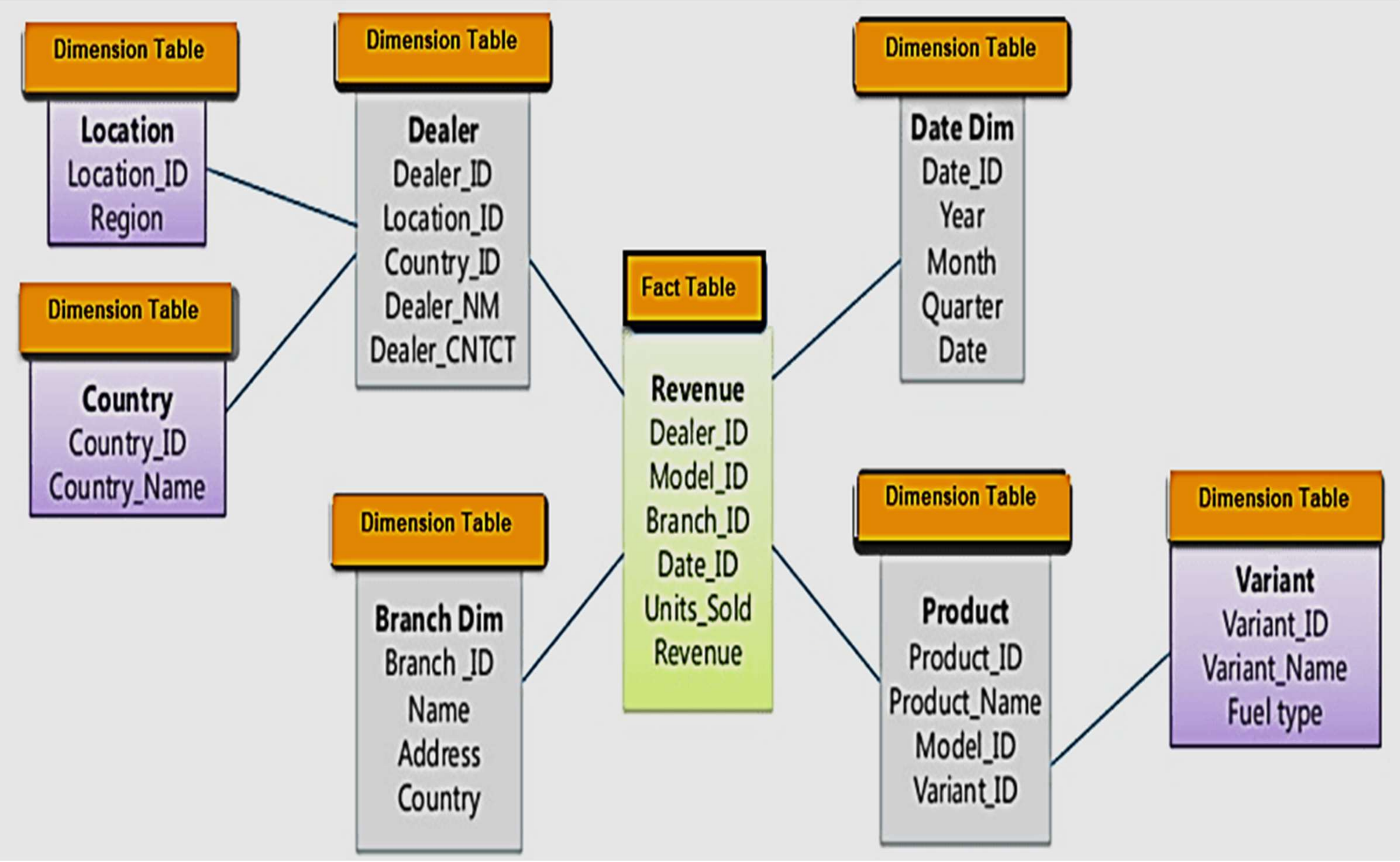
dimProduct	
ProductKey	PK
ProductName	
ProductDescription	
ProductType	

dimDate	
DateKey	PK
FullDate	
Date	
Month	
Year	
DayOfWeek	
DayOfYear	
Week	
Quarter	

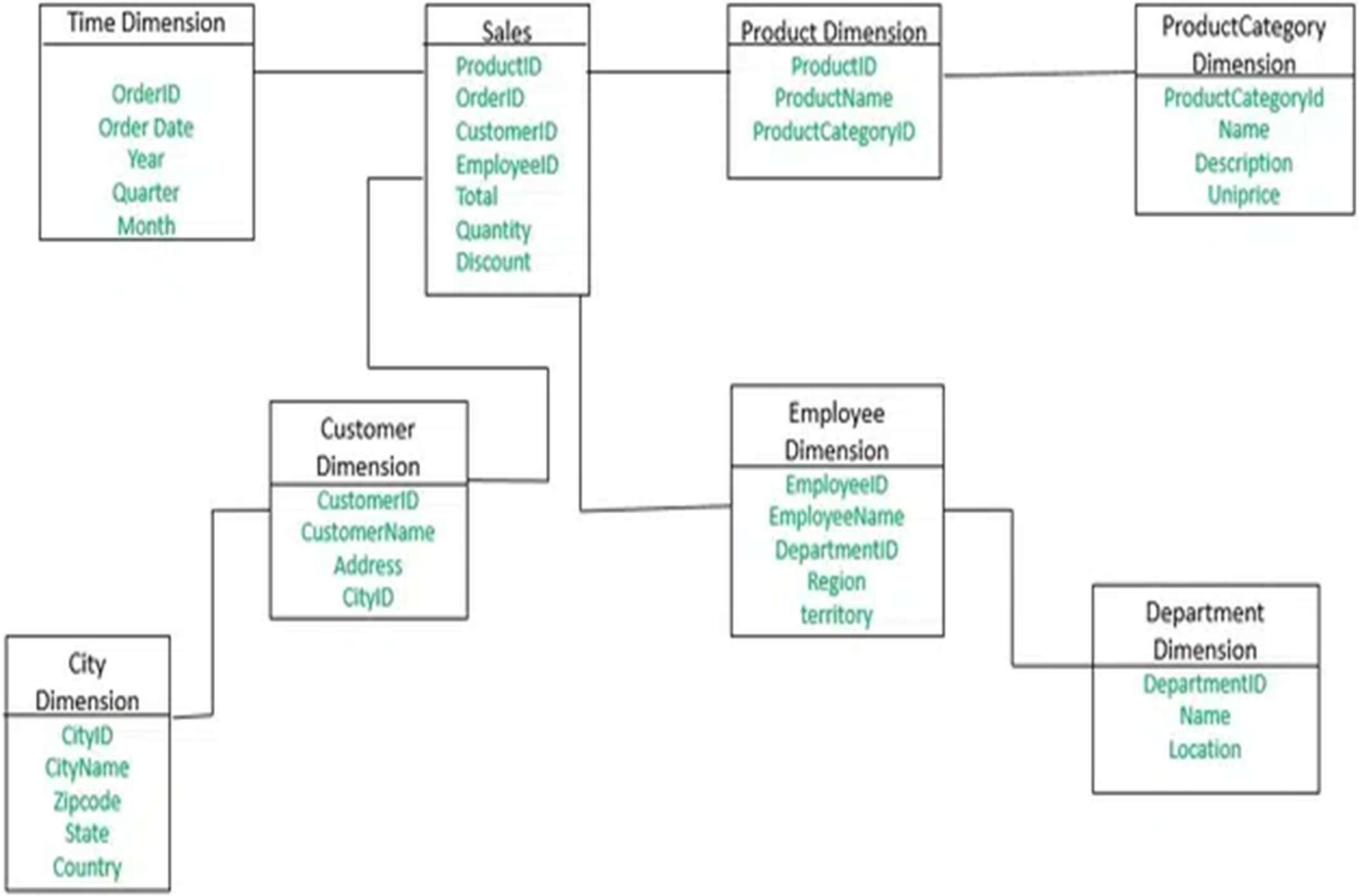
Snowflake Schema

- **Snowflake Schema** in data warehouse is a logical arrangement of tables in a multidimensional database such that the ER diagram resembles a snowflake shape. A Snowflake Schema is an extension of a Star Schema, and it adds additional dimensions. The dimension tables are normalized which splits data into additional tables.
- In the following Snowflake Schema example, Country is further normalized into an individual table.

Snowflake Schema



Example: 2



Time Dimension

OrderID
Order Date
Year
Quarter
Month

Sales

ProductID
OrderID
CustomerID
EmployeeID
Total
Quantity
Discount

Product Dimension

ProductID
ProductName
ProductCategoryID

ProductCategory Dimension

ProductCategoryId
Name
Description
Uniprice

Customer Dimension

CustomerID
CustomerName
Address
CityID

Employee Dimension

EmployeeID
EmployeeName
DepartmentID
Region
territory

City Dimension

CityID
CityName
Zipcode
State
Country

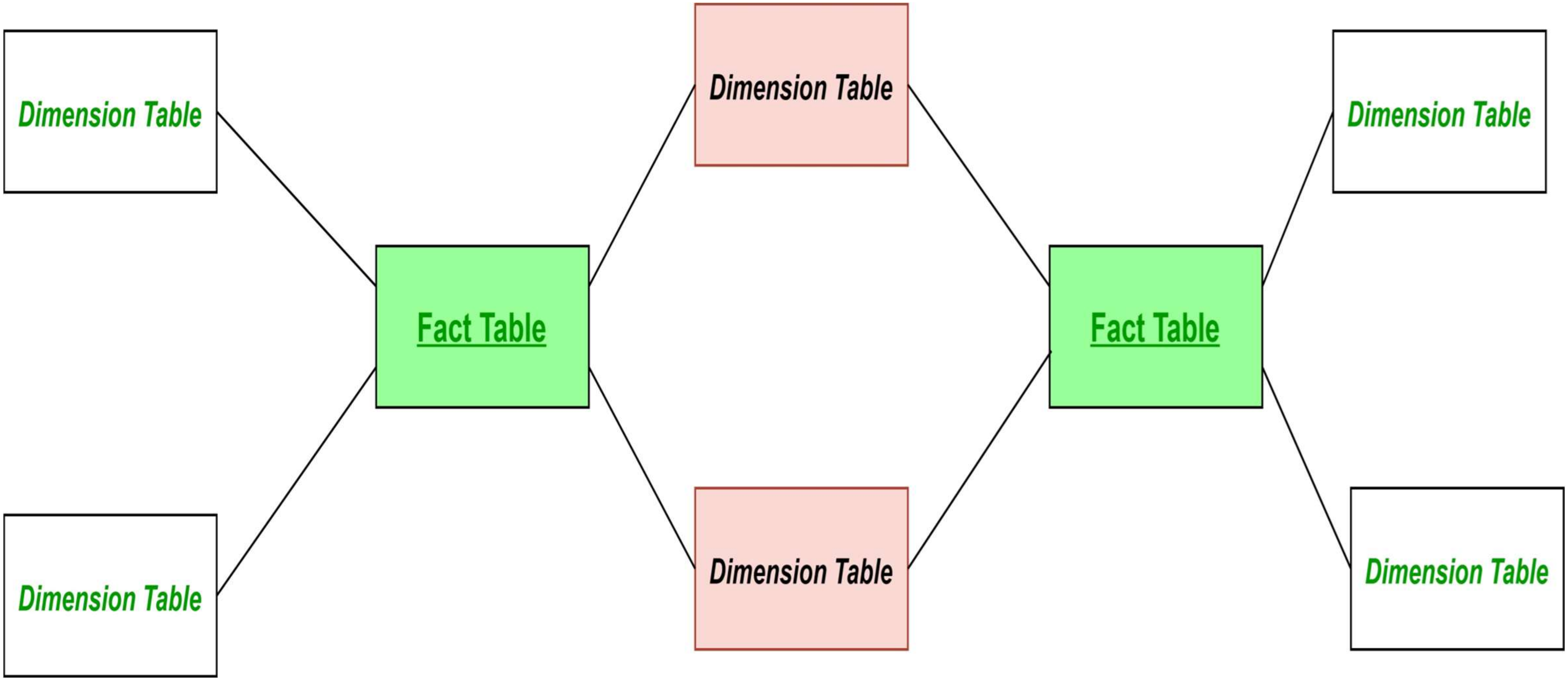
Department Dimension

DepartmentID
Name
Location

Multi-Dimensional Data Modelling

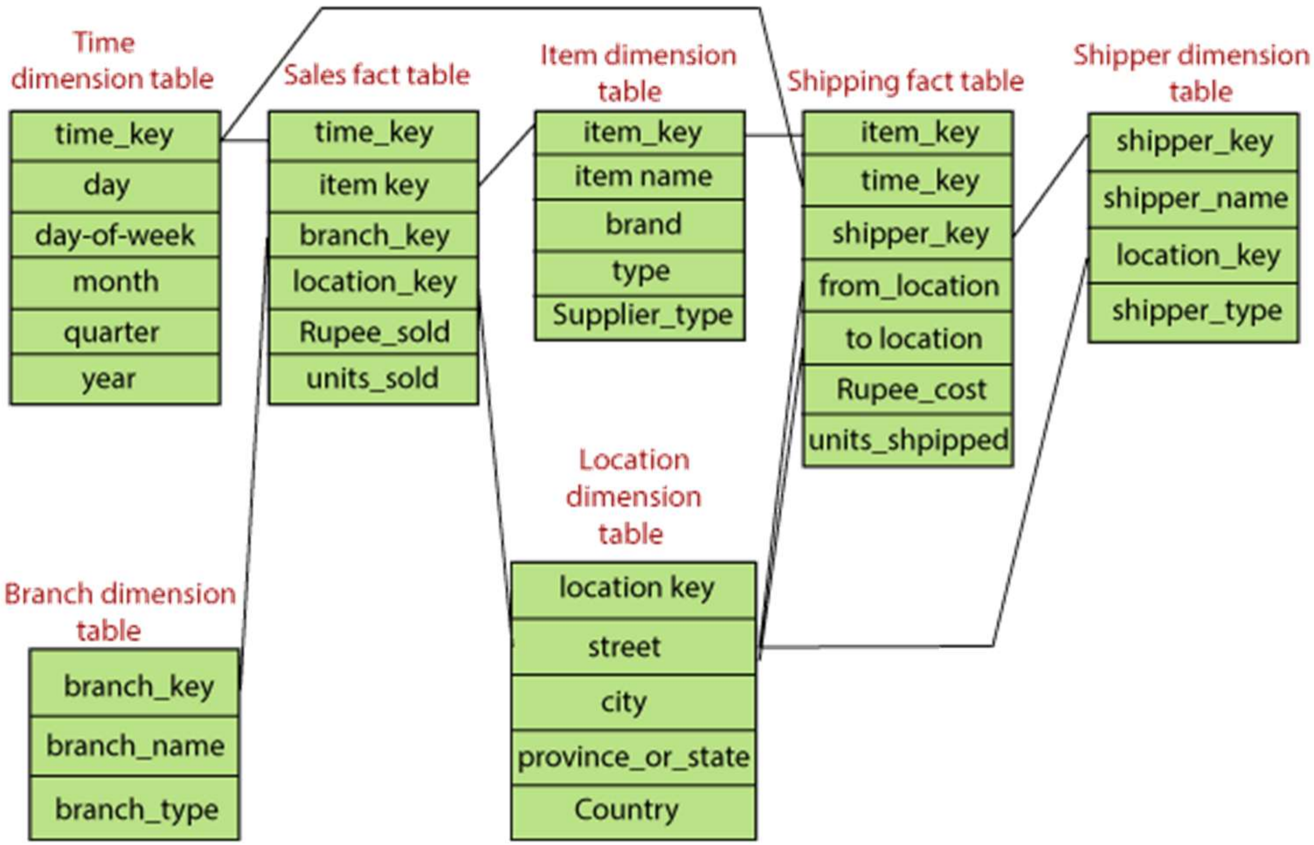
- **Multidimensional data model** in data warehouse is a model which represents data in the form of data cubes. It allows to model and view the data in multiple dimensions and it is defined by dimensions and facts.
- Multidimensional data model is generally categorized around a central theme and represented by a fact table.
- **Fact Constellation** is a schema for representing multidimensional model. It is a collection of multiple fact tables having some common dimension tables. It can be viewed as a collection of several star schemas and hence, also known as Galaxy schema. It is one of the widely used schema for Data warehouse designing and it is much more complex than star and snowflake schema. For complex systems, we require fact constellations.

Fact Constellation



- Here, the pink colored Dimension tables are the common ones among both the star schemas. Green colored fact tables are the fact tables of their respective star schemas.
- <u>General Structure of Fact Constellation</u>

Example



Time dimension table
time_key
day
day-of-week
month
quarter
year

Sales fact table
time_key
item key
branch_key
location_key
Rupee_sold
units_sold

Item dimension table
item_key
item name
brand
type
Supplier_type

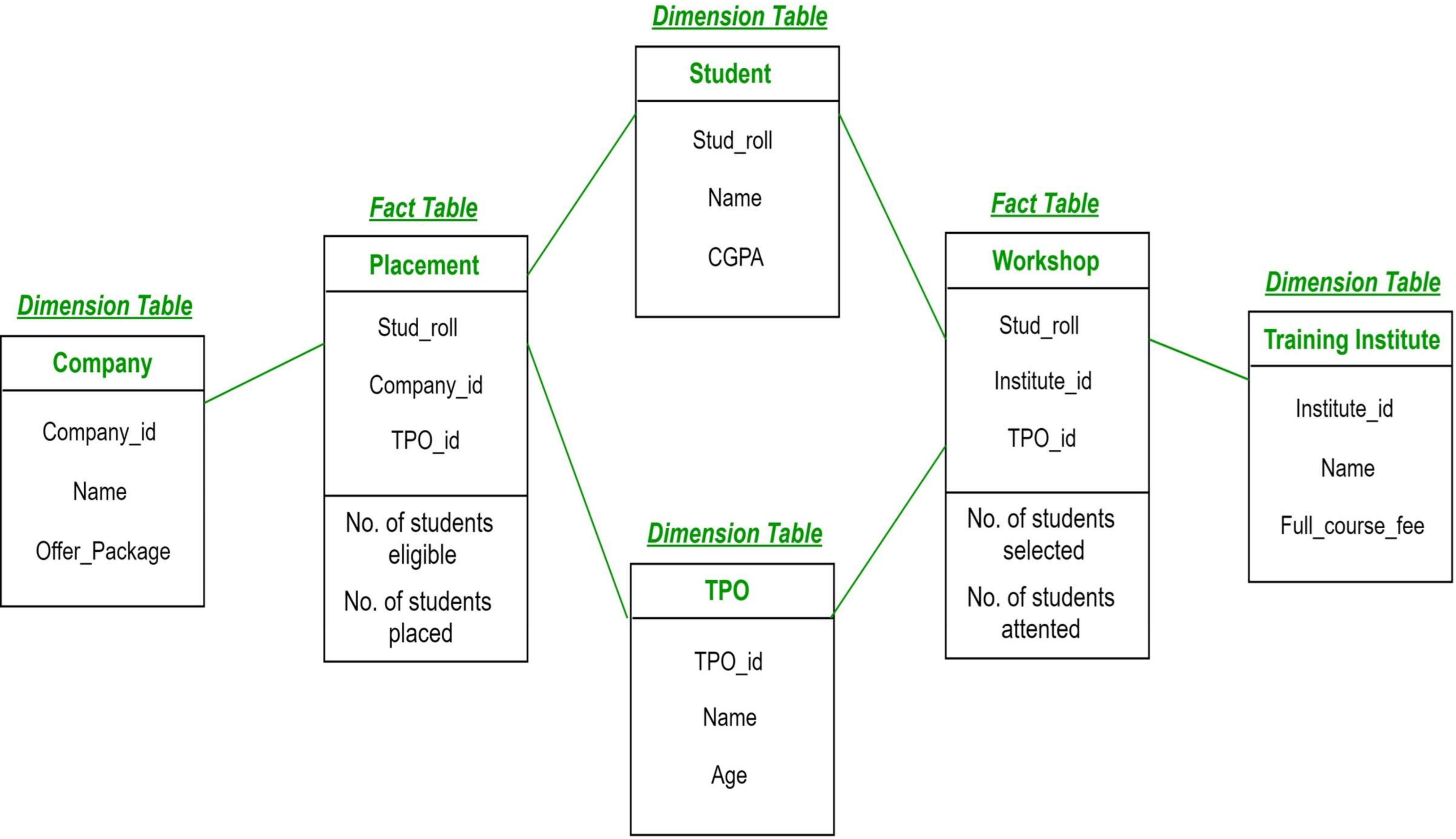
Shipping fact table
item_key
time_key
shipper_key
from_location
to location
Rupee_cost
units_shpipped

Shipper dimension table
shipper_key
shipper_name
location_key
shipper_type

Branch dimension table
branch_key
branch_name
branch_type

Location dimension table
location key
street
city
province_or_state
Country

Snowflake Schema



Dimension Table

Company

Company_id

Name

Offer_Package

Fact Table

Placement

Stud_roll

Company_id

TPO_id

No. of students eligible

No. of students placed

Dimension Table

Student

Stud_roll

Name

CGPA

Dimension Table

TPO

TPO_id

Name

Age

Fact Table

Workshop

Stud_roll

Institute_id

TPO_id

No. of students selected

No. of students attented

Dimension Table

Training Institute

Institute_id

Name

Full_course_fee

ER VS DM

ER Modeling	**Dimensional Modeling**
It is transaction-oriented.	It is subject-oriented.
Entities and Relationships.	Fact Tables and Dimension Tables.
Few levels of granularity.	Multiple levels of granularity.
Real-time information.	Historical information.
It eliminates redundancy.	It plans for redundancy.
High transaction volumes	Low transaction volumes using
using few records at a time.	many records at a time.
Highly Volatile data.	Non-volatile data.
Physical and Logical Model.	Physical Model.
Normalization is suggested.	De-Normalization is suggested.
OLTP Application.	OLAP Application.