UNIT 3 DIMENSIONAL MODELING

Structure

3.0 INTRODUCTION

In the earlier unit on Data Warehouse Architecture we have studied that many companies transform the data through an ETL (Extract, Transform and Load) process and store this data in a Data Warehouse for further analysis. In order to access the data from this Data Warehouse, companies use a process called Dimensional Modeling.

Dimensional Data Modeling is a data structure that helps optimize a Data Warehouse to retrieve data quickly. Ralph Kimball developed this technique that could read, analyze and summarize data in a Data Warehouse for further analysis.

Dimensional Modeling is a technique that uses Dimensions and Facts to store the data in a Data Warehouse efficiently. It optimizes the database for faster retrieval of the data. Dimensional Models have a specific structure and organize the data to generate reports that improve performance.

It stores the data in the most optimized way to ensure there is no redundancy of the data and to improve performance.

3.1 OBJECTIVES

After going through this unit, you shall be able to:

- understand the purpose of dimension modeling;.
- identifying the measures, facts, and dimensions;
- discuss the fact and dimension tables and their pros and cons;
- discuss the Star and Snowflake schema;
- explore comparative analysis of star and snowflake schema;
- describe Aggregate facts, fact constellation, and
- discuss various examples of star and snowflake schemas.

3.2 DIMENSIONAL MODELING

Dimensional modeling is a data model design adopted when building a data warehouse. Simply, it can be understood that dimension modeling reduces the response time of query fired unlike relational systems. The concept behind dimensional modeling is all about the conceptual design. Firstly let's see the introduction to dimensional modeling and how it is different from a traditional data model design. A data model is a representation of how data is stored in a database and it is usually a diagram of the few tables and the relationships that exist between them. This modeling is designed to read, summarize and compute some numeric data from a data warehouse.

A data warehouse is an example of a system that requires small number of large tables. This is due to many users using the application to read lot of data a characteristic of a data warehouse is to write the data once and read it many times over so it is the read operation that is dominant in a data warehouse. Now let's look at the data warehouse containing customer related information in a single table this makes it a lot easier for analytics just to count the number of customers by country but this time the use of tables in the data warehouse simplify the query processing.

The main objective of dimension modeling is to provide an easy architecture for the end user to write queries and also, to reduce the number of relationships between the tables and dimensions hence providing efficient query handling.

Dimensional modeling populates data in a cube as a logical representation with OLAP data management. The concept was developed by Ralph Kimball. It has *fact* and *dimension* as its two important measures. The transaction record is divided into either *facts*, which consists of business numerical transaction data, or *dimensions*, which are the reference information that gives context to the facts. The main objective of dimension modeling is to provide an easy architecture for the end user to write queries. Also it will reduce the number of relationships between the tables and dimensions, hence providing efficient query handling.

3.2.1 Key Features of Dimensional Modeling

Data Dimensional Modeling has gained popularity because of its unique way of analyzing data present in different Data Warehouses. The main features of dimensional modeling are as follows:

- **Easy to Understand**: It helps developers create and design databases and Schemas easily interpreted by business users. The relationship between Dimensions and Facts are pretty simple to read and understand.
- **Promote Data Quality**: Data modeling schemas enforce data quality before loading into Facts and Dimensions. Dimension and Fact are tied up by foreign keys that act as a constraint for referential integrity check to prevent fraudulent data from being loaded onto Schemas.
- **Optimise Performance**: Data modeling breaks the data into Dimensions and Facts and links them with foreign keys, thereby reducing the data redundancy. The data is stored in the optimized form and hence occupies less storage and can be retrieved faster.
- It reduces the number of relationships between different data elements.
- The aggregate functions used in the schemas optimize the query performance posted by the customers. Since data warehouse size keeps on increasing and with this increased size, the optimization becomes the concern which dimension modeling makes it easy.

3.2.2 Steps Involved in Dimensional Modeling

The following are the steps involved in Dimension modeling as shown in figure1.

- Identify Business objectives
- Identify Granularity
- Identify dimensions and attributes
- Build the Schema

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

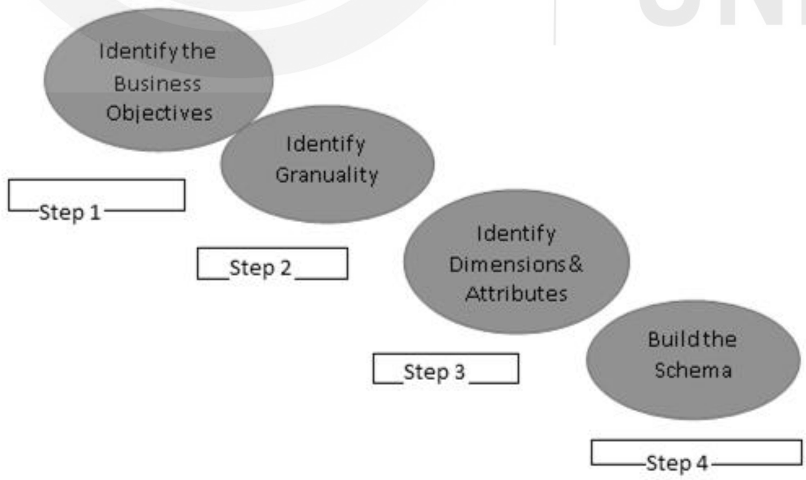

Figure 1: Steps involved in Dimension Modeling

Step 1: Identify the Business Objectives

Selection of the right business process to build a data warehouse and identifying the business objectives is the first step in dimension modeling. This is very important step otherwise this can lead to repeated process and software defects.

Step 2: Identifying Granularity
The grain literally means each minute detail of the business problem. This is decomposing of the large and complex problem into the lowest level information. For example, if there is some data month-wise. So, the table would contain details of all the months in a year. It depends on the report to be submitted to the management. This affects the size of the data warehouse.

Step 3: Identifying Dimensions and attributes
The dimensions of the data warehouse can be understood by the entities of the database. like, items, products, date, stocks, time etc. The identification of the primary keys and the foreign keys specifications all are described here.

Step 4: Build the Schema
The database structure or arrangement of columns in a database table, decides the schema. There are various popular schemas like, star, snowflake, fact constellation schemas - summarizing, from the selection of business process to identifying each and every finest level of detail of the business transactions. Identifying the significant dimensions and attributes would help to build the schema.

3.3 IDENTIFYING FACTS AND DIMENSIONS

We have just studied the steps involved in dimension modeling in the previous section. The last step is to build the schema which has greater significance in the overall modeling. So, let's see the elementary measures to build a schema.

Facts: A fact is an event. It is a measure which represents business items or transactions of items having association and context data.

Fact table: The Fact table contains the description of all the primary keys of all the tables used in the business processes which acts as a foreign key in the fact table. It also has an aggregate function to compute the business process on some entity. It is a numeric attribute of a fact, representing the performance or behavior of the business relative to the dimensions. The number of columns in the fact table is less than the dimension table. It is more normalized form.

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.

Dimension table: Dimension tables establish the context of the facts. The table stores fields that describe the facts. The data in the table are in de normalized form. So, it contains large number of columns as compared to fact table. The attributes in a dimension table are used as row and column headings in a document or query results display.

Example: In the example of student registration case study to any particular course can have attributes like student_id, course_id, program_id, date_of_registration, fee_id in fact table. Course summary can have course name, duration of the course etc. Student information can contain the personal details about the student like name, address, contact details etc.

Case Study or Scenario: Student Registration

Fact Table (student_id, course_id, program_id, date_of_registration, fee_id, Measure: Sum (Fee_amount))
Dimension Tables (Student_details,
Course_details
Program_details,
Fee_details,
Date)

In the context of Relational context, there are two basic models which are used in dimensional modeling:

- Star Model
- Snowflake Model

In the next section, let us study the details of these modeling techniques.

3.4 STAR SCHEMA

It represents the multidimensional model. In this model the data is organized into facts and dimensions. 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 star design around the primary table. This design is logically shown in the below figure 2.

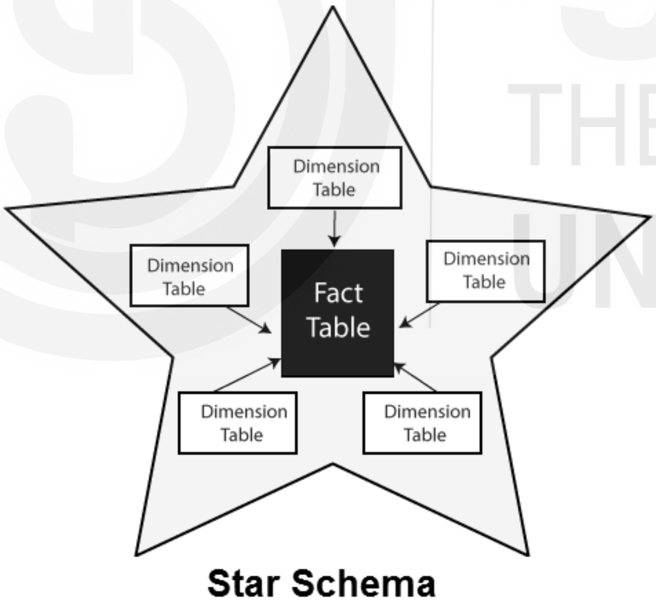

Figure 2: Star Schema

3.4.1 Features of Star Schema

- The data is in denormalized database.
- It provides quick query response
- Star schema is flexible can be changed or added easily.
- It reduces the complexity of metadata for developers and end users.

3.5 ADVANTAGES AND LIMITATIONS OF STAR SCHEMA

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.

3.5.1 Advantages of Star Schema

The main advantages of star schemas in a decision-support environment are:

- **Query performance**

Because a star schema database has a small number of tables and clear join paths, queries run faster than they do against an OLTP system. Small single-table queries, usually of dimension tables, are almost instantaneous. Large join queries that involve multiple tables take only seconds or minutes to run.

In a star schema database design, the dimensions are linked only through the central fact table. When two dimension tables are used in a query, only one join path, intersecting the fact table, exists between those two tables. This design feature enforces accurate and consistent query results.

- **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.

- **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.

- **Easily understood**

A star schema is easy to understand and navigate, with dimensions joined only through the fact table. These joins are more significant to the end user, because they represent the fundamental relationship between parts of the underlying business. Users can also browse dimension table attributes before constructing a query.

3.5.2 Limitations of Star Schema

As mentioned before, improving read queries and analysis in a star schema could involve certain challenges:

- Decreased data integrity: Because of the denormalized data structure, star schemas do not enforce data integrity very well. Although star schemas use countermeasures to prevent anomalies from developing, a simple insert or update command can still cause data incongruities.

- Less capable of handling diverse and complex queries: Databases designers build and optimize star schemas for specific analytical needs. As denormalized data sets, they work best with a relatively narrow set of simple queries. Comparatively, a normalized schema permits a far wider variety of more complex analytical queries.
- No Many-to-Many Relationships: Because they offer a simple dimension schema, star schemas don't work well for "many-to-many data relationships"

Example: Suppose a star schema is composed of a fact table as shown in Figure 3, SALES, and several dimension tables connected to it for time, branch, item, and geographic locations.

The TIME table has a column for each day, month, quarter, and year. The ITEM table has columns for each item_Key, item_name, brand, type, supplier_type. The BRANCH table has columns for each branch_key, branch_name, branch_type. The LOCATION table has columns of geographic data, including street, city, state, and country.

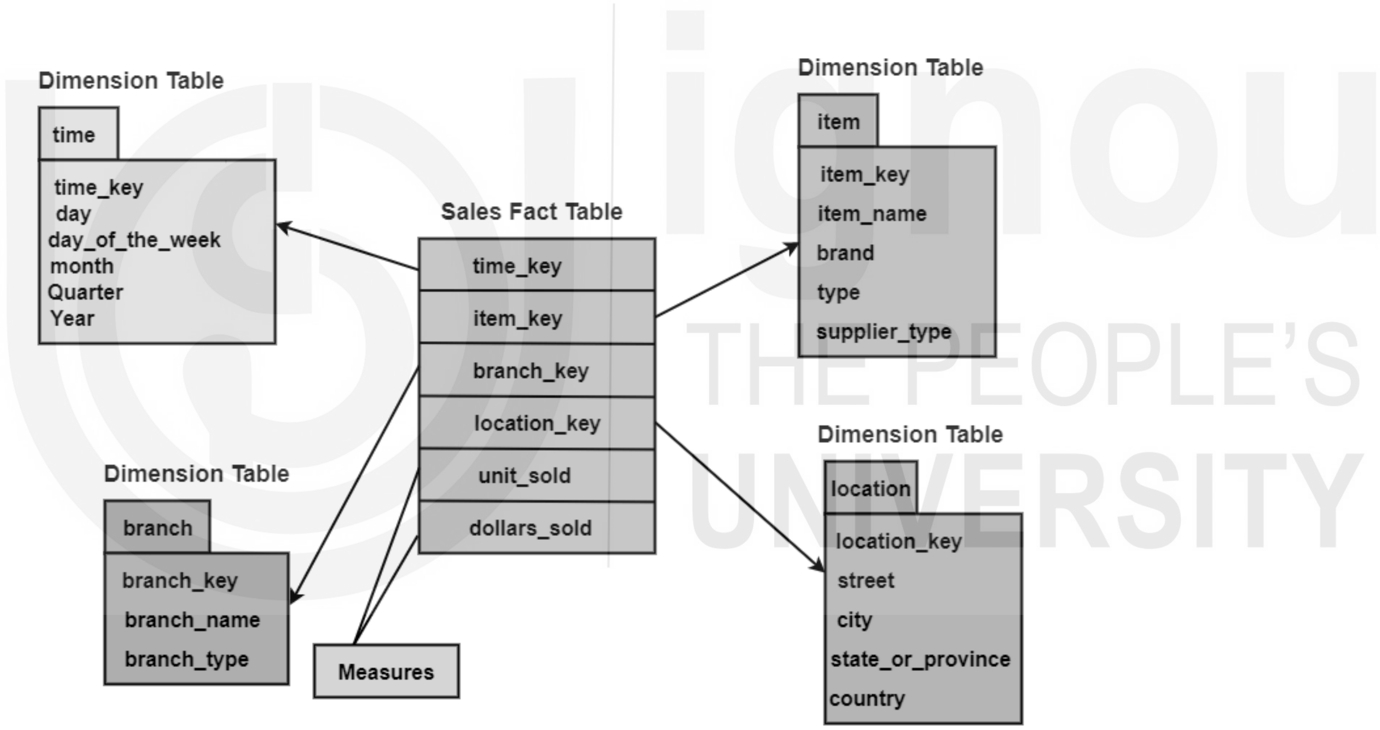

Figure 3: Example of Star Schema

☞ Check Your Progress 1:

1) Discuss the characteristics of star schema?

……………………………………………………………………………………
……………………………………………………………………………………
……………………………………………………………………………………

2) Mention the benefits and limitations of Dimensional Modeling.

……………………………………………………………………………………

……………………………………………………………………………………

……………………………………………………………………………………

3.6 SNOWFLAKE SCHEMA

The other modeling technique is Snowflake Schema. You can understand the term flakes as chocolate flakes on the pastry and ice-creams. These flakes add additional tastes to the chocolate. Similarly, snowflake schema is the extension of star schema which adds more dimensions to give more meaning to the logical view of the database. These additional tables are more normalized than star schema. The arrangement of data is like that the centralized fact table relates to multiple related dimensional tables. This can become more complex if the dimensions are more detailed and at multiple levels. In the conceptual hierarchy child table has multiple parent tables. You must keep in mind that we are just extending or flaking the dimension tables not the fact tables.

Snowflake Model: The snowflake model is the conclusion of decomposing one or more of the dimensions. 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.

3.6.1 Features of Snowflake Schema

- It has normalized tables
- Occupy less disk space.
- It requires more lookup time as many tables are interconnected and extending dimensions.

Example: In the below Figure 4 , the snowflake schema is shown of a case study of customers, sales, products, location wise quantity sold, and number of items sold are calculated. The customers, products, date, store are saved in the fact table with their respective primary keys acting in fact table as a foreign key. You will observe that the two aggregate functions can be applied to calculate quantity sold and amount sold. Further, the some dimensions are extended to the type of customer and also store information territory wise too. Note, date has been expanded into date, month, year. This schema will give you more opportunity to perform query handling in detail.

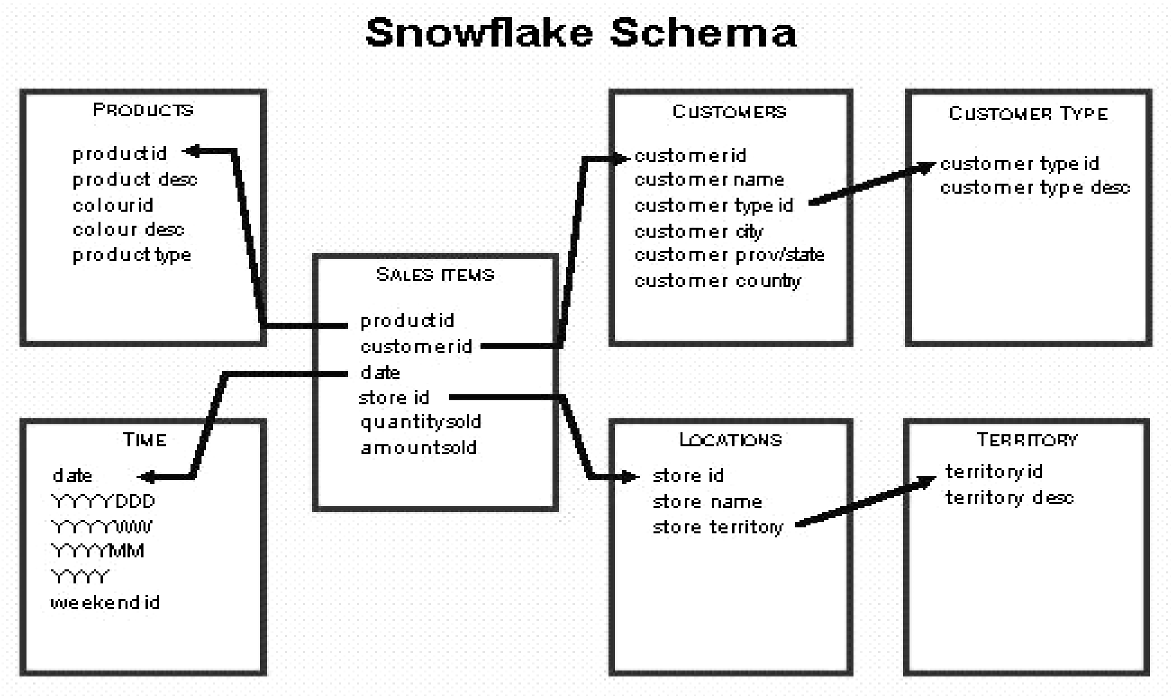

Figure 4: Snowflake Schema

3.7 ADVANTAGES AND LIMITATIONS OF SNOWFLAKE SCHEMA

Following are the advantages and limitations of Snowflake Schema:

3.7.1 Advantages of Snowflake Schema

- A Snowflake schema occupies a much smaller amount of disk space compared to the Star schema. Lesser disk space means more convenience and less hassle.
- Snowflake schema of small protection from various Data integrity issues. Most people tend to prefer the Snowflake schema because of how safe if it is.
- Data is easy to maintain and more structured.
- Data quality is better than star schema.

3.7.2 Limitations of Snowflake Schema

- Complex data schemas: As you might imagine, snowflake schemas create many levels of complexity while normalizing the attributes of a star schema. This complexity results in more complicated source query joins. In offering a more efficient way to store data, snowflake can result in performance declines while browsing these complex joins. Still, processing technology advancements have resulted in improved snowflake schema query performance in recent years, which is one of the reasons why snowflake schemas are rising in popularity.

- Slower at processing cube data: In a snowflake schema, the complex joins result in slower cube data processing. The star schema is generally better for cube data processing.
- Lower data integrity levels: While snowflake schemas offer greater normalization and fewer risks of data corruption after performing UPDATE and INSERT commands, they do not provide the level of transnational assurance that comes with a traditional, highly-normalized database structure. Therefore, when loading data into a snowflake schema, it's vital to be careful and double-check the quality of information post-loading.

3.7.3 Star Schema Vs Snowflake Schema

Let us study the differences between the Star and Snowflake schemas. The major differences are summarized in the following Table 1:

Table 1: Star Schema Vs Snowflake Schema

	Star Schema	**Snowflake Schema**
Dimension Table	The dimension tables in star schema are not normalized so they may contain redundancies	This schema has normalized dimension tables
Queries	The execution of queries is relatively faster as there are less joins needed in forming a query.	The execution of snowflake schema complex queries is slower than star schema as many joins and foreign key relations are needed to form a query. Thus performance is affected.
Performance	Star schema model has faster execution and response time	It has slow performance as compared to star schema
Storage Space	This type of schema requires more storage space as compared to snowflake due to unnormalised tables.	Snowflake schema tables are easy to maintain and save storage space due to normalized tables.
Usage	Star schema is preferred when the dimension tables have lesser rows	If the dimension table contains large number of rows, snowflake schema is preferred
Type of DW	This schema is suitable for 1:1 or 1: many relationships such as data marts.	It is used for complex relationships such as many: many in enterprise Data warehouses.
Dimension Tables	Star schema has a single table for each dimension	Snowflake schema may have more than one dimension table for each dimension.

3.8 AGGREGATE TABLES

Since, in the data warehouse the data is stored in multidimensional cube. In the information technology industry, there are various tools available to process the queries posted on the data warehouse engine. These tools are called business intelligence (BI) tools. These tools help to answer the complex queries and to take decisions. Aggregate word is very similar to the aggregation of the database schemas of relational tables that you must be familiar with. Aggregate fact tables roll up the basic fact tables of the schema to improve the query processing. The business tools smoothly select the level of aggregation to improve the query performance. Aggregate fact tables contain foreign keys referring to dimension tables.

Points to note about Aggregate tables:

- They are also called summary tables.
- It contains pre-computed queries of the data warehouse schema.
- It reduces the dimensionality of the base fact tables.
- It can be used to respond to the queries of the dimensions that are saved.

3.8.1 Need for Building Aggregate Fact Tables

Let us understand the need of building aggregate table. Aggregate tables also referred to pre-computed tables having partially summarized data. Simply putting in one word, it’s about speed or quick response to queries. This you can understand as an intermediate table which stores the results of the queries on I/O disk space. It uses aggregates functionality.

For example, there is a company ABC corporation limited which takes orders online and it there are millions of customer transactions placing orders. So, the dimension tables for the company could be Customer, Product and Order_date. In the fact table it maintains all the orders placed say, Fact_Orders. To generate a report of monthly orders by product type and by particular region, it needs aggregate (which are summary tables can be obtained) by Group by SQL query.

- It occupies less space than atomic fact tables. It nearly takes the half time of a general query processing.

- One of the more popular uses of aggregates is to adjust the granularity of a dimension. When the granularity of a dimension is changed, the fact table must be partially summarized to match the current grain of the new dimension, resulting in the creation of new dimensional and fact tables that fit this new grain standard.

- The Roll-up OLAP operation of the base fact tables generates aggregate tables. Hence the query performance increases as it reduces the number of rows to be accessed for the retrieval of data of a query.

3.8.2 Aggregate Fact Tables and Derived Dimension Tables

Aggregate facts are produced by calculating measures from more atomic fact tables. These tables contain computational SQL aggregate functions like AVERAGE, MIN, MAX, COUNT etc. It also contains function that helps to find output using group by. The aggregate fact tables produce summary statistics. Whenever, the speedy query handling is required the aggregate fact tables is the best option.

- Basically, aggregates allow you to store the intermediate results or pre-calculate the subqueries or queries fired on a data warehouse by summing data up to higher levels and storing them in a separate star.

- You can understand aggregate fact tables as the conformed copy of the fact table as it should provide you the same result of the query as the detailed fact table.

- This aggregate fact tables can be used in the case of large datasets or when there are large number of queries. It reduces the response time of the queries fired by users or customers. It is very useful in business intelligence application tools.

When you have complicated questions of multiple facts in multiple tables that are stored at different levels from one another, and when a reporting request includes yet another level, the levels at which facts are stored become even more relevant. You must be able to meet users' need for fact reporting at the business level. There's nothing wrong with improving the overall intelligence.
The levels at which facts are stored become especially important when you begin to have complex queries with multiple facts in multiple tables that are stored at levels different from one another, and when a reporting request involves still a different level.

You must be able to support fact reporting at the business levels which users require. There is nothing wrong with enhancing an aggregate with new facts or deriving new dimension. For measures, the only issue is if the new measures are atomic in the context of the aggregate fact. If, however, the new measures are received at a lower grain, you would be better off creating a new atomic fact for those measures prior to incorporating summarized measures into the aggregate. This would allow the new measures to be used for other purposes without having to go back to the source.

Let's say we have a fact table, ***FactBillReciept*** has monthly transactions. There can be different types of transaction receipts during a month for each supplier. This huge data would result in lot of calculations. So, we would build another aggregate table which is derived of base table.

FactBillMonthReceipt: It contains aggregated receipts per month, per supplier. But the problem is it has additional foreign keys like supplier_status for the month. To solve this, we have the concept of derived tables which contains additional measures and foreign keys that are not present in the base fact table.

Conformed Dimension

A conformed dimension is the dimension that is shared across multiple data mart or subject area. An organization may use the same dimension table across different projects without making any changes to the dimension tables.

Derived Tables

It is the significant addition to the Datawarehouse. Derived tables are used to create a second-level data marts for cross functional analysis.

Consolidated Fact tables: It is the fact table which has data from different fact tables used to form a schema with a common grain.

For example, to design a Sales department Datawarehouse schema assuming there are following entities and respective grains in them.

Sales: Employee, date, and product.
Budget: Department, Financial Year, Quarter-wise
Product can have various attributes like, product size, product _category and so on.

One thing to notice here is that the product attributes keep on changing as per the requirements, but product dimension remains the same. So, it is better to keep Product as a separate dimension.Let's design the tables and its grains.

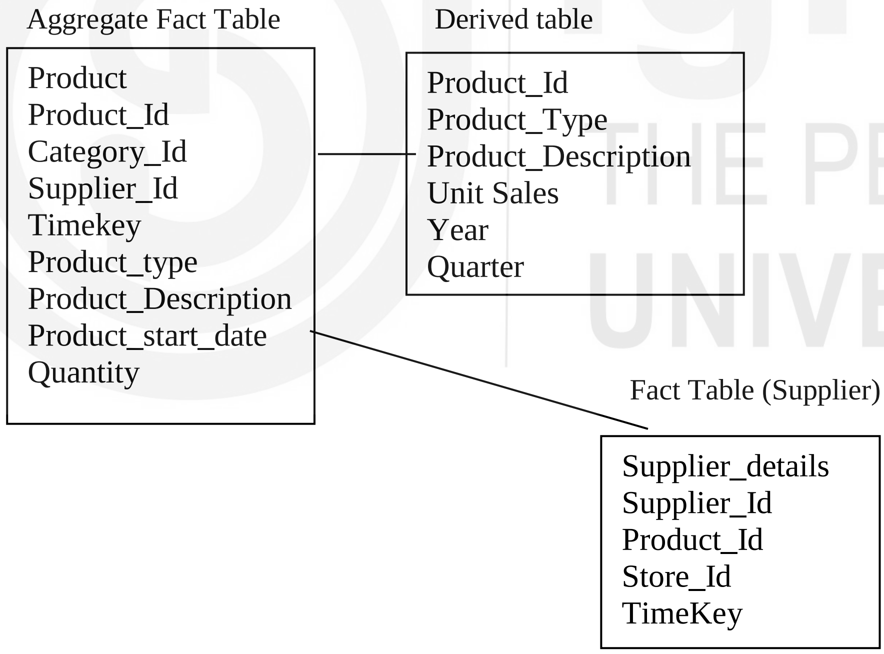

Figure 6: Aggregate Tables and Derived tables

The derived tables are very useful in terms of putting fewer loads on the Data warehouse engine for calculation.

☞ Check Your Progress 2

1) Write down an example of aggregate SQL query of a store Sales fact table(Date, Product, Store, Promotion, Transaction ID) Date aggregates(Month, Year), Product aggregates(Brand, Manufacturer),

Promotion aggregates(Ad, Discount, Coupon, In-Store Display) Store aggregates (District, State and Measurements Total_amt, Quantity?

…………………………………………………………………………………

…………………………………………………………………………………

…………………………………………………………………………………

2) What are fact and fact less tables in data warehouse?

…………………………………………………………………………………

…………………………………………………………………………………

…………………………………………………………………………………

3) Discuss the limitations of Aggregate Fact tables.

…………………………………………………………………………………

…………………………………………………………………………………

…………………………………………………………………………………

3.9 FACT CONSTELLATION SCHEMA

There is another schema for representing a multidimensional model. This term fact constellation is like the galaxy of universe containing several stars. It is a collection of fact schemas having one or more-dimension tables in common as shown in the figure below. This logical representation is mainly used in designing complex database systems.

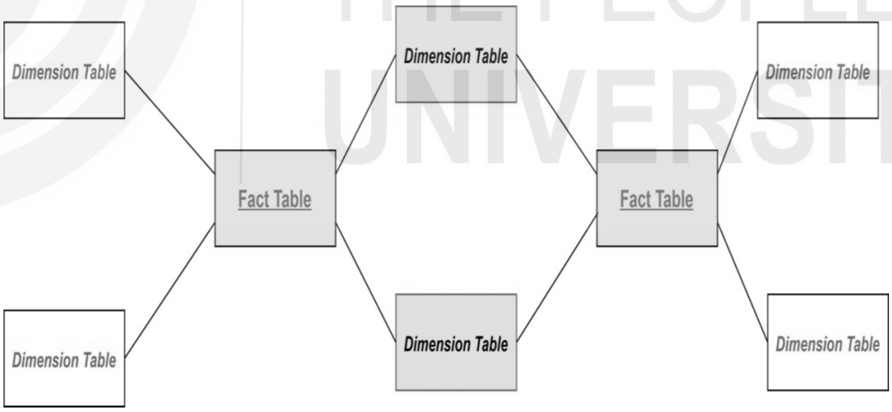

Figure 7: Fact Constellation Schema

In the above figure, it can be observed that there are two fact tables and two-dimension tables in the pink boxes are the common dimension tables connecting both the star schemas.

For example, if we are designing a fact constellation schema for University students. In the problem it is given that their fact table as

Fact tables

Placement (Stud_roll, Company_id, TPO_id) , need to calculate the number of students eligible and number of students placed.
Workshop (Stud_roll, Institute_id, TPO_id) need to find out the facts about number of students selected, number of students attended the workshop)

So, there are two fact tables namely, Placement and Workshop which are part of two different star schemas having dimension tables – Company, Student and TPO in Star schema with fact table Placement and dimension tables – Training Institute, Student and TPO in Star schema with fact table Workshop.

Both the star schema has two-dimension tables common and hence, forming a fact constellation or galaxy schema.

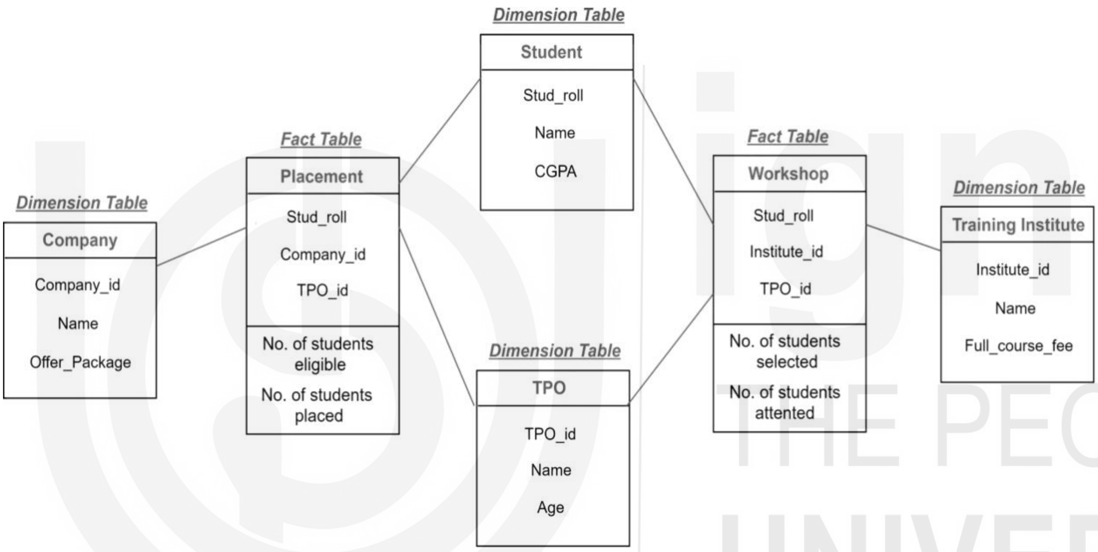

Figure 7: Fact Constellation of student and placement records

3.9.1 Advantages of Fact Constellation Schema

- Different fact tables are explicitly assigned to the dimensions.
- Provides a flexible schema for implementation

3.9.2 Limitations of Fact Constellation Schema

- Complexity of the schema involved because of several aggregations
- Fact constellation solution is hard to maintain and support

☞ Check Your Progress 3:

1. Suppose that a data warehouse consists of dimensions time, doctor, ward and patient, and the two measures count and charge, where charge is the fee that a doctor charges a patient for a visit. Draw (a) Star and (b) Snowflake schemas:

3.10 SUMMARY

This unit presented the basic designing of data warehouse. These topics are more focused on the various kind of modeling and schemas. It explored the grains, facts, and dimensions of the schemas. It is important to know about the dimensional modeling .as the appropriate modeling technique would yield the correct respond the queries.

A dimensional modeling is a kind of data structure used to optimize design of Data warehouse for the query retrieval operations. There are various schema designs. Here, it discussed star, snowflake, and fact constellations. From denormalized to normalized schemas uses dimension, fact, derived and aggregate fact table. Every table has some purpose and used for efficient designing in terms of space and query handling. This unit discusses the pros and cons of every tables. The number of examples used to explain the designing in different scenarios.

In the next unit, we will focus on Data Extraction, Transforming and Loading (ETL) process in Data Warehousing.

3.11 SOLUTIONS / ANSWERS

Check Your Progress 1:

1) Characteristics of Star Schema:

- Every dimension in a star schema is represented with only one-dimension table.
- The dimension table should contain the set of attributes.
- The dimension table is joined to the fact table using a foreign key
- The dimension table are not joined to each other
- Fact table would contain key and measure
- The Star schema is easy to understand and provides optimal disk usage.
- The dimension tables are not normalized. For instance, in the above figure, Country ID does not have Country lookup table as an OLTP design would have.
- The schema is widely supported by BI Tools

2) Following are some of the benefits of dimensional modeling:

- The Dimension Table stores the history information and a standard Dimension Table holds good quality data and allows easy access across the business.
- You can introduce new Dimensions without affecting other Dimensions and Facts in the Schema.
- Dimension and Fact Tables are easier to read and understand as compared to a normal table.
- Dimensional Models are built based on business terms, and hence it is quite understandable by the business.

- Dimensional Data Modeling in a Data Warehouse creates a Schema which is optimized for high performance. It means fewer joins between tables and it also helps with minimized data redundancy.
- The Dimensional Data Model also helps to boost query performance. It is more denormalized; therefore, it is optimized for querying.
- Dimensional Data Models can comfortably accommodate the change. Dimension Tables can have more columns added to them without affecting existing Business Intelligence applications using these tables.

Limitations of Dimensional Modeling

Although Dimensional Data Modeling is very crucial to any organization, it has a few limitations that companies need to take care of when incorporating the concept into their applications. Some of those limitations are given below:

- Designing and creating Schemas require domain knowledge about the data.
- To maintain the integrity of Facts and Dimensions, loading the Data Warehouses with a record from various operational systems is complicated.
- It is severe to modify the Data Warehouse operations if the organization adopts the Dimensional technique and changes the method in which they do business.
- Despite these limitations, the DDM technique has proved to be one of the simplest and efficient techniques to handle data in Data Warehouses till date

Check Your Progress 2:

1) Sales Fact Table

-Dimensions(Date, Product, Store, Promotion, TransactionID)
-Measurements Total_amt, Quantity
Create aggregate table with (Date, Store)

- SELECT Datakey, StoreKey, SUM(Total_amt), SUM(Qunatity)
  FROM Sales
  GROUP BY DateKey, StoreKey
- Store the result in Sales2 table

Queries that only reference Date and Store attributes can use the aggregate table instead.

- SELECT Store.District, SUM(Qunatity)
  FROM Sales, Store, Date
  WHERE Sales.DateKey = Date.Date_key
  AND Sales.Store_key = Store.Store_key
  AND Date.Month='April2021'
  GROUP BY Store.District
- Replaces "Sales" by "Sales2" → Same query result !

2) Fact less tables don't contain any facts. No measure or aggregate functions. It only contains foreign keys. It has data which has textual

information. Fact less tables does not refer to the data which requires calculations.
For example: Fact Table (Day_Id, Employee_ID,leave_Id, Time_Id
Dimension Table (Employee, leave, time, Day)

3) **Limitations of Aggregate fact tables:** Aggregate tables take lot of time to scan the rows of the base fact table. So, there will be more tables to manage. The size of aggregates in computing can be costly. Based on the greedy approach the size of aggregates is decided using hashing technique. If there are n dimensions in the table, then there can be 2^n possible aggregates. The load on the data warehouse becomes more complex.

Check Your Progress 3:

a) Star Schema

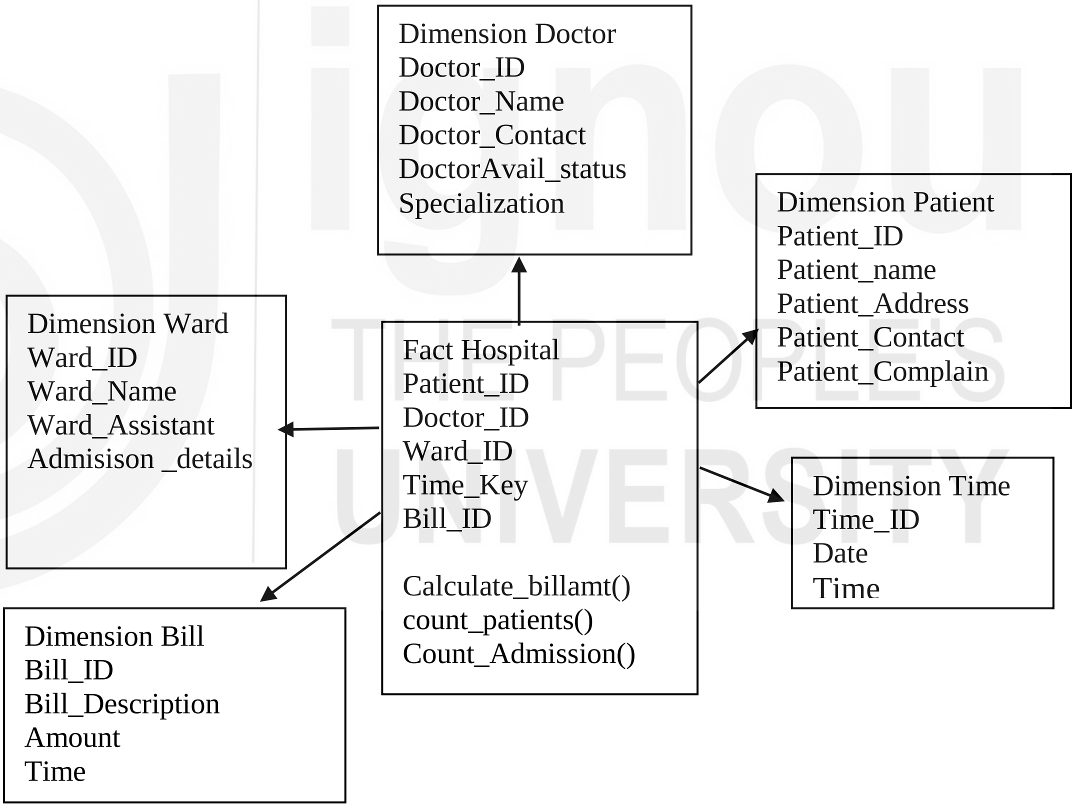

Figure 9 : Star Schema of Hospital Management System

a) Snowflake Schema

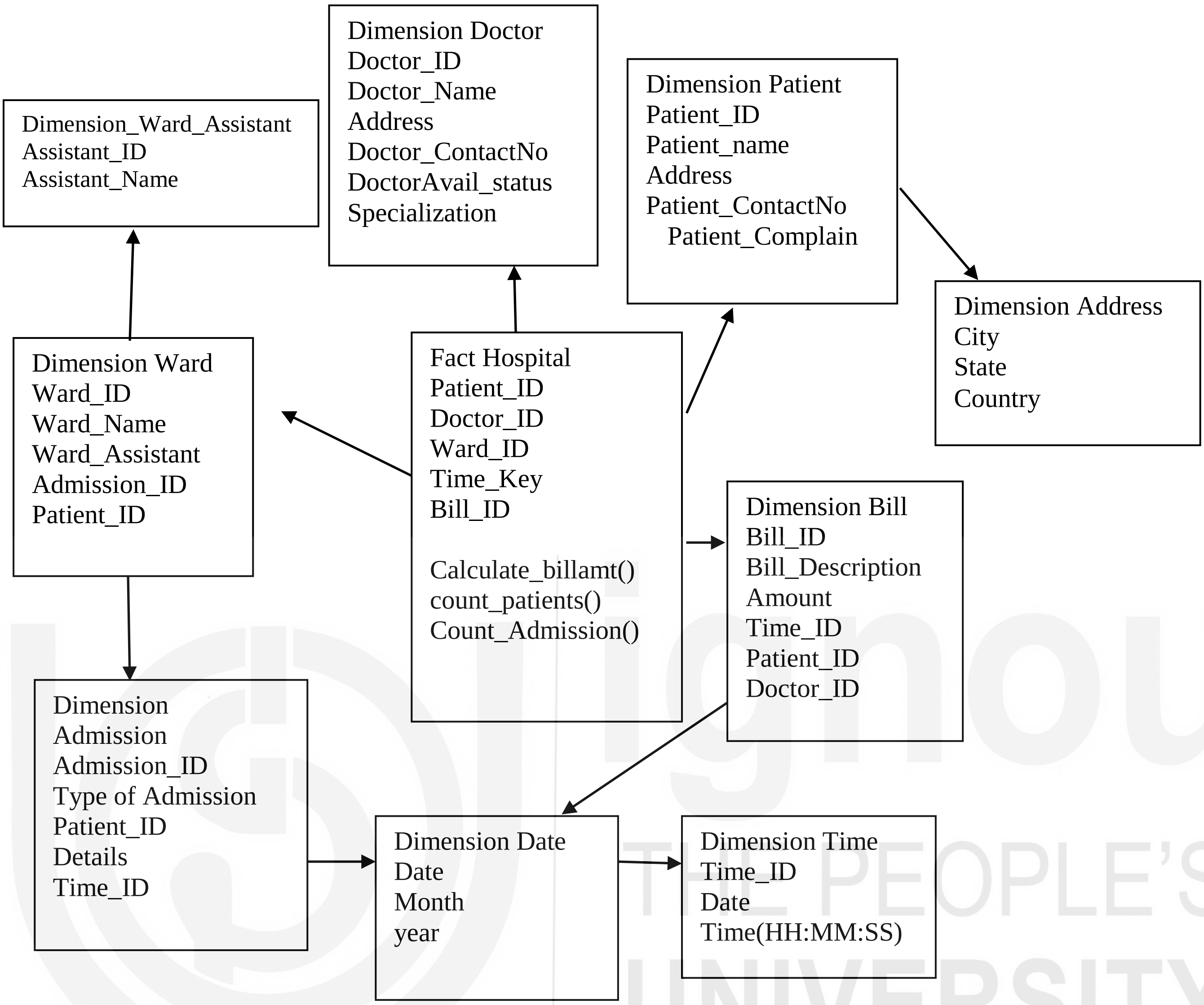

Figure 10: Snowflake Schema of Hospital Management System

3.12 FURTHER READINGS

- Jiawei Han and Micheline Kamber, ―Data Mining Concepts and Techniques, Third Edition, Elsevier, 2012.
- William H. Inmon, Building the Data Warehouse, Wiley, 4th Edition, 2005.
- Data Warehousing Fundamentals, Paulraj Ponnaiah, Wiley Student Edition
- Data Warehousing, Reema Thareja, Oxford University Press.
- Alex Berson and Stephen J.Smith, ―Data Warehousing, Data Mining & OLAP‖, Tata McGraw – Hill Edition, 35th Reprint 2016.
- K.P. Soman, Shyam Diwakar and V. Ajay, ―Insight into Data Mining Theory and Practice, Eastern Economy Edition, Prentice Hall of India, 2006.