Unit 2

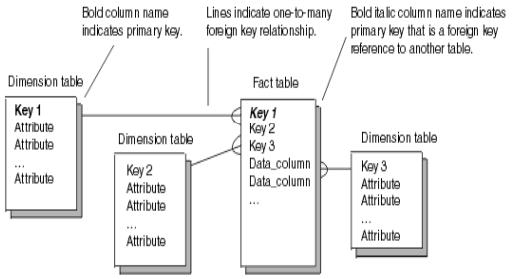
The Star Schema

- A star schema consists of fact tables and dimension tables
- Fact tables contain the quantitative or factual data about a business--the information being queried. This information is often numerical, additive measurements and can consist of many columns and millions or billions of rows. A table can be classified as a fact less fact table which only contains the concatenated keys, and does not contain any fact or measure
- Dimension tables are usually smaller and hold descriptive data that reflects the dimensions, or attributes, of a business
- SQL queries then use joins between fact and dimension tables and constraints on the data to return selected information

Star Schema Keys

- Any table that references or is referenced by another table must have a primary key, which is a column or group of columns whose contents uniquely identify each row
- In a simple star schema, the primary key for the fact table consists of one or more foreign keys
- A foreign key is a column or group of columns in one table whose values are defined by the primary key in another table

Examples of Star Schema Keys



- In the Previous Slide figure illustrates the relationship of the fact and dimension tables within a simple star schema with a single fact table and three dimension tables
- The fact table has a primary key composed of three foreign keys, Key1, Key2, and Key3, each of which is the primary key in a dimension table
- Non key columns in a fact table are referred to as data columns. In a dimension table, they are referred to as attributes

In the slide the figure is used to illustrate schemas:

- The items listed within the box under each table name indicate columns in the table.
- Primary key columns are labeled in bold type
- Foreign key columns are labeled in italic type
- Columns that are part of the primary key and are also foreign keys are labeled in bold italic type
- Foreign key relationships are indicated by lines connecting tables
- Although the primary key value must be unique in each row of a dimension table, that value can occur multiple times in the foreign key in the fact table--a many-to-one relationship

Advantages of Star Schema

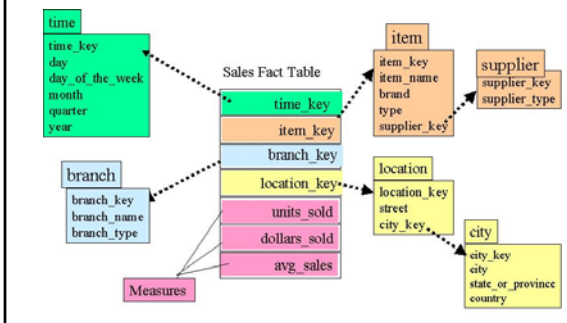
The main advantages of star schemas is that they:

- Provide a direct and intuitive mapping between the business entities being analyzed by end users and the schema design
- Provide highly optimized performance for typical star queries
- They are widely supported by a large number of business intelligence tools, which may anticipate OR even require that the data-warehouse schema contains dimension tables
- The schema is easier to understand and tends to involve less joins than a snowflake or E-R schema
- Star schemas are much easier to use and (more importantly) make perform well with ad-hoc query tools such as Business Objects or Report Builder

- Partitioning a star schema is relatively straightforward as only the fact table needs to be partitioned.
- Partition elimination means that the query optimizer can ignore partitions that could not possibly participate in the query results, which saves on I/O
- Slowly changing dimensions are much easier to implement on a star schema than a snowflake

The Snowflake Schema

- The snowflake schema is a more complex data warehouse model than a star schema, and is a type of star schema.
- Snowflake schemas normalize dimensions to eliminate redundancy. That is, the dimension data has been grouped into multiple tables instead of one large table
- For example, a location dimension table in a star schema might be normalized into a location table and city table in a snowflake schema (*refer to the next slide*)
- 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



Factless Fact Table

- Fact table is a collection of many facts and measures having multiple keys joined with one or more dimension tables.
- Facts contain both numeric and additive fields. But factless fact table are different from all these.
- A ***factless fact table is fact table that does not contain fact.*** They contain only dimesional keys and it captures events that happen only at information level but not included in the calculations level, just an information about an event that happen over a period

- Common examples of factless fact tables include:
  - Identifying product promotion events (to determine promoted products that didn't sell)
  - Tracking student attendance or registration events
  - Tracking insurance-related accident events
  - Identifying building, facility, and equipment schedules for a hospital or university

Aggregate Fact Tables

- Aggregate tables, also known as summary tables, are fact tables which contain data that has been summarized up to a different level of detail.
- For example, let's say that our data warehouse contains a transaction table with the following characteristics

Table dimensionality:

account id, transaction type, day id, transaction amount

Average number of transactions per day: 30 million
Number of days stored in the transaction table: 30
Approximate number of rows: 900 million rows

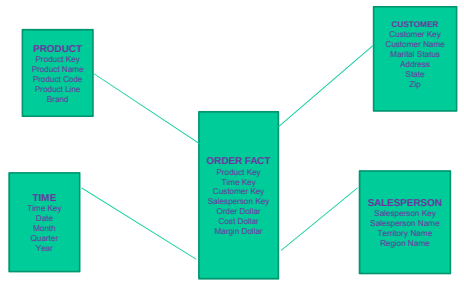
Let's assume that half of the daily transactions are deposits, so there are approximately 450 million rows that represent deposit transactions. The other half are withdrawals.

Suppose a DW user wants to know how much money was deposited into the bank during the past month

- *To answer this question, we will build an aggregate table which summarizes the transaction table by transaction type. The aggregate may be defined as follows:*
- *create table fact_transaction_aggregate as*
  select day_id, transaction_type,
  sum(transaction_amount) as transaction_amount
  from transaction_fact
  group by day_id, transaction_type

Updates to Dimension Tables

- Changes to dimension tables may be classified into 3 categories:
  - Type 1:Correction of errors
  - Type 2:Preservation of history
  - Type 3:Tentative soft Revisions



Principle for Type 1 change

- Correction in source system
- Old value in source system discarded
- Overwrite the attribute value in dimension table row with new value
- No change in dimension rows
- Easiest

Principle for Type 2 change

- True change in source system
- Old value in source system preserved
  - Add new dimension row in the table
    - New row inserted with new surrogate key
  - With a new value of the changed attribute
  - Key of the original row is not affected
- No change in actual value
- Change partitions the history in data warehouse
- Every change for the same attribute must be preserved

Principle for Type 3 change

- Relate to soft change or tentative change attribute in source system
- Need to keep track of history of old and new values of the change attributes.
- Use to compare performance across transition
- Provide ability to track forward and backward
- No new dimension row is needed.
- Existing query will seamlessly switch to the current value
- Any query that need to use old value must be revised accordingly
- Technique best one for sot change at a time

Miscellaneous Dimensions

- Large Dimensions
- Rapidly Changing Dimensions
- Multiple Hierarchies
- Junk Dimensions

Large Dimensions

- A large dimension is very deep; has a large number of rows.
- It may be very wide & may have a large number of attributes.
- Large dimensions call for special considerations.

Multiple Hierarchies

- Large dimensions often have multiple hierarchies.
- Example:Product dimension of a large retailer
- One set of attributes may from the hierarchy for the marketing department.
- Users from that department use these attributes to drill up & drill down.
- In the same way the finance department may need to use their own set of attributes from the same product dimension to drill up & drill down.

Rapidly Changing Dimensions

- With Type 2 change,
- Additional dimension table row with the new value of the changed attribute can be created .
- Helps to preserve the history.
- If same attribute changes a second time, create one or more dimension table row with the latest value.

Junk Dimensions

- Some of the flags & textual data may be too obscure to be real value.
- These may not be included as significant fields in the major dimensions.
- At the same time these flags & texts cannot be discarded either.
- Some of the choices are:
  - Exclude & discard all flags & texts.
  - Place the flags & texts unchanged in the fact table.
  - Make each flag & text a separate dimension table on its own.
  - Keep only those flags & texts that are meaningful.

Families of STARS

- Almost all data warehouses contain multiple **STAR** schema structures. Each **STAR** serves a specific purpose to track the measures stored in the fact table.
- A set of related **STAR** schemas make up a family of **STARS**. Families of **STARS** are formed for various reasons. A family can be just formed adding aggregate fact tables and the derived dimension tables to support the aggregates. Sometimes, a core fact table can be created containing facts interesting to most users and customized fact tables for specific user groups.
- The fact tables of the STARS in a family share dimension tables. Usually, the time dimension is shared by most of the fact tables in the group. Also, dimension tables from multiple STARS may share the fact table of one STAR.
- Examples are snapshot and transaction tables, core and custom tables, and tables supporting a value chain or a value circle. A family of STARS relies on conformed dimension tables and standardized fact tables.

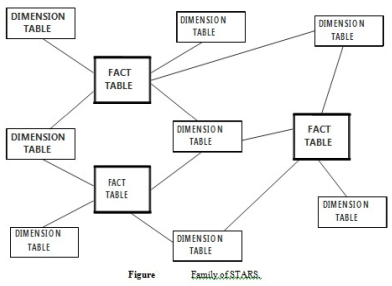


Figure Family of STARS.

© Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi-63, by Imran Khan Asst. Professor . U2.

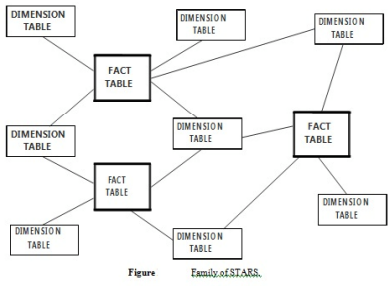


Figure Family of STARS.

© Bharati Vidyapeeth's Institute of Computer Applications and Management, New Delhi-63, by Imran Khan Asst. Professor . U2.

Steps for Design and Construction of Data Warehouse

- The phases of a data warehouse project are given below.
  - Identify and gather requirements
  - Design the dimensional model
  - Develop the architecture, including the Operational Data Store (ODS)
  - Design the relational database and OLAP cubes
  - Develop the data maintenance applications
  - Develop analysis applications
  - Test and deploy the system

Identify and Gather Requirements

- Identify stakeholders i.e. the persons who are directly or indirectly connected with the data warehouse project. Shareholders must understand and support the business value of the various process and the project.
- Understand the business requirements and the business process with all the stakeholders including the technical experts. Focus should be on understanding the business processes and not on the data that is involved.

Design the Dimensional Model

- The requirements gathered during the requirements phase and the data realities drive the design of the dimensional model. It must address business needs, grain of detail, and what dimensions and facts to be included
- The dimensional model must suit the requirements of the users and support ease of use for direct access
- The model must also be designed so that it is easy to maintain and can adapt to future changes
- The model design must result in a relational database that supports OLAP cubes to provide "instantaneous" query results for analysts.

Other Considerations

- Dimensional Model Schemas i.e. whether to go in for Star Schema or go in for Snow Flake Schema
- Dimensional Table which must take into account hierarchies, surrogate keys etc.
- Date and Time dimensions
- Granularity issues
- Slowly Changing dimensions ; rapidly changing dimensions etc.

Design the Architecture

- The data warehouse architecture reflects the dimensional model that is developed to meet the business requirements
- Dimension design largely determines dimension table design, and fact definitions determine fact table design
- It is to be taken into account whether to create a star or snowflake schema. This depends more on implementation and maintenance considerations rather than on business needs.
- Information can be presented to the user in the same way regardless of whether a dimension is snow flaked or not.

- The data warehouse architecture reflects the dimensional model that is developed to meet the business requirements
- Dimension design largely determines dimension table design, and fact definitions determine fact table design

Design for updates and expansion

- Data warehouse architectures must be designed to accommodate ongoing data updates, and to allow for future expansion with minimum impact on existing design

Design the Relational Database and OLAP Cubes

- In this phase, the star or snowflake schema is created in the relational database, surrogate keys are defined and primary and foreign key relationships are established. Views, indexes, and fact table partitions are also defined. OLAP cubes are designed that support the needs of the users
- Considerations are made in this phase with respect to the keys and their relationships in the dimensional tables. Keys such as primary keys, surrogate keys, views, indexes etc. are taken into account

Develop the Operational Data Store

- Business problems are best addressed by creating a database designed to support tactical decision-making
- The Operational Data Store (ODS) is an operational construct that has elements of both data warehouse and a transaction system

Develop the Data Maintenance Applications

- The data maintenance applications, including extraction, transformation, and loading processes, must be automated, often by specialized custom applications.
- Data Transformation Services (DTS) in SQL Server 2000 is a powerful tool for defining many transformations

Develop Analysis Applications

- The applications that support data analysis by the data warehouse users are constructed in this phase of data warehouse development
- OLAP cubes and data mining models are constructed using Analysis Services tools, and client access to analysis data is supported by the Analysis Server
- Other analysis applications, such as Microsoft PivotTables, predefined reports, Web sites, and digital dashboards, are also developed in this phase, as are natural language applications using English Query. Specialized third-party analysis tools are also acquired and implemented or installed

Test and Deploy the System

- It is important to involve users in the testing phase
- After initial testing by development and test groups, users should load the system with queries and use it the way they intend to after the system is brought on line
- Substantial user involvement in testing will provide a significant number of benefits

Among the benefits are:

- Discrepancies can be found and corrected
- Users become familiar with the system
- Index tuning can be performed

Types of OLAP Servers

What is an OLAP Server ?

- An OLAP server is a high-capacity, multi-user data manipulation engine specifically designed to support and operate on multi-dimensional data structures
- A multi-dimensional structure is arranged so that every data item is located and accessed based on the intersection of the dimension members which define that item
- The design of the server and the structure of the data are optimized for rapid ad-hoc information retrieval in any orientation, as well as for fast, flexible calculation and transformation of raw data based on formulaic relationships
- The OLAP Server may either physically stage the processed multi-dimensional information to deliver consistent and rapid response times to end users, or it may populate its data structures in real-time from relational or other databases, or offer a choice of both

Type of OLAP Servers

There are four types:

- MOLAP stands for Multidimensional Online Analytical Processing
- ROLAP stands for Relational Online Analytical Processing
- HOLAP stands for Hybrid Online Analytical Processing
- Secialized SQL Servers

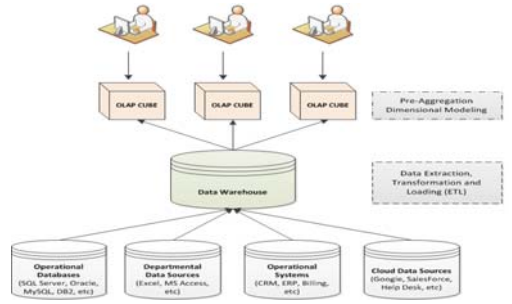
ROLAP Server

- ROLAP servers contain both numeric and textual data, serving a much wider purpose than other OLAP counterparts. ROLAP DBMSs are supported by relational technology. RDBMSs support numeric, textual, spatial, audio, graphic, and video data, general-purpose DSS analysis, freely structured data, numerous indexes, and star schema's. ROLAP servers can have both disciplined and ad hoc usage and can contain both detailed and summarized data
- ROLAP supports large databases while enabling good performance, platform portability, exploitation of hardware advances such as parallel processing, robust security, multi-user concurrent access (including read-write with locking), recognized standards, and openness to multiple vendor's tools. ROLAP is based on familiar, proven, and already selected technologies

MOLAP Server

- MOLAP (multidimensional online analytical processing) is online analytical processing (OLAP) that indexes directly into a multidimensional database.
- MOLAP processes data that is already stored in a multi dimensonal array in which all possible combinations of data are reflected, each in a cell that can be accessed directly. For this reason, MOLAP is, for most uses, faster and more user-responsive than relational online analytical processing (ROLAP), the main alternative to MOLAP.
- MOLAP is often used as part of a data warehouse application.

Architecture of MOLAP Server



Discussion…..
Queries??

Thank You !