CS –32
Data Warehousing with SQL Server 2012

02. Designing and Implementation of Data Warehousing

Ch.02 Syllabus

- Logical Design for data warehouse
- Physical Design for data warehouse
- Design dimension table, fact table for data warehouse
- Design and implement effective physical data structure for data warehouse

Logical Design for Data Warehouse

- A logical design is conceptual and abstract. You do not deal with the physical implementation details yet. We deal only with defining the types of information that we need.
- One technique you can use to model your organization's logical information requirements is entity-relationship modeling.

Logical Design for Data Warehouse

- Entity-relationship modeling involves identifying the things of importance (entities), the properties of these things (attributes), and how they are related to one another (relationships).
- The process of logical design involves arranging data into a series of logical relationships called entities and attributes, an entity often maps to a table. An attribute is a component of an entity that helps define the uniqueness of the entity. In relational databases, an attribute maps to a column.

Data Warehouse

- To ensure that your data is consistent, you must use unique identifiers. A unique identifier is something you add to tables so that you can differentiate between the same item when it appears in different places.
- In a physical design, this is usually a primary key.

Data Warehouse

- While entity-relationship diagramming has traditionally been associated with highly normalized models such as OLTP applications, the technique is still useful for data warehouse design in the form of dimensional modeling.

Data Warehouse

- In dimensional modeling, instead of seeking to discover atomic units of information (such as entities and attributes) and all of the relationships between them, you identify which information belongs to a central fact table and which information belongs to its associated dimension tables.

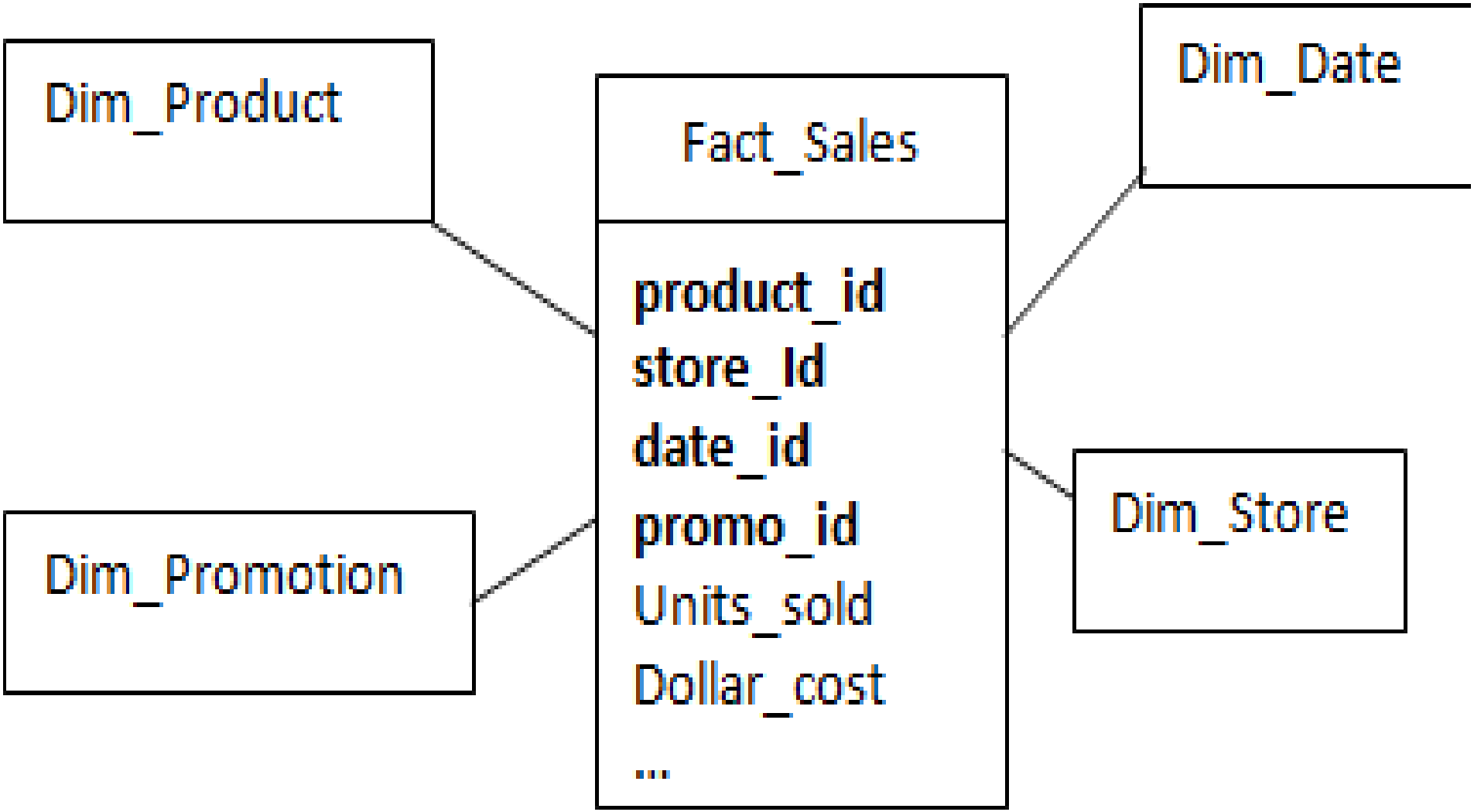
Data Warehouse

- You identify business subjects or fields of data, define relationships between business subjects, and name the attributes for each subject.

Star Schemas :

- The star schema is the simplest data warehouse schema. It is called a star schema because the diagram resembles a star, with points radiating (વિસર્જન) from a center.
- The center of the star consists of one or more fact tables and the points of the star are the dimension tables, as shown in the figure.

Star Schemas :



Data Warehousing : Fact Tables

- A fact table typically has two types of columns : those that contain numeric facts (often called measurements), and those that are foreign keys to dimension tables.
- A fact table contains either details-level facts or facts that have been aggregated (એકત્રિત).

Data Warehousing : Fact Tables

- Fact tables that contain aggregated facts are often called summary tables.
- A fact table usually contains facts with the same level of aggregation. Though most facts are additive facts can be aggregated by simple arithmetical addition.
- A common example of this is sales. Non-additive facts cannot be added at all.

Data Warehousing : Fact Tables

- An example of this is averages. Semi-additive facts can be aggregated along some of the dimensions and not along others. An example of this is inventory levels, where you cannot tell what a level means simply by looking at it.

Requirements of Fact Tables :

- You must define a fact table for each star schema. From a modeling standpoint, the primary key of the fact table is usually a composite key that is made up of all of its foreign keys.

Data Warehousing : Dimension Tables

- A dimension is a structure, often composed of one or more hierarchies, tat categorizes data.
- Dimensional attributes help to describe the dimensional value. They are normally descriptive, textual values.
- Several distinct dimensions, combined with facts, enable you to answer business questions.

Data Warehousing : Dimension Tables

- Commonly used dimensions are customers, products and time.
- Dimension data is typically collected at the lowest level of detail and then aggregated into higher level totals that are more useful for analysis.
- These natural rollups or aggregations within a dimension table are called hierarchies.

Hierarchy :

- The specification of levels that represents relationship between different attributes within a dimension.
- For Example,
  - One possible hierarchy in the Time dimension is
    - Year → Quarter → Month → Day

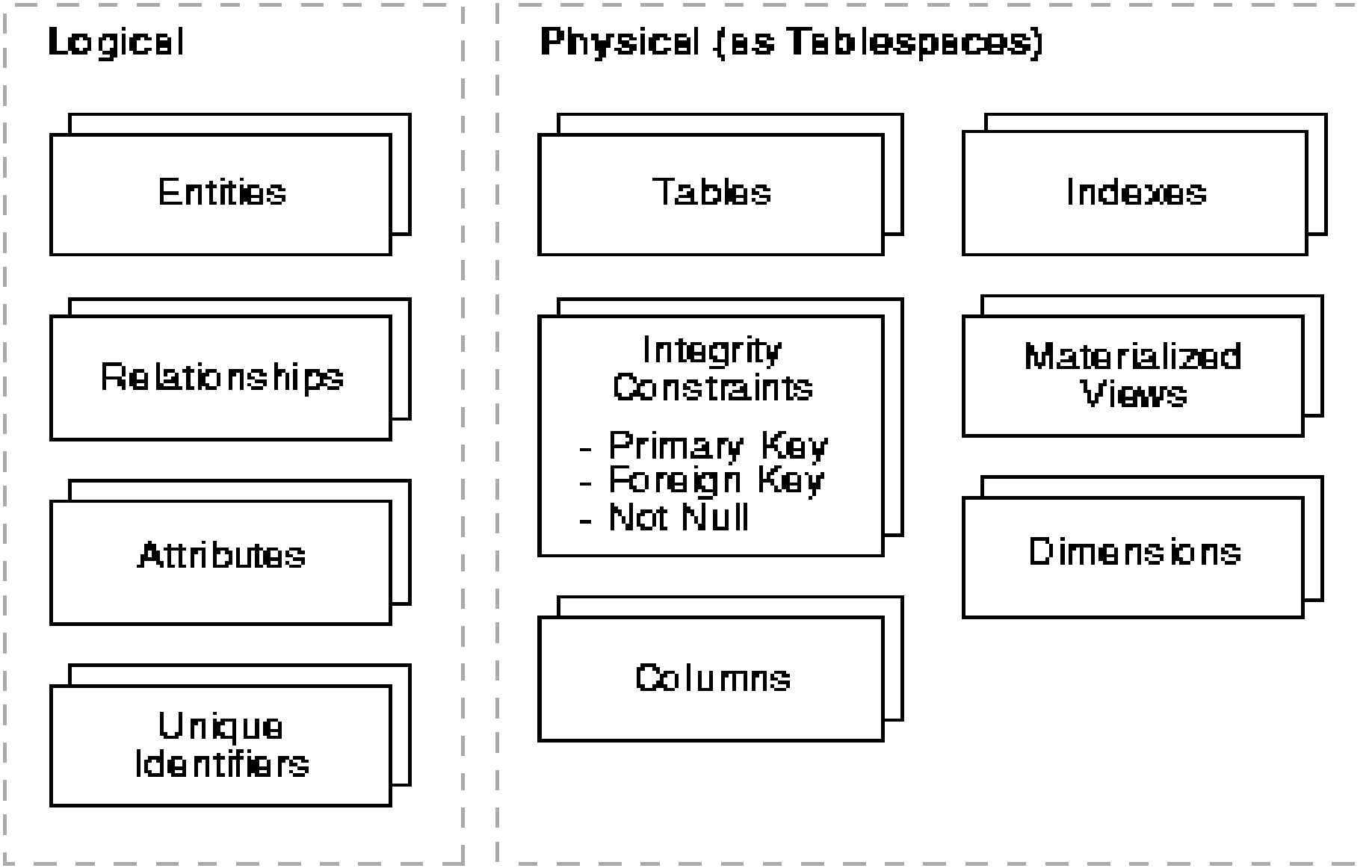
Physical Design :

- Logical design is what you draw with a pen and a paper and paper or design with Oracle Warehouse Builder or Oracle Designer before building your data warehouse. Physical design is the creation of the database with SQL Statements.
- During the logical design phase, you defined a model for your data warehouse consisting of entities, attributes, and relationships.

Physical Design :

- The entities are linked together using relationships. Attributes are used to describe the entities. The unique identifier (UID) distinguishes (અલગ પડે છે) between one instance of an entity and another.
- Here given figure will illustrates a graphical way of distinguishing between logical and physical designs.

Physical Design Diagram :



Physical Design :

- During the physical design process, you translate the expected schemas into actual database structures. At this time, you must map:
  - Entities to tables
  - Relationships to foreign key constraints
  - Attributes to columns
  - Primary unique identifiers to primary key constraints.
  - Unique identifiers to unique key constraints.

Physical Design Structures :

- Once you have converted your logical design to a physical one, you must create some or all of the following structures:
  - Tablespaces :
    - A **tablespace** consists of one or more **datafiles**, which are physical structures within the operating system you are using. A datafile is associated with only one **tablespace**.

Physical Design Structures :

- Table and Partitioned Tables:
  - **Partitioning** is done to enhance performance and facilitate easy management of **data**. **Partitioning** also helps in balancing the various requirements of the system. It optimizes the hardware performance and simplifies the management of **data warehouse** by**partitioning** each fact **table** into multiple separate**partitions**.

Physical Design Structures :

- Views:
  - A view is a tailored presentation of the data contained in one or more tables or other views.
  - A view takes the output of a query and treats it as a table.
  - Views do not require any space in the database.

Physical Design Structures :

- Integrity (અખંડિતતા) Constraints (મર્યાદાઓ) :
  - Integrity constraints are used to enforce business rules associated with your database and to prevent having invalid information in the tables.
  - Integrity constraints in data warehousing differ from constraints in OLTP environments.

Physical Design Structures :

- In OLTP environments, they primarily prevent the insertion of invalid data into a record, which is not a big problem in data warehousing environments because accuracy has already been guaranteed.
- In data warehousing environments, constraints are only used for query rewrite.

Physical Design Structures :

- NOT NULL constraints are particularly common in data warehouses.
- Under some specific circumstances (સંજોગો), constraints need space in the database. These constraints are in the form of the underlying unique index.

Physical Design Structures :

- Dimensions :
  - Some of these structures require disk space. Other exist only in the data dictionary. Additionally, the following structures may be created for performance improvement.

Physical Design Structures :

- Indexes and Partitioned Indexes :
  - Indexes are optional structures associated with tables or clusters. In addition to the classical B-tree indexes, bitmap indexes are very common in data warehousing environments.
  - Additionally, they are necessary for some optimized (અશારૂપ) data access methods such as star transformations.

Physical Design Structures :

- Indexes are just like tables in that you can partition them, although the partitioning strategy is not dependent upon the table structure.
- Partitioning indexes makes it easier to manage the data warehouse during refresh and improves query performance.

Physical Design Structures :

- Materialized Views :
  - Materialized views are query results that have been stored in advance so long-running calculations are not necessary when you actually execute your SQL statements.

Physical Design Structures :

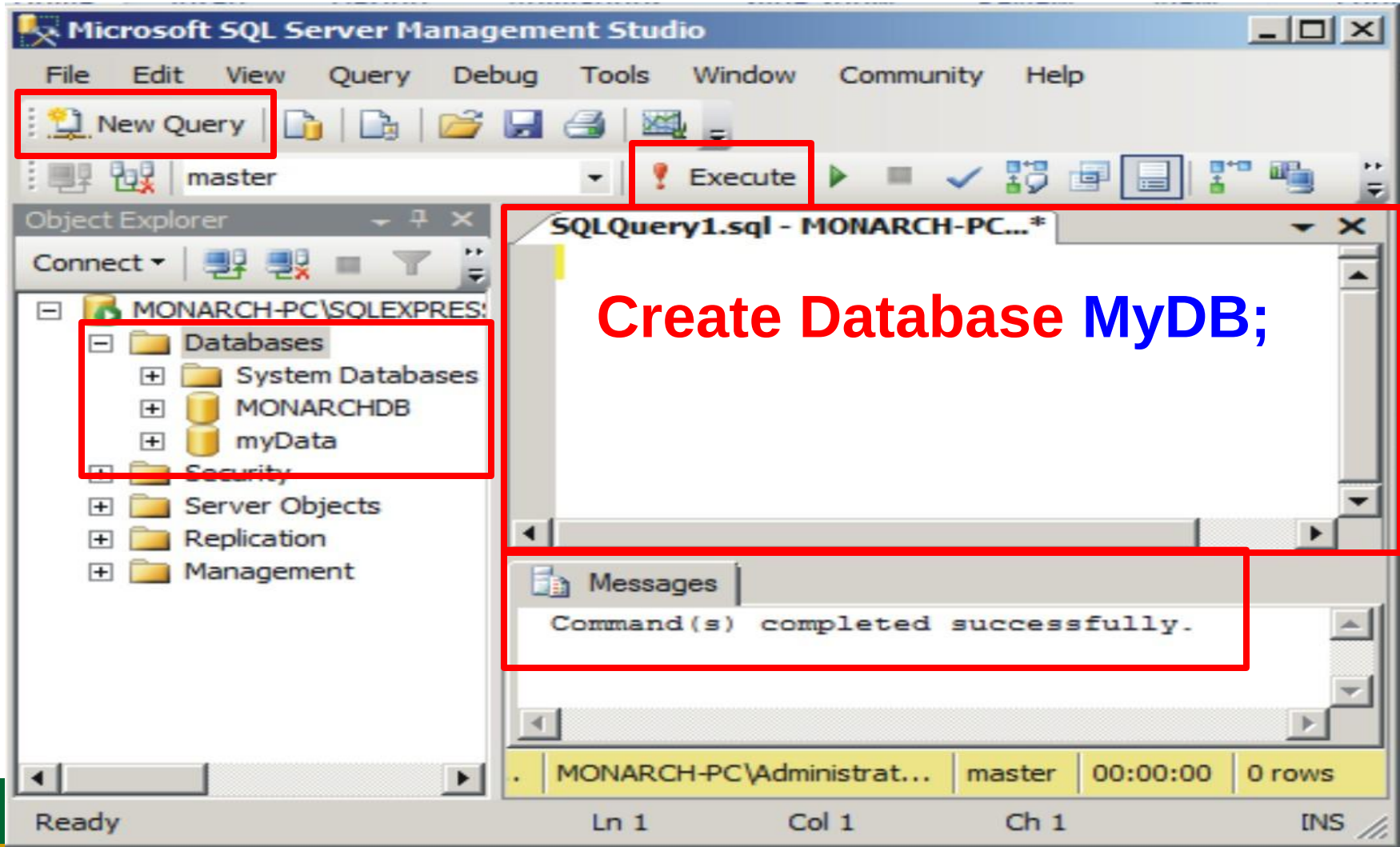
- From a physical design point of view, materialized views resemble tables or partitioned tables and behave like indexes in that they are used transparently and improve performance.

Implementation of the physical data model :

- This section describes the (DDL) Data Definition Languages statements that ware used to implement the physical data model.
- In this physical data model includes database partition groups, table spaces, tables, indexes, constraints and MQTs (Materialized Query Table).

SQL Commands :

- How to EXECUTE sql commands?



By

SQL Commands :

- Create a new database.
  - Syntax :

    CREATE DATABASE <databasename>;
  - Purpose :

    To create a new database we can use create database command.
  - Example :

    Create Database **MyDB**;

Def. Create New Databases as given and write queries.

- SchoolDB
- OfficeDB
- MediDB
- TravelsDB Etc…

Def. Create Tables for Informational system. With fields, data types, size Using SQL queries. Write ALL Queries.

- Database Name : SchoolMast
  - Tables : YearMaster
    - UserType
    - User
    - Departments
    - Class
    - Subjects
    - ExamMaster
    - StudentData
    - StaffData
    - Attendance
    - Complaints
    - Activities
    - Reminder