Business Intelligence (BI) and Data Warehousing (DW)

Dimensional Model Schemas

Jackline Ssanyu

February 4, 2020

Introduction to Dimensional Schemas

- A database is comprised of one or more tables, and the relationships among all the tables in the database is collectively called the **database schema**.
- Although there are many different schema designs, databases that query historical data usually use a **dimensional schema design.**
- Use dimensional modeling to achieve the following benefits:
  - To create queries that answer business questions. Typically, a query calculates some measure of performance over several business dimensions.
  - You can create SQL queries. The SQL language is used by most RDBMS vendors.
- A dimensional schema physically separates the measures that quantify the business from the descriptive elements (also called dimensions) that describe and categorize the business.

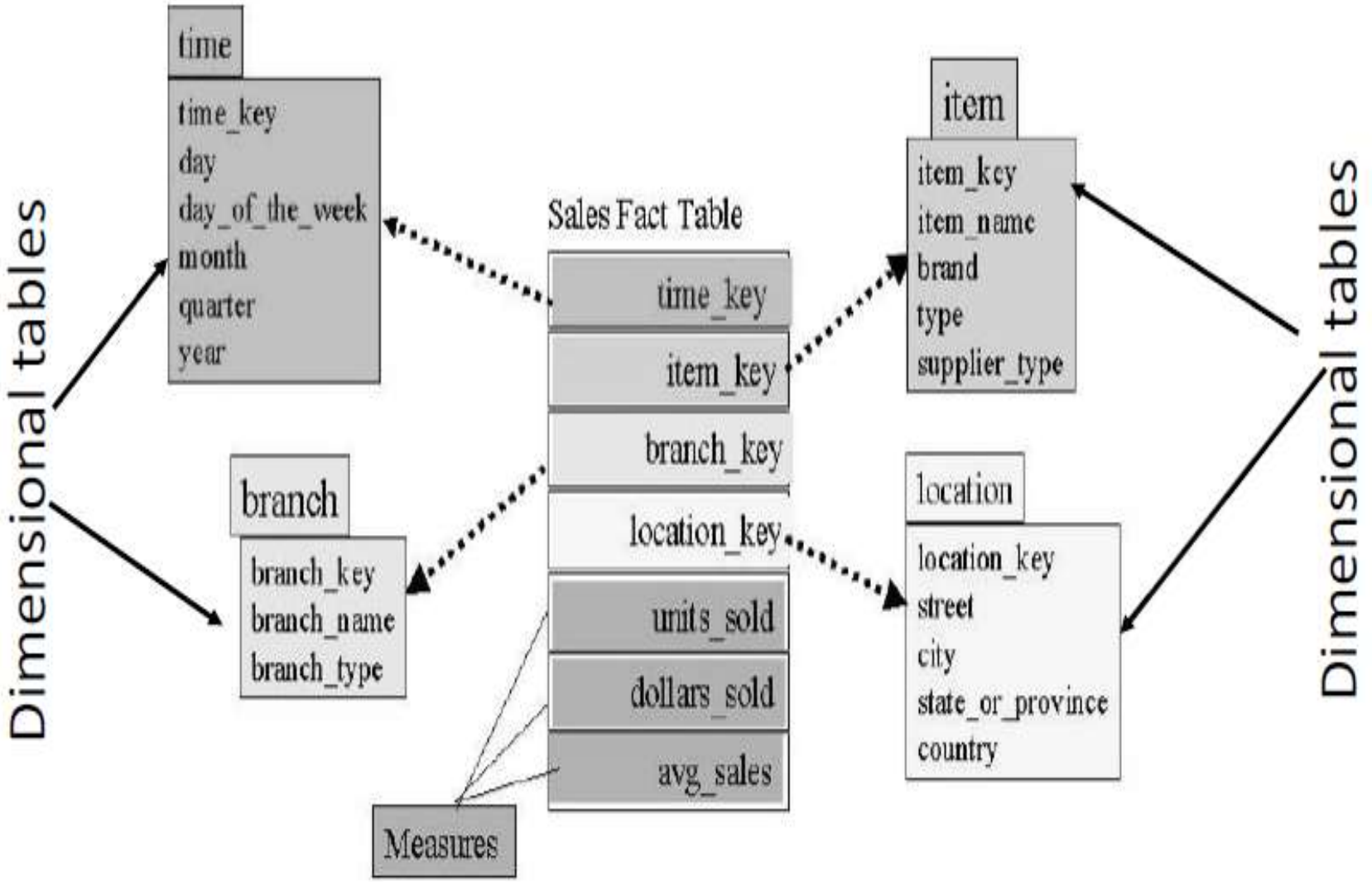
Introduction to Dimensional Schemas...

- The dimensional schema is a physical or logical schema.
  - A physical dimensional schema is typically represented in the form of a **star, snowflake or fact constellation** schema, where the objects in the star , snowflake or fact constellation schema are actually database tables.
  - In a logical dimensional schema, the fact, measures, and dimensions are represented as entities and attributes that are independent of a database vendor and can therefore be transformed to a physical dimensional schema for any database vendor.

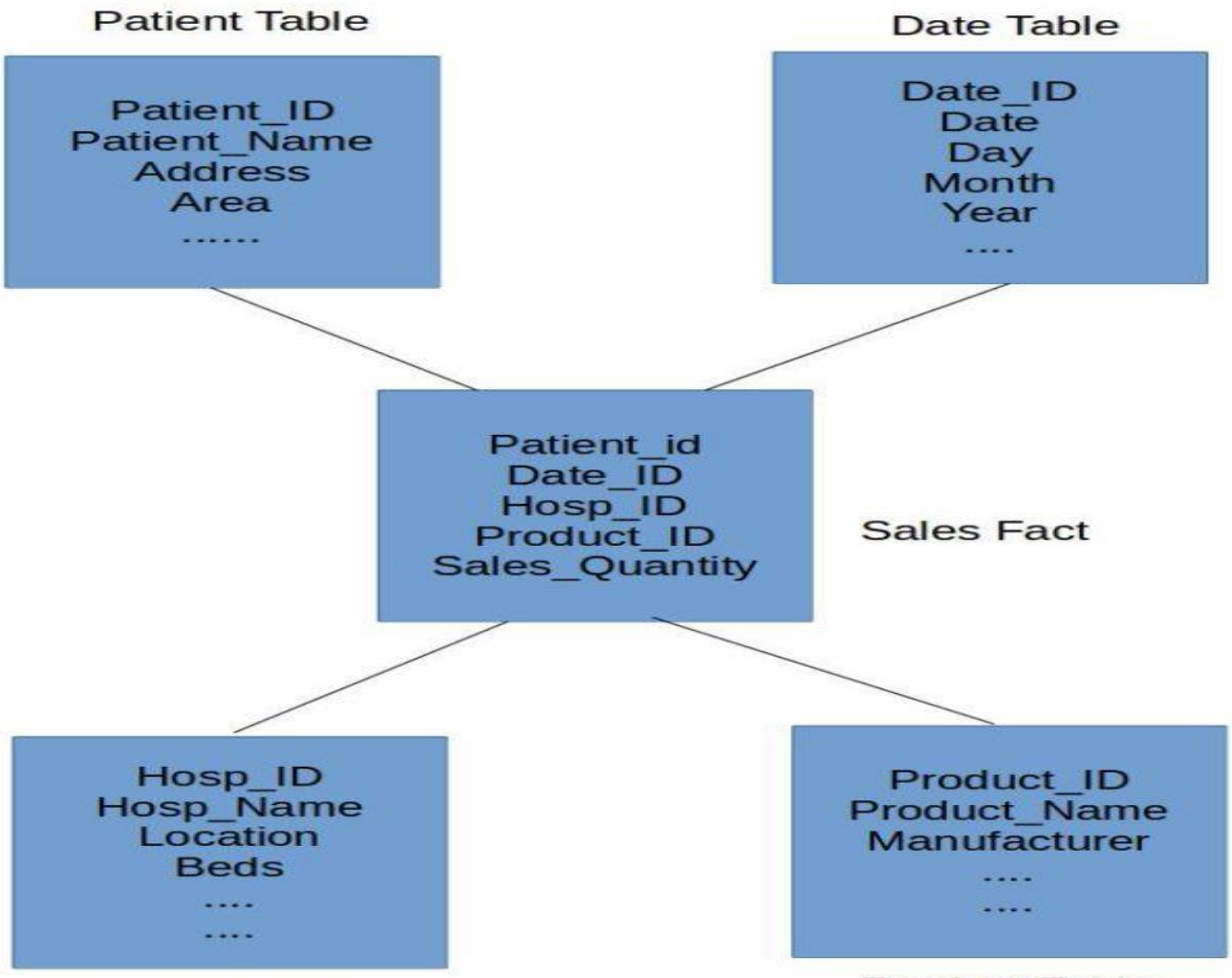
Star Schema

- The star schema is a data-modeling technique used to map multidimensional decision support data into a relational database.
- In effect, the star schema creates the near equivalent of a multidimensional database schema from the existing relational database.
- The star schema was developed because existing relational modeling techniques, ER, and normalization did not yield a database structure that served advanced data analysis requirements well.
- The basic star schema has four components: facts, dimensions, attributes, and attribute hierarchies.
- A star query is a join between a fact table and a number of dimension tables; a one to many relationship.
- Facts and dimensions are normally represented by physical tables in the data warehouse database.

Star Schema - Example 1



Star Schema - Example2



Characteristics of a Star Schema

- Dimensional Tables are not normalized.
- The dimension table should contain the set of attributes.
- Dimension table is joined to only Fact tables. They are not joined to each other.
- Fact table stores keys from dimension tables and measure.

Star Schema Advantages

Star schemas are easy for end users and applications to understand and navigate. With a well-designed schema, users can quickly analyze large, multidimensional data sets. The main advantages of star schemas in a decision-support environment are:

- **Optimizes Navigation:**
  - The purpose of defining the relationship among entities in the database schema is to provide you the ability to navigate through the database. The relationship can be used to move between tables to obtain the information you wish to find.
  - If the paths from one table to another are numerous and complicated, the navigation process will become complex and slow. But if the join paths are straightforward, the navigation process is simple, optimized and faster.
  - This is where the Star Schema shines brightly. This method optimizes the navigation through the database, and even if a complex query result is expected, the navigation is simple and uncomplicated.

Star Schema Advantages...

- **Load performance and administration** - Structural simplicity also reduces the time required to load large batches of data into a star schema database. By defining facts and dimensions and separating them into different tables, the impact of a load operation is reduced. Dimension tables can be populated once and occasionally refreshed. You can add new facts regularly and selectively by appending records to a fact table.
- **Feeding Cubes:** The Star Schema is used by all Online Analytical Processing (OLAP) systems to build OLAP cubes resourcefully. In fact, most OLAP systems provide a mode of operation called ROLAP which uses the Star Schema directly as a source without having to build a proprietary cube structure.

Star Schema Advantages...

- **Built-in referential integrity**
  - A star schema has referential integrity built in when data is loaded. Referential integrity is enforced because each record in a dimension table has a unique primary key, and all keys in the fact tables are legitimate foreign keys drawn from the dimension tables.
  - A record in the fact table that is not related correctly to a dimension cannot be given the correct key value to be retrieved.

Star Schema Advantages...

- **Easily understood**
  - The users of OLTP (Online Transaction Processing) systems work with the application through a predefined Graphical User Interface or preset query templates.
  - Due to this, the users do not have to understand data structures going on behind the scenes. The database schema and data structures remain with IT professionals.
  - However, the users of decision support systems will themselves formulate queries and if and when they try to interact with the data warehouse, they must have an understanding of data structures and understand when which data will be available.
  - A star schema is easy to understand and navigate, with dimensions joined only through the fact table.

Star Schema Disadvantages

- The main disadvantages is, that data integrity is not enforced due to its denormalized state. Inserts and updates can result in data anomalies, which normalized schemas, as in OLTP databases, are designed to avoid.
- This model don't support many to many relationship between business entities. These types of relationships are simplified in star schema

Performance-Improving Techniques for the Star Schema

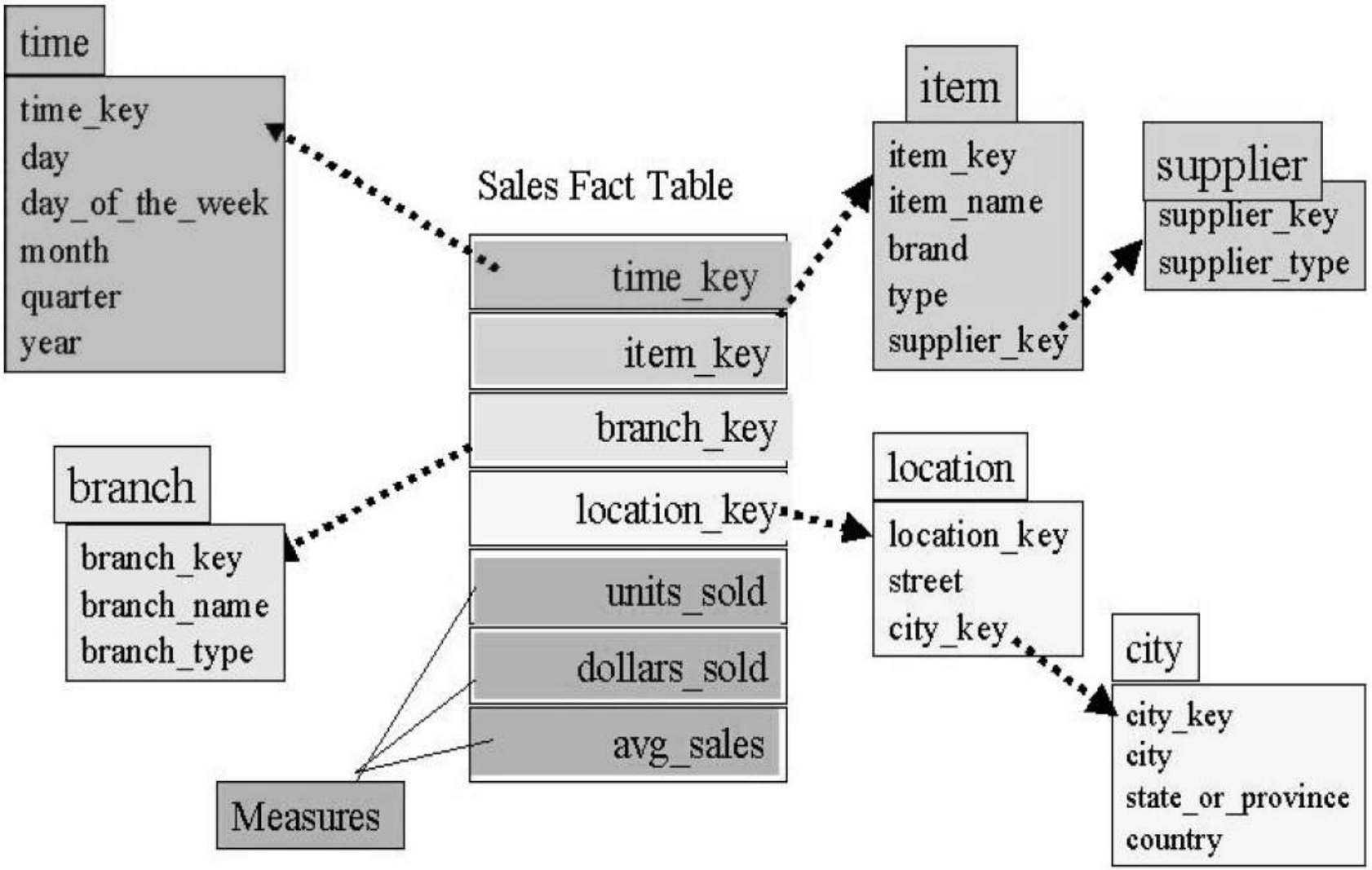
The most common techniques often used to optimize data warehouse design for query speed are:

- Normalizing dimensional tables.
- Maintaining multiple fact tables to represent different aggregation levels.

Normalizing Dimensional Tables

- Dimensional tables are normalized to eliminate redundancy. That is, the dimension data has been grouped into multiple tables instead of one large table.
- Normalization is done by looking for dimension tables containing transitive dependencies and revising those relationships to the 3NF (third normal form).
- A normalized star schema is called a snow flake.
- For example, in the previous star schema figure:
  - a location dimension table might be normalized into a location table and city table in a snow flake schema.
  - item dimension table might be normalized into item table and supplier table in a snow flake schema.
- While this saves space, it increases the number of dimension tables and requires more foreign key joins. The result is more complex queries and reduced query performance.

Resulting Snow Flake



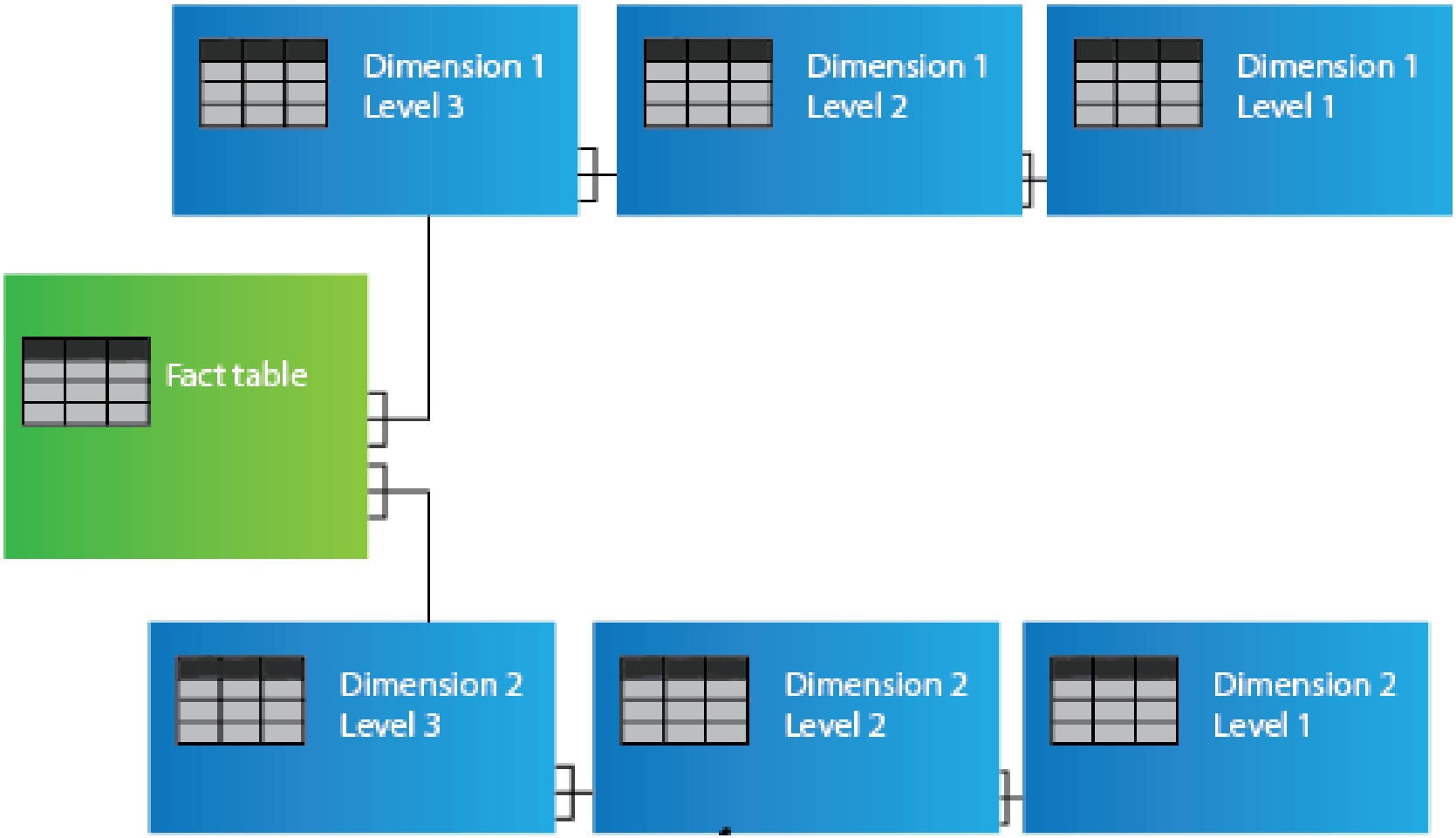
Snow Flake

- a snowflake schema is a logical arrangement of tables in a multidimensional database such that the entity relationship diagram resembles a snowflake shape. The snowflake schema is represented by centralized fact tables which are connected to multiple dimensions.
- "Snowflaking" is a method of normalizing the dimension tables in a star schema. When it is completely normalized along all the dimension tables, the resultant structure resembles a snowflake with the fact table in the middle. The principle behind snowflaking is normalization of the dimension tables and forming separate tables.
- The snowflake schema is similar to the star schema. However, in the snowflake schema, dimensions are normalized into multiple related tables, whereas the star schema's dimensions are denormalized with each dimension represented by a single table.

Snow Flake...

- Normalization splits up data to avoid redundancy (duplication) by moving commonly repeating groups of data into new tables.
- Normalization therefore tends to increase the number of tables that need to be joined in order to perform a given query, but reduces the space required to hold the data and the number of places where it needs to be updated if the data changes.

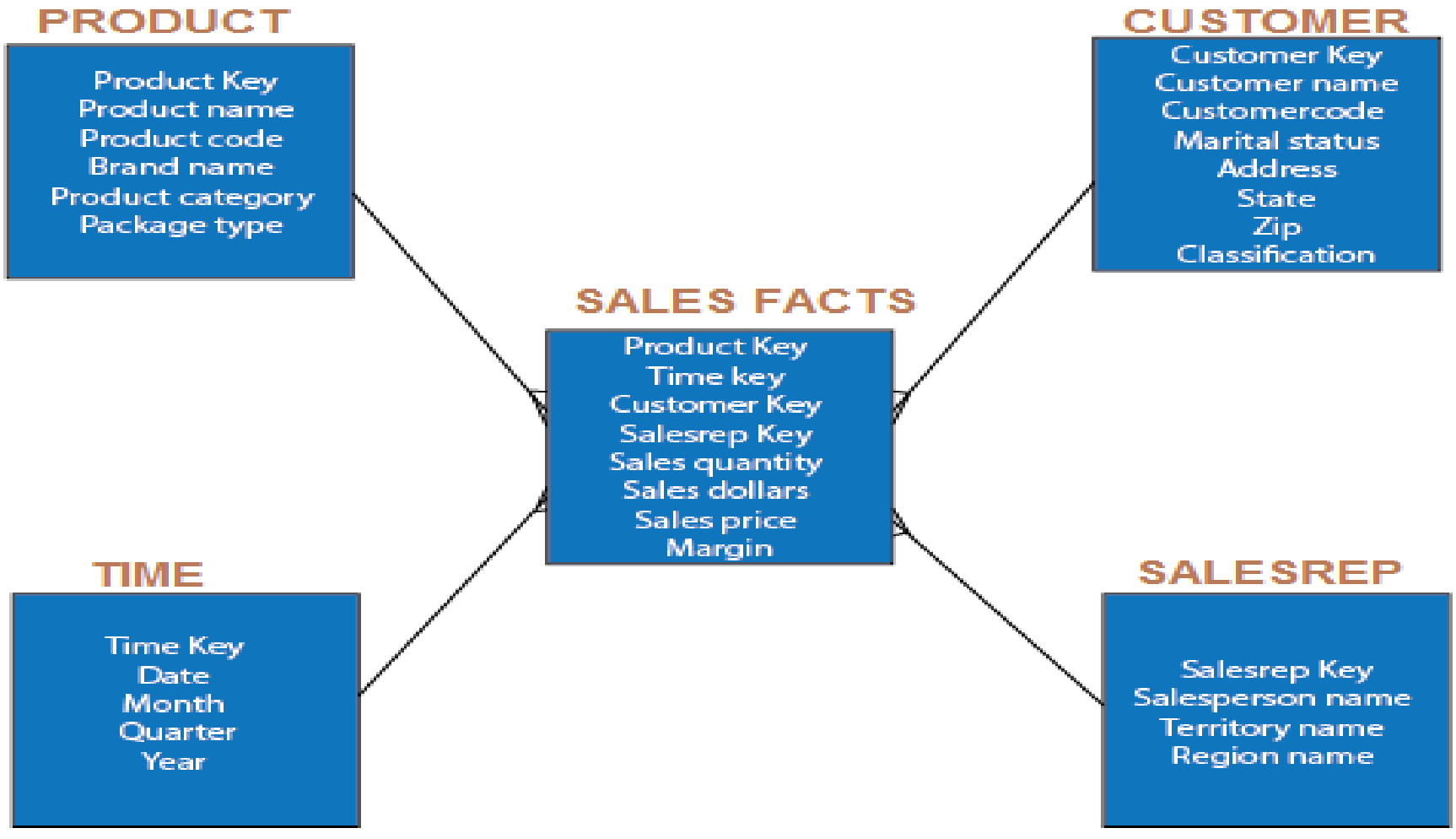
General Diagram of a Snow Flake



Snow Flake...

- The diagram in the previous slide shows a snowflake schema with two dimensions, each having three levels. A snowflake schemas can have any number of dimension, and each dimension can have any number of levels.

Star schema to Snow Flake - Example

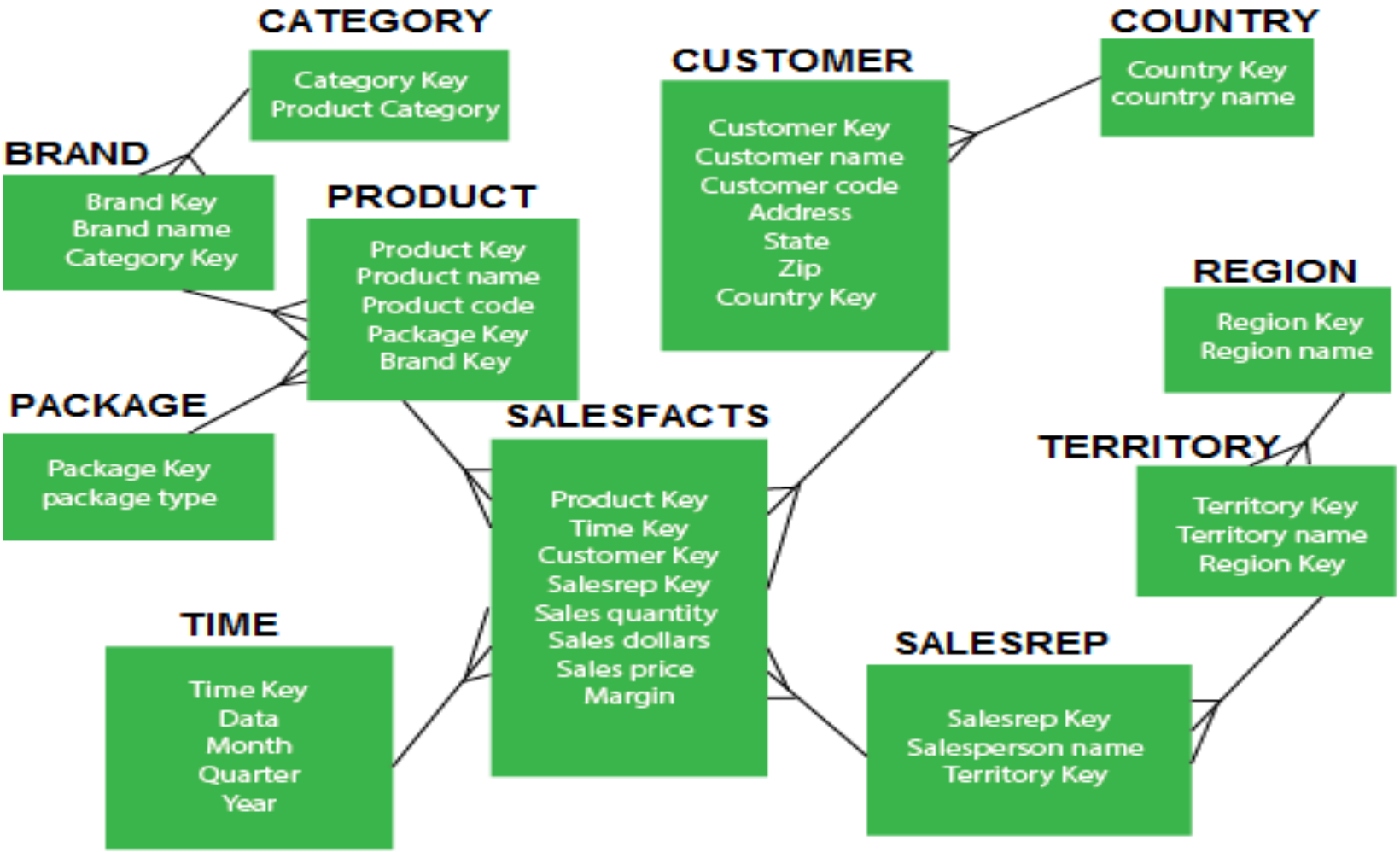


STAR Schema

Star schema to Snow Flake - Example...

- Figure shows a simple STAR schema for sales in a manufacturing company. The sales fact table include quantity, price, and other relevant metrics. SALESREP, CUSTOMER, PRODUCT, and TIME are the dimension tables.
- The STAR schema for sales, as shown above, contains only five tables, whereas the normalized version now extends to eleven tables. We will notice that in the snowflake schema, the attributes with low cardinality in each original dimension tables are removed to form separate tables. These new tables are connected back to the original dimension table through artificial keys.
- See the resulting snow flake in the next slide.

Star schema to Snow Flake - Example...



Benefits of Snow Flake

The snowflake schema provides some advantages over the star schema in certain situations, including:

- The primary advantage of the snowflake schema is the development in query performance due to minimized disk storage requirements and joining smaller lookup tables. Normalizing attributes results in storage savings.
- No redundancy, so it is easier to maintain.

Disadvantages of Snow Flake

The snowflake schema provides some advantages over the star schema in certain situations, including:

- The primary disadvantage of the snowflake schema is that the additional levels of attribute normalization adds complexity to source query joins difficult to understand, when compared to the star schema.
- Additional maintenance efforts are required due to the increasing number of lookup tables.
- More tables more join so more query execution time.

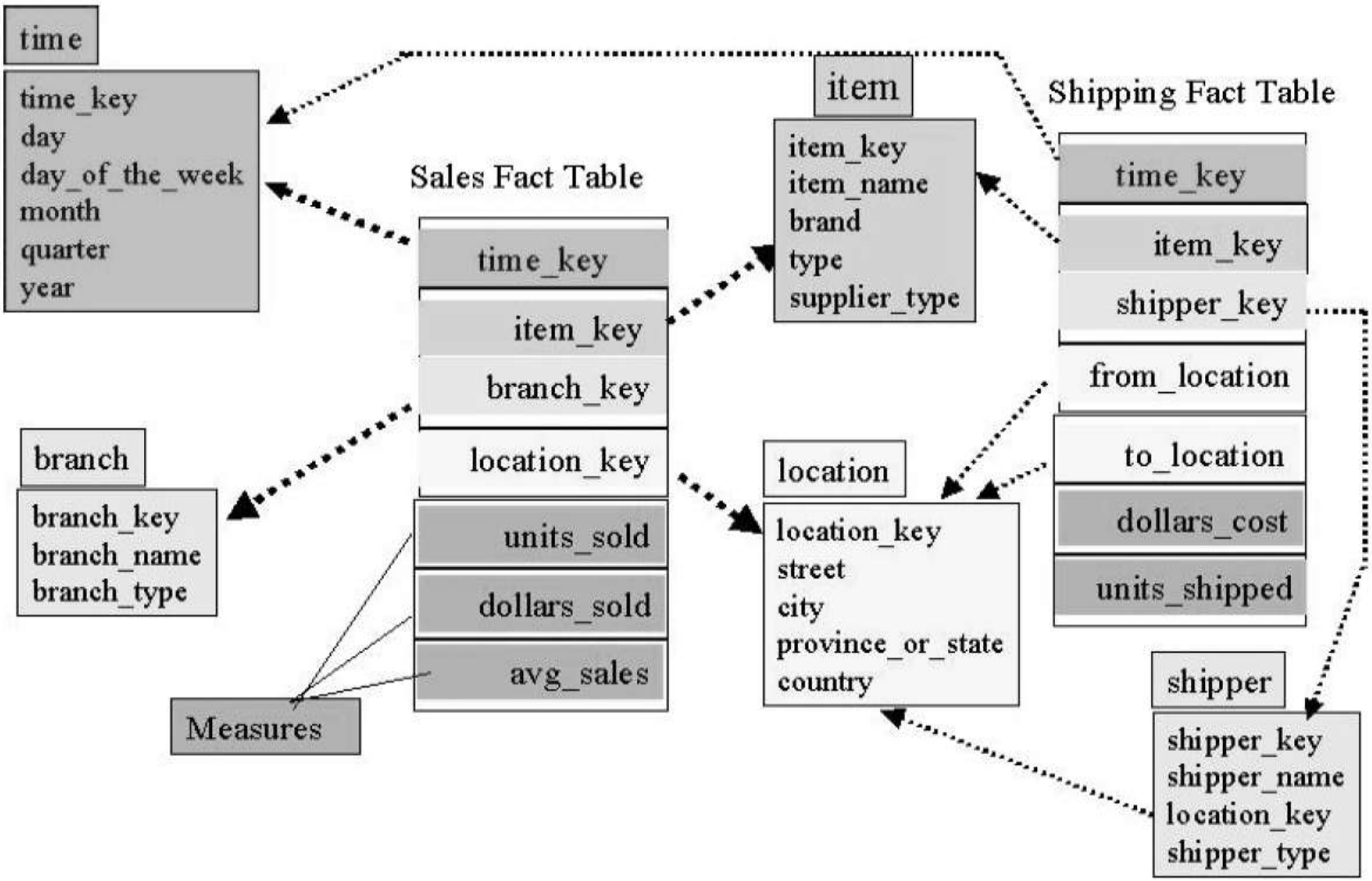
Maintaining Multiple Fact Tables That Represent Different Aggregation Levels

- You can also speed up query operations by creating and maintaining multiple fact tables related to each level of aggregation
- This produces a Fact Constellation schema.
- For each star schema it is possible to construct fact constellation schema (for example by splitting the original star schema into more star schemes each of them describes facts on another level of dimension hierarchies).
- The fact constellation architecture contains multiple fact tables that share many dimension tables.

Maintaining Multiple Fact Tables That Represent Different Aggregation Levels...

- For data ware houses, the fact constellation schema is commonly used, since it can model multiple interrelated subjects.
- A data mart on the other hand, is a departmental subset set of data warehouse that focuses on selected subjects and thus its scope is department-wide. For data marts, a star or snowflake is commonly used, since both are geared towards modeling single subjects, although a star schema is more popular and efficient.

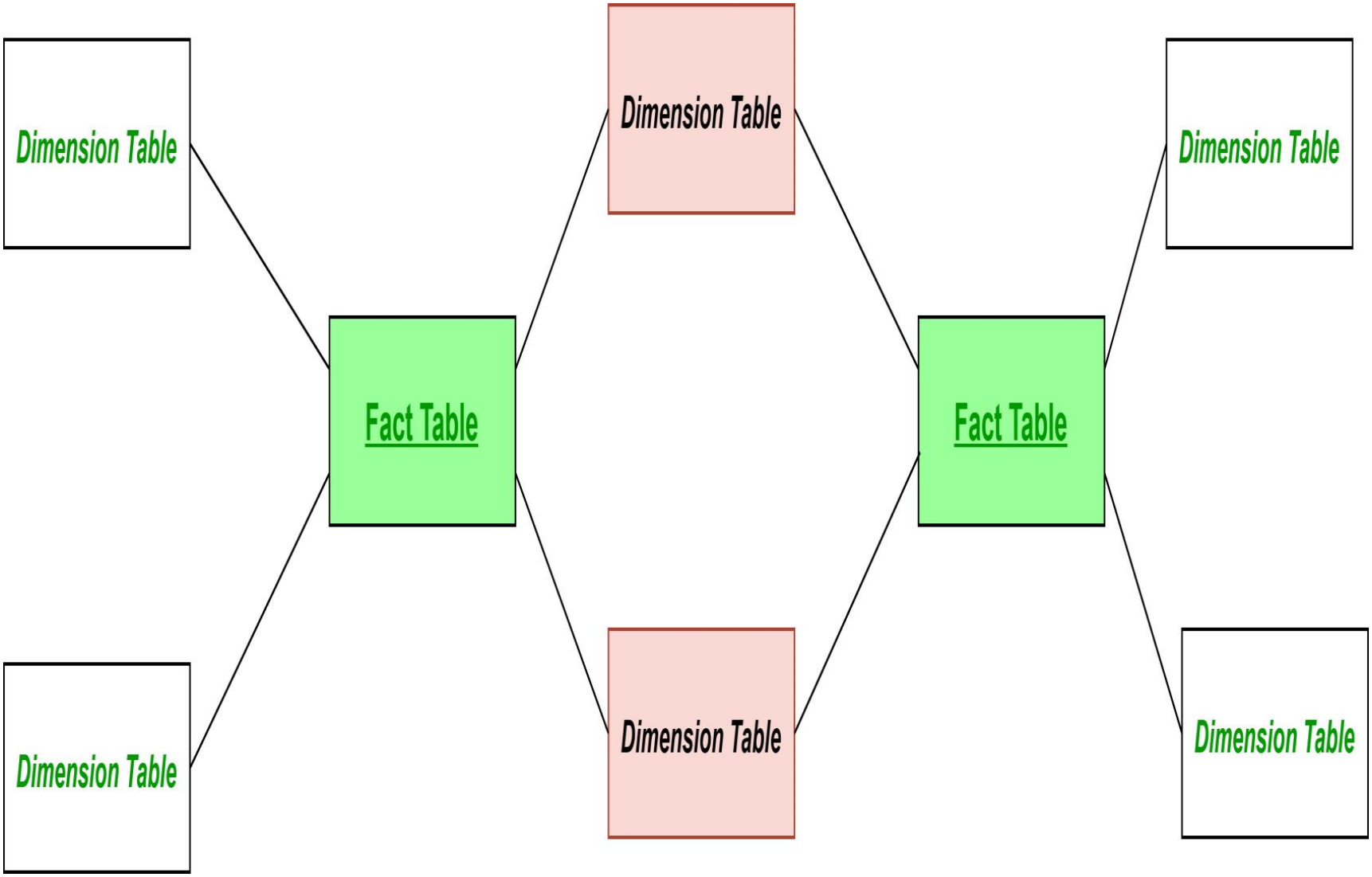
Fact Constellation- Example 1



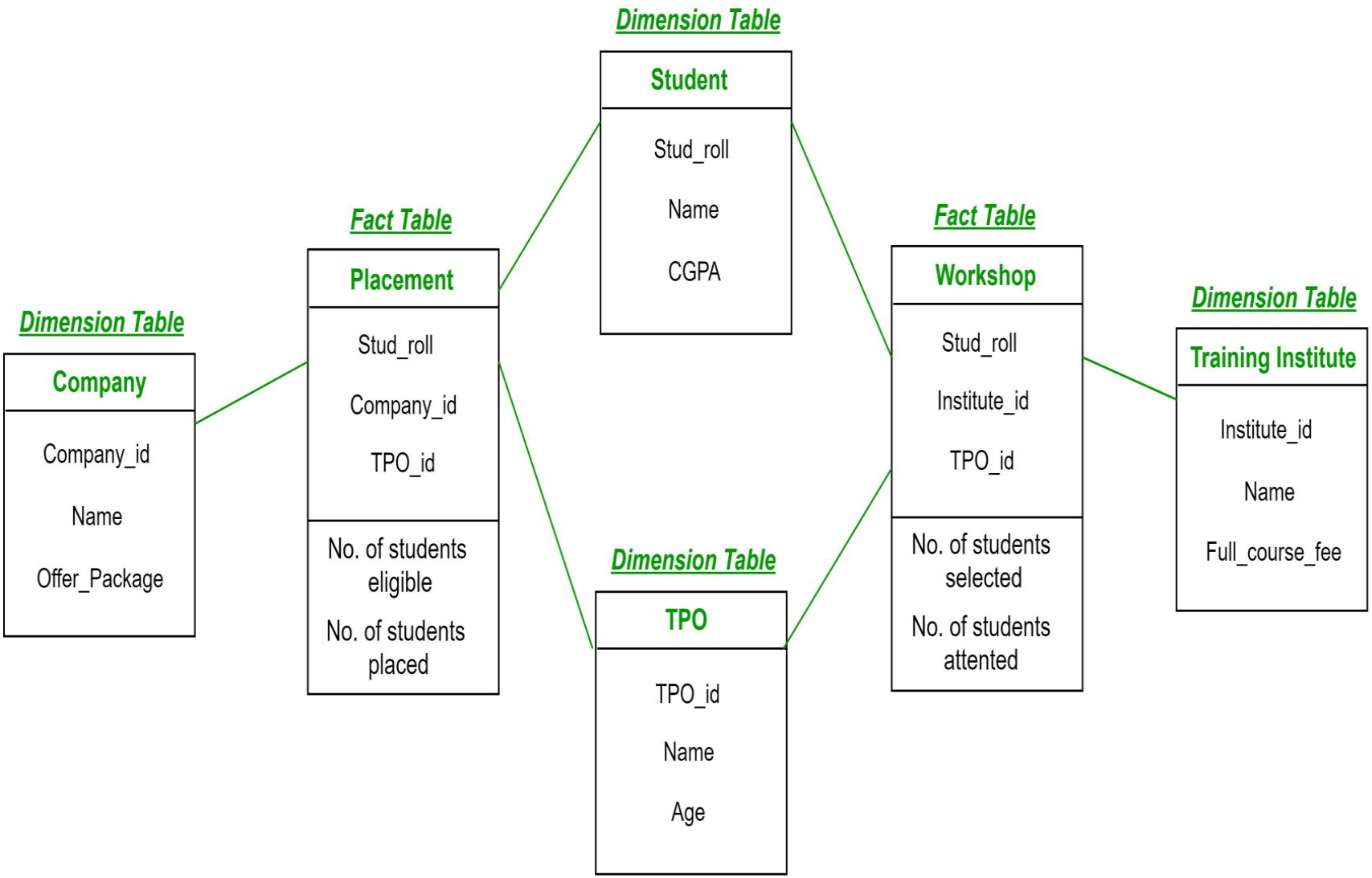
Fact Constellation

- A Fact constellation means two or more fact tables sharing one or more dimensions. It is also called Galaxy schema.
- **Advantage:** Provides a flexible schema.
- **Disadvantage:** It is much more complex and hence, hard to implement and maintain.

General Structure of Fact Constellation



Fact Constellation- Example 2



Fact Constellation- Example 2 ...

In above demonstration:

- Placement is a fact table having attributes: (Stud_roll, Company_id, TPO_id) with facts: (Number of students eligible, Number of students placed).
- Workshop is a fact table having attributes: (Stud_roll, Institute_id, TPO_id) with facts: (Number of students selected, Number of students attended the workshop).
- Company is a dimension table having attributes: (Company_id, Name, Offer_package).
- Student is a dimension table having attributes: (Student_roll, Name, CGPA).
- TPO is a dimension table having attributes: (TPO_id, Name, Age).
- Training Institute is a dimension table having attributes: (Institute_id, Name, Full_course_fee).

Converting an E/R model to a dimensional model

- A dimensional model can be created from the enterprise data warehouse or directly from OLTP source systems.
- The following are the steps for converting an E/R model to a dimensional model:
  - Identify the business process from the E/R model.
  - Identify many-to-many tables in the E/R model to convert to fact tables.
  - Denormalize remaining tables into flat dimension tables.
  - Identify date and time from the E/R model.

Identify the business process from the E/R model

- An E/R model consists of several business processes.
- Therefore, it can be segmented into multiple dimensional models.
- For example, an E/R model for an ERP system includes several business processes, such as retail sales, order management, procurement, inventory, and store and warehouse inventory management.

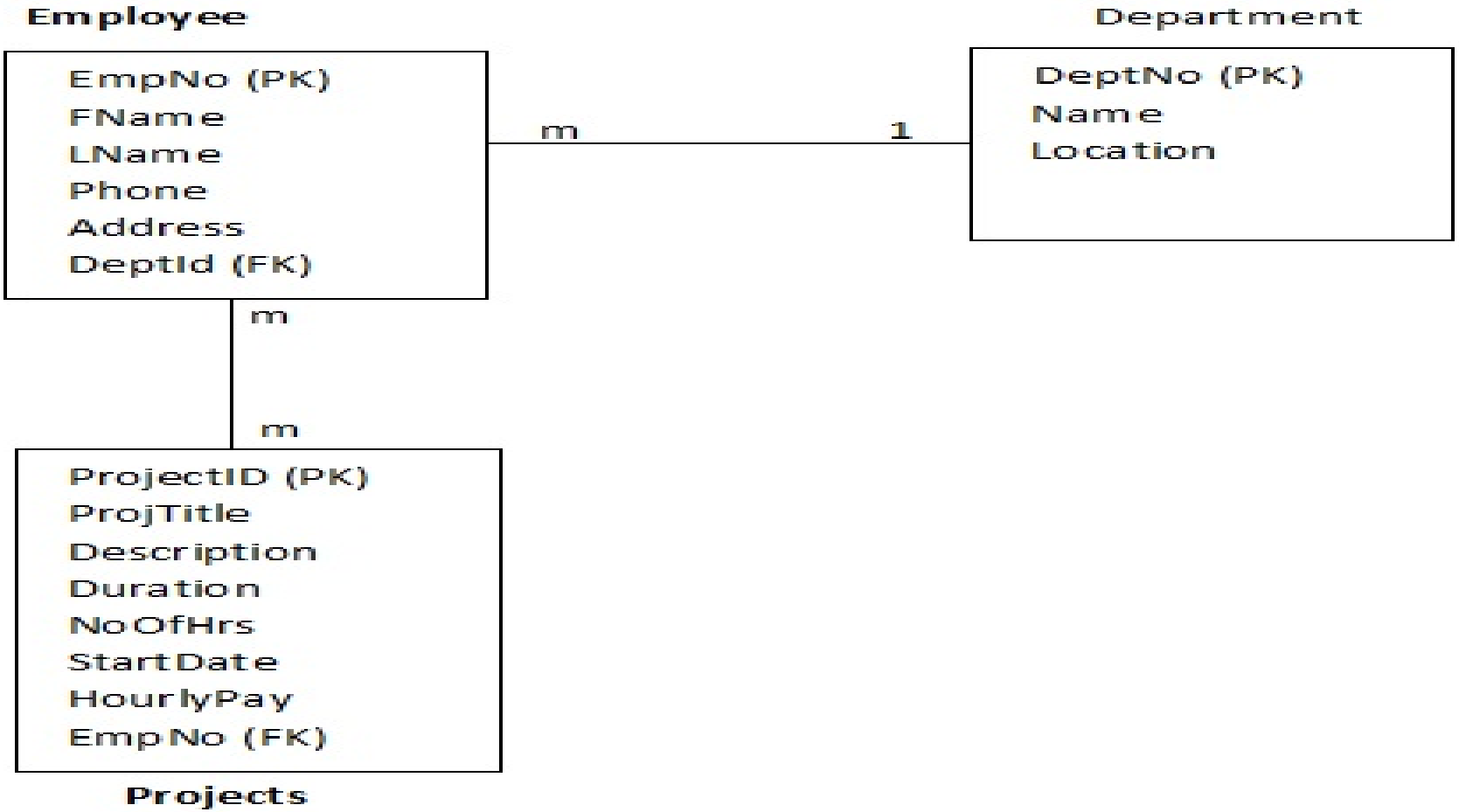
Identify many-to-many tables in E/R model

- Once the business processes are separated, the next step is to identify the many-to-many tables (many-to-many relationships) in the E/R model and convert them to dimensional model fact tables.
- These many-to-many relationships contain numeric and additive non-key facts which generally become facts inside the fact table.
- The idea behind this step is to identify the transaction-based tables that serve to express many-to-many relationships inside an E/R model.
- Every E/R model consists of transaction-based tables which constantly have data inserted, or are updated with data, or have data deleted from them.

Identify many-to-many tables in E/R model

- Some of these tables also express a many-to-many relationship.
- For example, in an ERP database, there are transaction tables, such as Invoice and Invoice_Details, which are constantly inserted and updated because they are transaction-based tables.
- However, tables such as Employee and Products in an E/R model may be fairly static.
- Many-to-many (m:n) relationships add complexity and confusion to the model and to the application development process.

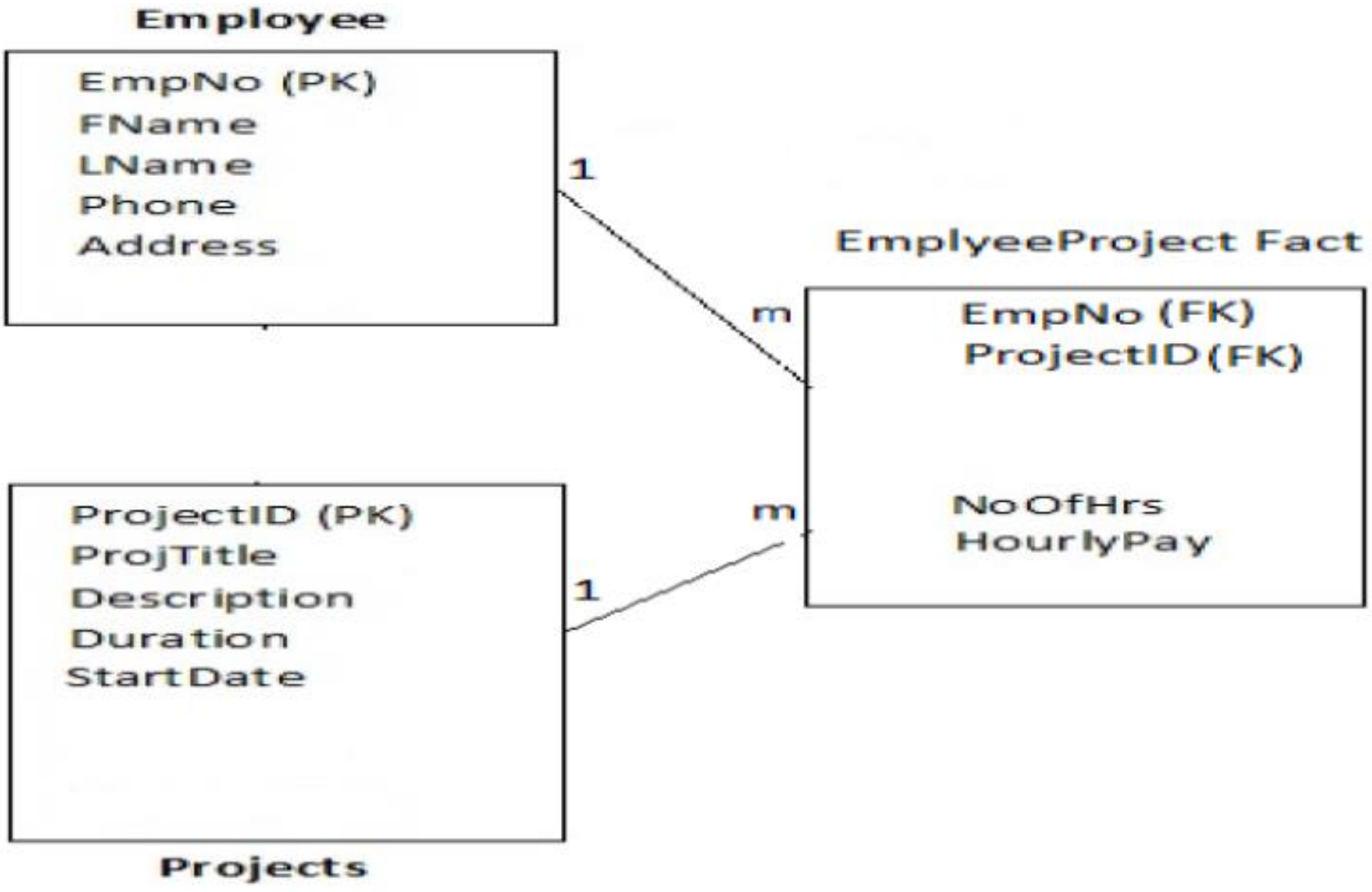
Example with M:M Relationship



Identify many-to-many tables in E/R model

- The key to resolving m:n relationships is to separate the two entities and create two one-to-many (1:n) relationships between them with a third intersect entity.
- The intersect entity usually contains attributes from both connecting entities.
- The example in the previous slide has a many to many relationship between Employee and Projects entities.
- To resolve this m:m relationship, you can add an intersect entity between the Employee and projects entities, as depicted in the next slide.

Conversion of M:M Relationship to Dimensional Model



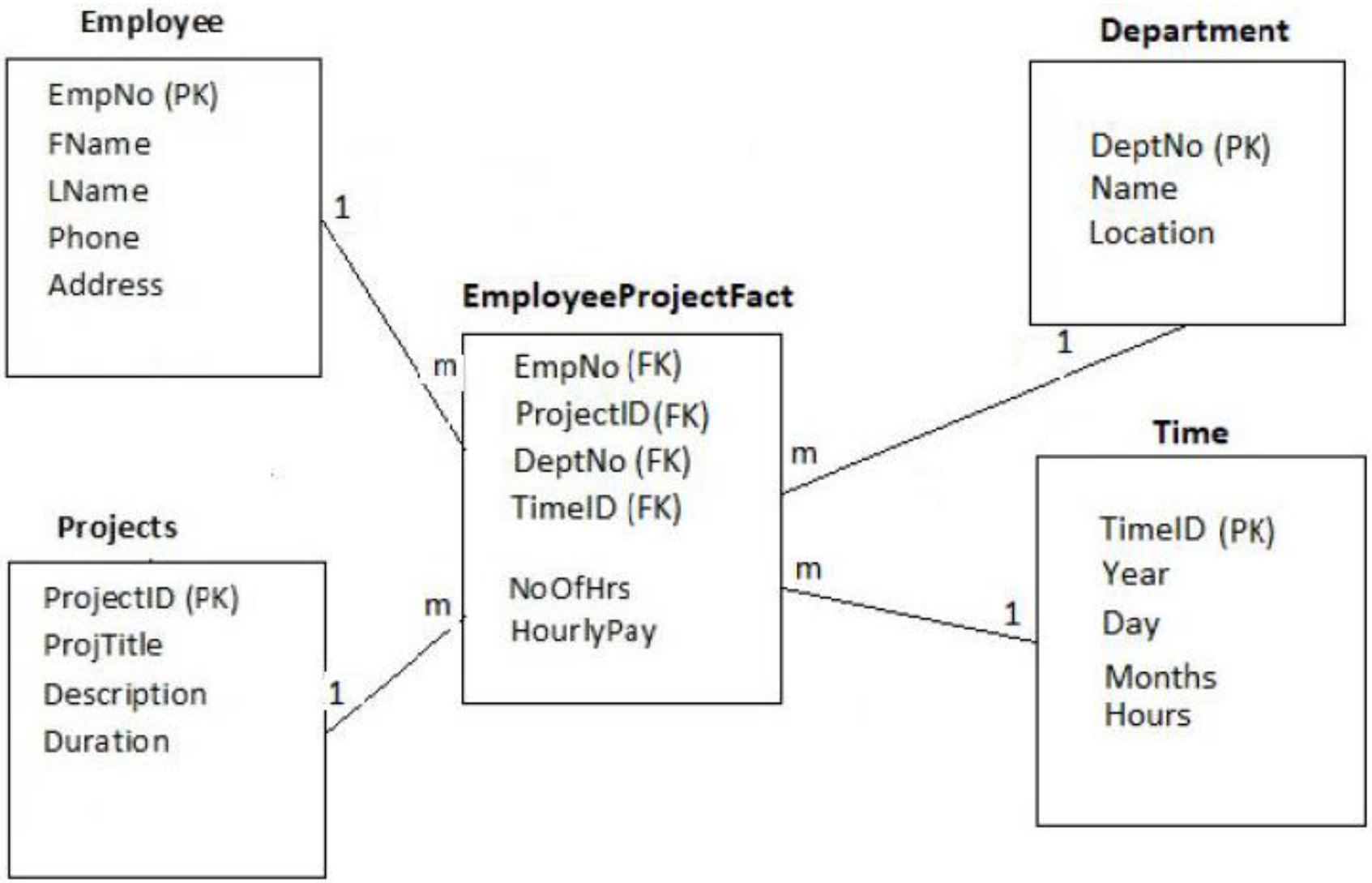
Denormalize remaining tables into flat dimension tables

- Take the remaining tables in the E/R model and denormalize them into dimension tables for the dimensional model.
- The primary key of each of the dimensions connects directly to the fact table.

Identify date and time dimension from E/R model

- The last step generally involves identifying the date and time dimension.
- Dates are generally stored in the form of a date timestamp column inside the E/R model.
- You will observe that date and time-related columns are generally found in the transaction-based tables.

Final Star Schema from ER Model M:M Relationship



Questions

- What are transitive dependencies. Explain with an example how to remove transitive dependency from a table.
- Discuss the most common performance improvement techniques used in star schemas. Give examples. **NB:** Don't use the the examples discussed in class.