Unit 6 Physical File Design & Data Base Design

Lesson Structure

6.0 Objective

After going through this unit, you should be able to understand the:

- advantage of databases over files;
- difference between logical and physical design;
- rules for good database design practices;
- concepts of fields, records and database;

- how to design the fields and records in a database table;
- understand various constrains enforced during database design;
- typical database design concepts; and
- explain the concepts of records, record types, and files, as well as the different techniques for placing file records on disk.

6.1 Introduction

A database is a model of a thing in the real world. Like their physical model counterparts, data models enable you to get answers about the facts that make up the objects being modeled. Database design is the craft of relating things in the real world to data on a computer under the constraints and affordances of computer technology (read/write; supported data types, storage and access). Tellingly, popular database data types include booleans, strings, numbers, time – but not one explicitly for money. In addition to its use (in many forms) in what was once known as new media, databases pervade many parts of the post modern condition (electronic voting, banking records, known and unknown government "collection lists", etc). The strengths and danger of databases lies in the ability to selectively access and combine entries from large amounts of structured data. Furthermore, relationships between data entries can be found (via queries) in a way that reveals additional information. This is the domain of data mining.

Once the user finalized the logical system design, the process of physical design of the system can be started. Physical database design involves actual implementation of the logical database in the DBMS. The requirements of physical design of the system require the logical design of the system.

The database design involves three levels of design concepts namely:

Conceptual,

Logical and

Physical schemas.

Conceptual model produces a data model which accounts for the relevant entities and relationships within the target application domain.

Logical model ensures via normalization procedures and the definition of integrity rules that the stored database will be non-redundant and properly connected.

Physical model specifies how database records are stored, accessed and related to ensure adequate performance. A good database design helps efficient storage and retrieval of data.

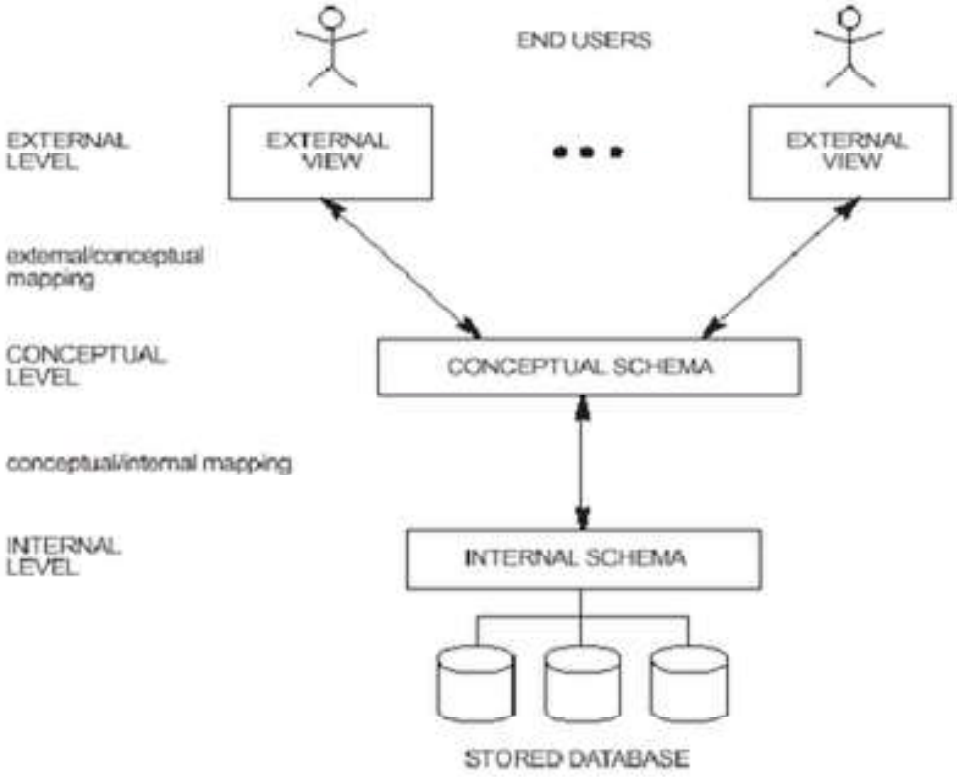


Figure 1

6.2 Introduction to Database Design

Database Management System (DBMS) is a collection of interrelated data.It consist of a set of programs to access the data . DBMS contains information about a particular enterprise.It provides an environment that is both convenient and efficient to use.Some of the Database Applications are as following:

- Banking: all transactions
- Airlines: reservations, schedules
- Universities:Registration,grade
- Manufacturing:production, inventory, order, supply chain.
- Sales:customers,products and purchases

6.2.1 Flat files vs. Database

DBMS (Database Management System) and File System are two ways that could be used to manage, store, retrieve and manipulate data. A File System is a collection of raw data files stored in the hard-drive whereas DBMS is a bundle of applications that is dedicated for managing data stored in databases. It is the integrated system used for managing digital databases, which allows the storage of database content, creation/ maintenance of data, search and other functionalities. Both systems can be used to allow the user

to work with data in a similar way. A File System is one of the earliest ways of managing data. But due the shortcomings present in using a File System to store electronic data, Database Management Systems came in to use sometime later, as they provide mechanisms to solve those problems. But it should be noted that, even in a DBMS, data are eventually (physically) stored in some sort of files.

File System

As mentioned above, in a typical File System electronic data are directly stored in a set of files. If only one table is stored in a file, they are called flat files. They contain values at each row separated with a special delimiter like commas. In order to query some random data, first it is required to parse each row and load it to an array at run time. But for this file should be read sequentially (because, there is no control mechanism in files), therefore it is quite inefficient and time consuming. The burden of locating the necessary file, going through the records (line by line), checking for the existence of a certain data, remembering what files/records to edit is on the user. The user either has to perform each task manually or has to write a script that does them automatically with the help of the file management capabilities of the operating system. Because of these reasons, File Systems are easily vulnerable to serious issues like inconsistency, inability for concurrency, data isolation, threats on integrity and lack of security.

DBMS

DBMS, sometimes just called a database manager, is a collection of computer programs that is dedicated for the management (i.e. organization, storage and retrieval) of all databases that are installed in a system (i.e. hard drive or network). There are different types of Database Management Systems existing in the world, and some of them are designed for the proper management of databases configured for specific purposes. Most popular commercial Database Management Systems are Oracle, DB2 and Microsoft Access. All these products provide means of allocation of different levels of privileges for different users, making it possible for a DBMS to be controlled centrally by a single administrator or to be allocated to several different people. There are four important elements in any Database Management System. They are the modeling language, data structures, query language and mechanism for transactions. The modeling language defines the language of each database hosted in the DBMS. Currently several popular approaches like hierarchal, network, relational and object are in practice. Data structures help organize the data such as individual records, files, fields and their definitions and objects such as visual media. Data query language allow for maintaining and the security of the database. It monitors login data, access rights to different users, and protocols to add data to the system. SQL

is a popular query language which is used in Relational Database Management Systems. Finally, the mechanism that allows for transactions help concurrency and multiplicity. That mechanism will make sure same record will not be modified by multiple users at the same time, thus keeping the data integrity in tact. Additionally, DBMSs provide backup and other facilities as well. With all these advancements in place, DBMS solves almost all problems of the File System, mentioned above.

Difference between DBMS and File System

In File System, files are used to store data while, collections of databases are utilized for the storage of data in DBMS. Although File System and DBMS are two ways of managing data, DBMS clearly has many advantages over File Systems. Typically when using a File System, most tasks such as storage, retrieval and search are done manually and it is quite tedious whereas a DBMS will provide automated methods to complete these tasks. Because of this reason, using a File System will lead to problems like data integrity, data inconsistency and data security, but these problems could be avoided by using a DBMS. Unlike File System, DBMS are efficient because reading line by line is not required and certain control mechanisms are in place.

6.2.2 Steps in Database Design

The process of database design is divided into different parts. It consists of a series of steps.They are

- Requirement Analysis
- Conceptual Database Design (ER-Diagram)
- Logical Database Design (Tables, Normalization etc)
- Physical Database design (Table Indexing, Clustering etc)

Requirement Analysis

In this phase a detailed analysis of the requirement is done.The objective of this phase is to get a clear understanding of the requirements.It make use of various information gathering methods for this purpose. some of them are

- Interview
- Analyzing documents
- Survey
- Site visit
- Joint Applications Design (JAD) & Joint Requirements Analysis (JRA)
- Prototyping

Conceptual Database Design

The requirement analysis is modeled in this conceptual design. The ER Model is used at the conceptual design stage of the database design.The ER

diagram is used to represent this conceptual design. ER diagram consists of Entities, Attributes and Relationships.

Logical Database Design

Once the relationships and dependencies are identified the data can be arranged into logical structures and is mapped into database management system tables. Normalization is performed to make the relations in appropriate normal forms.

Physical Database Design

It deals with the physical implementation of the database in a database management system. It include the specification of data elements, data types, indexing etc. All these information are stored in the data dictionary.

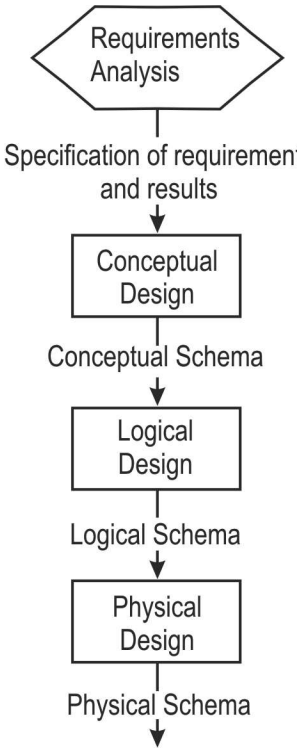


Figure 2

6.2.3 Inputs to Physical Database Design

1. Logical structure of Database (Normalized relations).
2. Definition of attributes – data type, integrity control, error handling.
3. Choice of RDBMS
   a. Hierarchical
   b. Network
   c. Relational (DB2, MySQL)
   d. Object relational (Oracle 8i/9i)
4. Estimation of database size growth rate and frequency of usage.
5. Requirements for backup, recovery, response time & retention time.

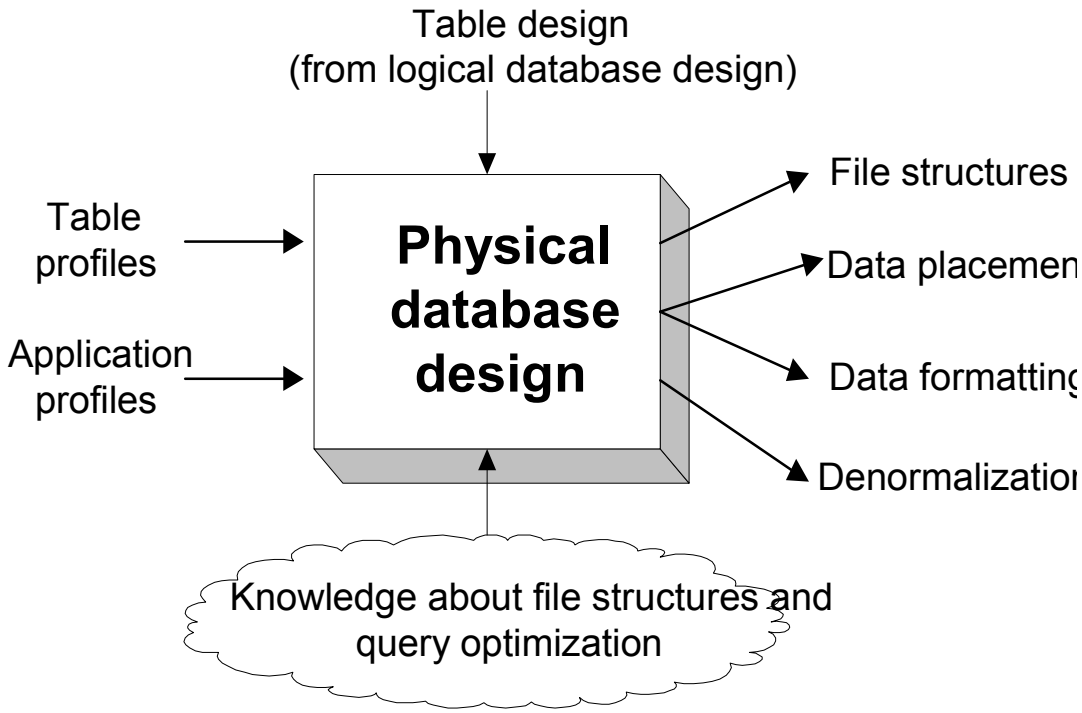


Figure 3: Inputs, Outputs and Environments of physical database design

6.2.4 Guidelines for Database Design

The following are various guidelines for Database Design :

- ensure that the data stored in the files (database tables) are atomic. Data stored in the atomic form can be combined later to generate data in specific form;
- every table must have a primary key which identifies each record in the table distinctly. Descriptive and meaningful name is to be used while naming a field in the table (For example, use product_id instead of ID);
- use single column primary key whenever possible. As most of the join operations are made on primary key and composite primary keys make the operation slower;
- use numeric key whenever possible;
- use primary key name as foreign key for better readability;
- avoid allowing null values to go into the columns that have discrete range of possible values; and
- avoid multiple tables with similar structure when one table is sufficient.

6.3 Design of Database Fields

A field is the basic unit of data entry in a record. To define a new field, you give it a name. Then you select options that determine how the fields

interprets, enters, calculates, stores, and displays data.After defining a field, you can set validation, auto entry, and storage options.

In the E-R model attributes are known as fields. A field is the smallest unit of data that is stored and manipulated. Fields are used in conventional files as well as in databases. It is the implementation of attributes and can be termed as smallest unit of meaningful data.

6.3.1 Types of Fields

The following are various types of fields in databases:

Primary key:

Attributes are data items that describe an entity. An attribute instance is a single value of an attribute for an instance of an entity. For example, Name and hire date are attributes of the entity EMPLOYEE. “Ram Kumar” and “3 March 2005” are instances of the attributes name and hire date.

Primary and foreign keys are the most basic components on which relational theory is based. Primary keys enforce entity integrity by uniquely identifying entity instances. Foreign keys enforce referential integrity by completing an association between two entities.

The primary key is an attribute or a set of attributes that uniquely identify a specific instance of an entity. Every entity in the data model must have a primary key whose values uniquely identify instances of the entity.

To qualify as a primary key for an entity, an attribute must have the following properties:

- it must have a non-null value for each instance of the entity
- the value must be unique for each instance of an entity
- the values must not change or become null during the life of each entity instance

In some instances, an entity will have more than one attribute that can serve as a primary key. Any key or minimum set of keys that could be a primary key is called a candidate key. Once candidate keys are identified, choose one, and only one, primary key for each entity.Candidate keys which are not chosen as the primary key are known as alternate keys.Lets consider an example of an entity that could have several possible primary keys is Employee. Let’s assume that for each employee in an organization there are three candidate keys: Employee ID, Social Security Number, and Name.In this case Employee ID and Social Security Number can be primary key but we will choose one and only one attribute as primary key.

Sometimes it requires more than one attribute to uniquely identify an entity. A primary key that made up of more than one attribute is known as a *composite key*. Figure 4 shows an example of a composite key. Each

instance of the entity Work can be uniquely identified only by a composite key composed of Employee ID and Project ID.

Employee ID	Project ID	Hours_Worked
2001	101	200
2001	201	120
2002	303	142
2002	601	24
3004	506	54
3004	903	25

Figure 4

Foreign keys are table attributes, the values of which are the same as those of primary keys of another table. It is often desirable to label foreign key columns explicitly. For instance, by adopting a naming convention. Existence of foreign key enforces the referential integrity constrains (discussed later in this Unit). A referential integrity constraint (references) should be declared as part of the CREATE statement in a DBMS while creating the table.

Most of modern RDBMS are tuned in for queries on integers, so it is advisable to use this datatype as a primary key. Many RDBMS provide a special serial number or sequence number of integer type, which generate a sequence of unique integers as a row is inserted into the table. Declaring a column to be of this type guarantees that a unique key is generated for each inserted row.

Descriptive fields: Attributes that are not used as key but store business data

Secondary key: Also known as Alternate key. This is a field or collection of fields in the table which can be used as primary key in addition to the already existing primary key.

Foreign Key:A *foreign key* is an attribute that completes a relationship by identifying the parent entity. Foreign keys provide a method for maintaining integrity in the data (called referential integrity) and for navigating between different instances of an entity. Every relationship in the model must be supported by a foreign key.

Every dependent and category (subtype) entity in the model must have a foreign key for each relationship in which it participates. Foreign keys are formed in dependent and subtype entities by migrating the entire primary key from the parent or generic entity. If the primary key is composite, it may not be split.

Figure 5 shows the existence of primary,foreign and composite key in the respective table

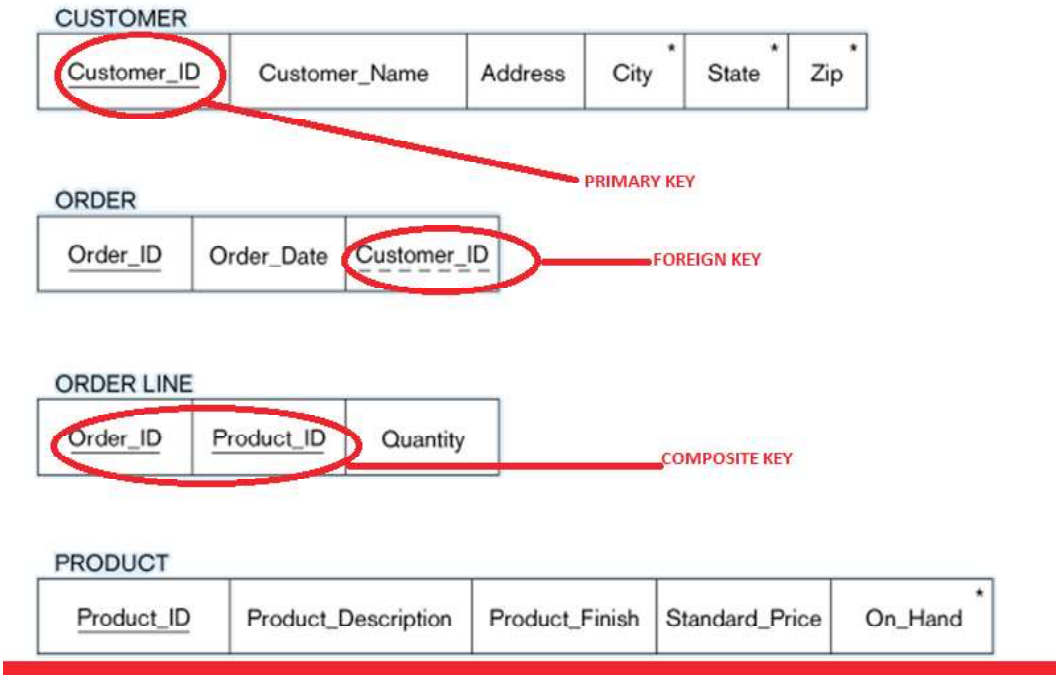


Figure 5

6.3.2 Rules for Naming Tables and Fields

The names for all database elements defined should be:

- Unique
- Meaningful
- Short

Restrictions for naming tables:

- Use no acronyms or abbreviations. Should be descriptive to convey meaning.
- Should not imply more than one subject

Restriction for naming fields:

- No acronyms
- Use abbreviations only if clear and meaningful
- Should not imply more than one subject
- Should be singular.

While designing database fields, it is required to set the properties of the fields which are as following

1. Name: A name is used to refer the attribute in the DBMS that uniquely labels the field. The name of the attribute in the logical data model and the name of the field in the physical data model must be same. For example, student name in a student table.

2. Data type: It defines the type of data the field is expected to store. This could be numeric, alphanumeric etc. The data type, supported by various RDBMS varies to a great extent. For example, employee_name CHAR(25), indicates that the name of the employee is of character data type, 25 indicates the maximum size of the data that can be stored in the field. The data type selected should ensure the following:

- it involves minimum usage of memory and represents all possible values
- supports all types of data manipulation that is expected from the business transaction.

3. Size: It indicates the size of the database fields. Many RDBMS support sizes that are variable. For example, VARCHAR data type in Oracle.

4. Null or not Null: specifies whether the field will accept null value. Not null constrains applied in DBMS ensure that null values are not entered to the respective fields. A null value is a special value distinct from 0 or blank. A null value indicates that the value is either missing or unassigned yet. We may specify that customer_name in a customer table to be not null. When a field is declared a primary key DBMS automatically ensures that the field in not null.

5. Domain: It indicates the range of values that are accepted by the fields. For example: Basic_Pay in a employee table can assume any value between the lowest basic_pay and highest basic_pay existing in the company. In such cases, the value of the field can be restricted to the one between the highest and lowest value to avoid entry of non-existing basic_pay.

6. Default value: It refers to the value that is stored by default in the field. For example, ship_date in a invoice is most of the time same as invoice_date (current date). When a default value is assigned to a field, it reduces a lot of data entry time and reduces the chances of error.

7. Referential integrity: It refers to a set of rules that avoid data inconsistency and quality problems. Referential integrity ensures that a foreign key value cannot be entered unless it matches a primary key value in another table. RDBMS automatically enforces the referential integrity once the database designer identifies and implements primary and foreign key relationship.

- It prevents orphaned records. i.e. when a row containing a foreign key is created, the referential integrity constrains enforced to the RDBMS ensure that the same value also exists as a primary key in the related table.
- When a row is deleted, it should be ensured that no foreign key in related tables is the same value as primary key of the deleted row of the primary table

Figure 6:depicts primary and foreign keys for two tables namely customer and order.

Customer_ID	Customer_Name	Address	City	State	zip
1001	Raj	Flat 102	Patna	Bihar	8004
3002	Abhi	Flat 202	Gaya	Bihar	8011

Table : Customer

Order_ID	Order_Date	*Customer_ID*
0342	04/01/2012	*1001*
0441	09/02/2012	*3002*

Figure 6 **Table : Order**

Customer_ID is the primary key in customer table and is a foreign key in order table. This referential integrity constraints ensure that the value customer_id in order table must exist in the customer table. The primary key is shown in bold and the foreign key in intalics.

6.4 Design of Physical Records

A physical record is a group of fields stored in adjacent memory locations and are retrieved together as a unit.They have fixed Length and variable fields.

A Record is a collection of fields. Records are common to both databases and files. Records are collection of fields in a predefined format. The design of physical record involves putting the collection of fields in a single logical unit so that the fields are stored in adjacent locations for better storage and retrieval.

The main objective of the design of physical records is to store and retrieve them efficiently. Also, the fields should be stored in adjacent locations in such a way that the storage is used efficiently and speed of data processing is appropriate**.**

Physical pages or blocks are units of information moved between disk and memory buffers. They hold not only records, or table entries, but other information such as the amount of free space currently available in the block, the starting position of each record, etc. Blocks of data (pages) are normally read or written by the operating system. Page is referred to as the amount of data written in one I/O operation of operating system.

Blocking factor refers to the number of physical records per page. If a record size is 1340 bytes and the page size is 2048 bytes, then 708 bytes are wasted if DBMS does not allow physical records to span different pages. Selecting a block size involves a trade-off. In principle, the larger the block size, the fewer read-write operations need be performed to access a file by the operating system and therefore the more efficient is the processing.

However, it requires a correspondingly large allocation of buffer space in memory. Since this is limited (and perhaps shared by many users), there is in practice, an upper bound. Moreover, large block sizes are primarily advantageous for sequential access.

Denormalization is the process of transforming normalized relations into unnormalized physical record specifications. The motivation behind de-normalization is poor performance of normalized table. The following may be of use for denormalization.

- Combine two entities with one-to-one relationship into one entity. This avoids the cost of joining two tables when the data are required from both the tables.
- Another form of de-normalization is to repeat the non key attribute (field) of one table in another table to facilitate the execution of query faster.

However, it depends on the application at hand.

Figure 7 depicts denormalization for optimized query processing.

Customer_ID	Customer_Name	Address	City	State	Zip	Order_ID	Order_Date
1001	Raj	Flat 102	Patna	Bihar	8004	**0342**	04/01/2012
3002	Abhi	Flat 202	Gaya	Bihar	80011	**0441**	09/02/2012

Figure 7

In a particular application it is seen that queries about order also require the cutomer_name. In case of normalized table, this would always require joining Customer table and order table each time the query is processed. We have modified the order table by adding back the customer_name from the customer table in order table. Now all queries will require only the order table as all relevant information are available in this table.

Activities to enhance performance

1. Combining tables to avoid joins
2. Horizontal partitioning refers to placing different rows of a table into separate files. For example, in an order table order pertaining to different regions can be kept in a separate table for efficient retrieval of records.
3. Vertical partitioning refers to placing different columns of a table into separate files by repeating the primary key in each of the files.
4. Record partitioning refers to a combination of both horizontal and vertical partitioning as in distributed database processing
5. Data replication refers to the same data being stored in multiple places in the database.

Process of Denormalization of Tables

- Select the dominant process based on frequency of execution and frequency of data access;
- Define the join table for dominant process;
- Evaluate the cost of query, updates and storage for the schema. Consider possibility of renormalization to avoid table joins.

Fixed length records and variable length records

In fixed length record format, the length of record is always the same. The fixed length records are easier to design and implement but are more wasteful than variable length record formats when comes to utilization of space.

In variable length record format, the records are of variable length. To identify the end of a record, variable length records use end of record marker. It can be a special character.

6.5 Design of Physical Files

As we know that Computers are used for storing the information for a permanent time or the Files are used for storing the Data of the users for a long time Period.The files can contains any type of information means they can Store the text, any Images or Pictures or any data in any Format. So that there must be some mechanism which can be used for storing the information, accessing the information and also performing some operations on the files.

Files are collection of logically connected records. In RDBMS, files are called tables. However, the way the files are stored in memory depend on the operating system. Many operating systems allow files to be split into pieces but the same is transparent to the user.

6.5.1 Types of Files

1. Master file

this type of file holds descriptive data; the actual data that is supposed to be processed and holds the resultant data after the process is completed (ex. names, addresses, sales, etc.). The data can be organized using keys.

2. Transaction file

It contains the transactions; changes that are supposed to be made to the data in the master file.

- In batch processing all transactions are collected in the transaction file and the changes are applied to the master file sequentially in a single pass. For this to be possible, both the master and transaction file have to be sorted first.

- In an online system the changes are applied to the master file the moment the transactions occur or are recorded.

Algorithm to update the Master file

Algorithm 1.

1. Order transaction file by key.
2. Create new master file.
3. Compare first keys in master file to first key in transaction file.
4. Continue this until end of both master and transaction file.

Algorithm 2.

1. Apply any changes defined by the transaction file to the record.
2. Write the record with the lowest key to the new master file.
3. Go to the next record in the original master file and the transaction file.
4. Compare keys as in step 3.
5. Closeall files.

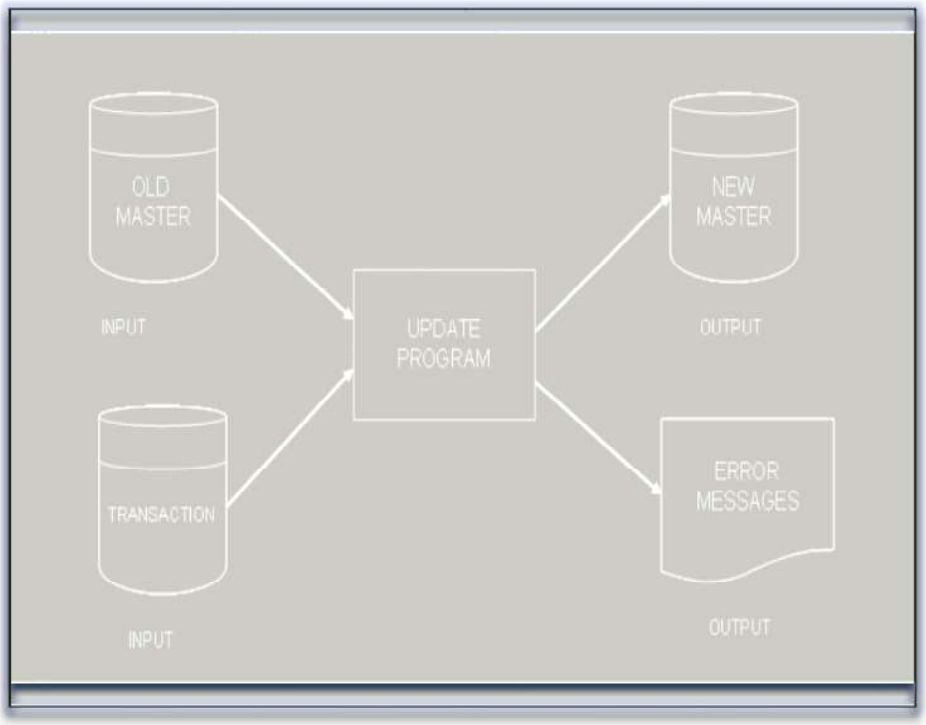


Figure 8

3. Archive file

An Archive file is a file of data in permanent storage (usually, the data is stored for legal reasons or to perform trend analysis). Archive files will contain all information about the past dealings of the business and would normally be stored on a different site to facilitate recovery in case of a disaster such as fire.

4. Audit file

An Audit file is a file that does not store business data but data related to transaction log. For example, data and time of access, modification etc. of data , values of fields before and after modification etc.

5. Work file

A Work file is file temporarily created to hold intermediate result of the data processing. For example, a sorted file of list of customers.

6.5.2 File Organization

The physical organization of records on the disk is known as file Organization. There are different types of file organizations depending on the organization of records in the disk and other secondary storage are following 1. Serial file organization, 2. Sequential file organization, 3. Indexed sequential file organization, 4. Hashed file organization.

Before deciding on a specific file organization, we should ensure that its application leads to the following:

- Fast retrieval of records
- Reduce disk access time
- Efficient use of disk spaces.

1. Serial file organization

A serial file is created by placing the record as it is created. It leaves no gap between the records that are stored on the disk. The utilization of space called packing density approaches 100 percent in this case. Examples of serial files are print file, dump file, log files, and transaction files. These files are created once and are not used for addition or deletion or any kind of record searching operation.

2. Sequential file organization

In this organization, the records are physically ordered by primary key. To locate a particular record, the program starts searching from the beginning of the file till the matching primary key is found. Alphabetic list of customers is a common example of sequential file organization. Deletion of record may cause wastage of space and adding a new record requires rewriting of the file. This type of file organization is suitable for master files and is not used where fast response time is required.

3. Indexed sequential file organization

Records are not physically ordered in this type of organization. Index is created to facilitate searching of records. Index Records give physical location of each data record. Indexes are separate files with a link to the main file. This type of file organization is used where faster response time is required

4. Hashed file organization

Hashing algorithm is used to physically ordered the records. The address where each record is stored is determined using hashing algorithms.

The following is a typical hashing algorithm :

1. Uses a field in record called the hash field (generally the key filed).
2. Divides by prime number known as hash function.
3. Produces record address called the hash address.

6.6 Design of Database

Database design is similar to the pillars of a building. Any negligence, errors in Database design may lead to degraded performance of the software. In some cases such as real time applications, it may also lead to disasters.

The following are various steps in Database design :

- Selection of database architecture
- Designing database schema
- Selecting indexes
- Estimating capacity of the database.

Selection of database architecture

Selecting database architecture is one of the most challenging parts of database design for any information system. Before deciding on the target DBMS where the database is to be implemented, few considerations are required.

Hierarchical database structure is a kind of database management system that links records in tree data structure such that each record has only one owner. For example an order is owned by only one customer.

Network database structure is more flexible structure than Hierarchical model as well as relational model, but not preferred due to the high processing time.

Relational database structure is most commonly used database model. Reference key joins different tables together. Indexes provide rapid access to specific record in the database. For example, DB2, MySQL, Oracle are some RDBMS.

Object Oriented Database management system is based on object oriented paradigm. In object oriented database, data is stored as objects and can be interpreted only by using the methods specified by its class.

A blue print of the database is a physical model. A schema defines the database in terms of tables, keys, indexes and integrity rules. A relational schema consists of a relation(table), name of the attributes in the relations and restrictions on the relations called *integrity constraints*.

A **database schema** is a set of relation schemas. Changes to a schema or database schema are expensive. So, careful thought must be given to

design of a database schema. The following are some guidelines for the design of a database schema.

- Each entity should be implemented as a database table.
- Each attribute should be implemented as a field.
- Each table must have a primary key and an index based on the key.
- Each table may have zero or more secondary keys.
- Appropriate foreign keys.

Estimating capacity of the database:

Database administrator needs to calculate the amount of disk space required for the database.It can be calculated by using the following formula:

Table size = record size * number of records in the table.

where the number of records that will be present in each table at a particular period of time should be forecast.

Database size is sum of sizes of all tables in that database. As rule of thumb, add a factor of 50% for indexes and other overheads to get the expected database size. While designing a database, future growth of database should also be kept in mind. Most of the business databases have a liner growth trend.

6.7 Case Study

The physical database design is the process of transforming a logical data model into an actual working physical database. A logical data model is required before designing a physical database.

The very first step is to create an initial physical data model by transforming the logical data model into a physical implementation based on the target DBMS to be used for deployment. In reality, the physical data model depends largely on the target DBMS to be used for implementing the database. Therefore, to successfully create a physical database design requires a good working knowledge of the features of the DBMS including:

- knowledge of the database objects supported by DBMS, the physical structures and files required to support those objects;
- details regarding how the target DBMS supports indexing, referential integrity, constraints, data types, and other features that augment the functionality of database objects;
- knowledge of the DBMS configuration parameters and limitations; and
- data definition language (DDL) to translate the physical design into actual database objects.

The following are the steps to transform logical data model into physical model:

- An effective and efficient database is created from a logical data model,i.e. E-R diagram i.e.a simple translation from logical terms to physical objects;
- transforming entities in ER diagram into database tables;
- transforming attributes into table columns; and
- transforming domains into data types and constraints to primary key and foreign key relationship.

Deciding a primary key is an integral part of the physical design of entities and attributes. A primary key should be assigned for every entity in the logical data model. One should try to use the primary key as selected in the logical data model. However, if suitable attribute is not available, multiple attributes can be used as primary key. It may be required to choose a primary key other than the attributes in logical design by inserting a column that does not store business data (surrogate key) for physical implementation.

In a physical database, each table column must be assigned a data type supported by the DBMS. Certain data types require a maximum length to be specified, e.g.,a character data type could be specified as CHAR(25), indicating that up to 25 characters can be stored in the column.

Hence we can summarise it as

A Relational Model is made up of tables

1. A row of table = a relational instance/tuple
2. A column of table = an attribute
3. A table = a schema/relation
4. Cardinality = number of rows
5. Degree = number of columns

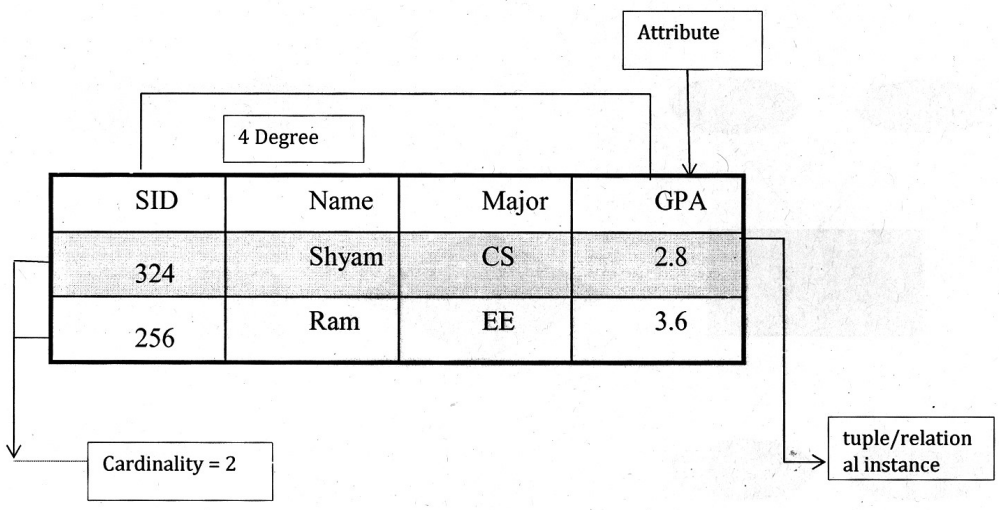
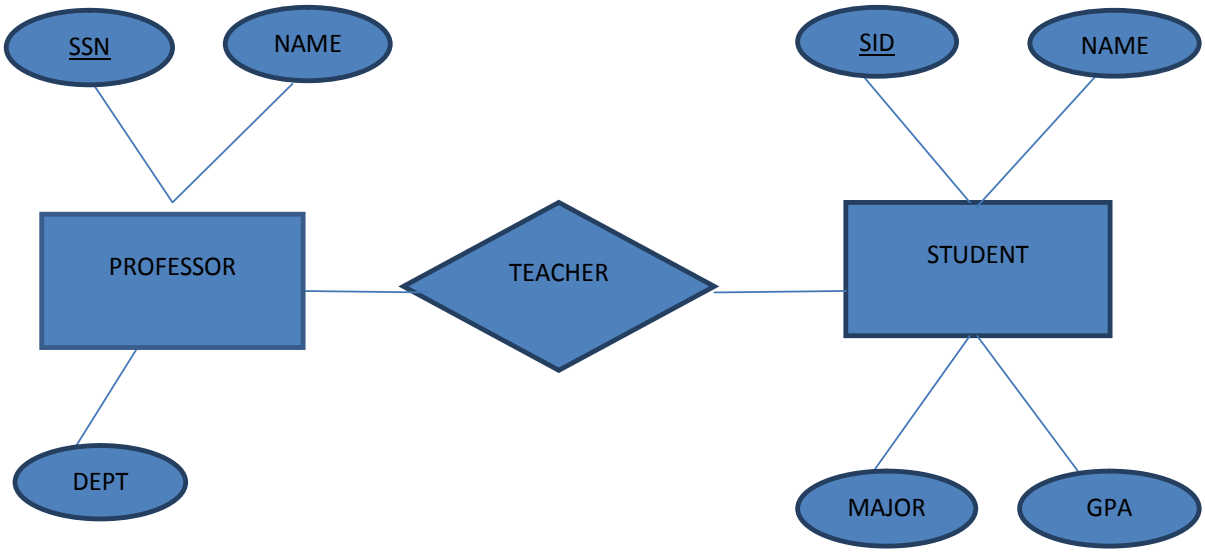


Figure 9 **A Schema/Relation**

Conversion from ER Model to Relational Model

Basic Ideas:

- Build a table for each entity set
- Build a table for each relationship set if necessary (more on this later)
- Make a column in the table for each attribute in the entity set
- Indivisibility Rule and Ordering Rule
- Primary Key



SSN	NAME	DEPT
9999	Mr. Mahesh	MATH
8888	Mr. Suresh	CS

SID	NAME	MAJOR	GPA
324	SHYAM	CS	2.8
256	RAM	EE	3.6

Figure 10

The aim of database design should be to create a database plan to fit present and future objectives. A logical data model should be used as the blueprint for designing and creating a physical database. But, the physical database cannot be created properly with a simple logical to physical mapping.

6.8 Summary

File is being used for keeping data.A database management system (DBMS) is a computer application program designed for the efficient and effective storage, access and update of large volumes of information. Use of DBMS has been the standard to store data for today's information systems due to their various advantages. Relational database is mostly being used unless the application has specific requirements. The physical database design process can be considered as a mapping from logical model to physical working database, which involves design of fields, design of records and finally design of the database. While transforming the logical model to physical model, many implementation issues related to the information system and target DBMS are to be addressed. Database volume estimation is an important part of database design. The present size and future growth of database is to be estimated before implementing the database.The transformation of E-R model to relational database model is also discussed in this unit.

6.9 Questions for Exercise

1. What are data and information, and how are they related in a database?
2. What is a DBMS?What do you mean by Primary key,Composite Key and Foreign Key.
3. Define File Organization.What are the different types of organization.
4. What do you mean by Physical files.What are the different types of file.
5. What do you mean by database design.

6.10 Further Readings

1. "Database System Concepts" by Abraham Silberschatz, Henry Korth, and S. Sudarshan
2. Jeffrey L. Whitten, Lonnie D. Bentley, Kevin C. Dittman; Systems Analysis and Design Methods; Tata McGraw Hill; Fifth Edition; 2001.
3. "Database Management Systems" by Raghu Ramakrishnan
4. "An Introduction to Database Systems" by Bipin Desai
5. Joey George, J Hoffer and Joseph Valacich; Pearson Education, Modern System Analysis and Design;2001.

Reference Websites

- http://lms.ibu.edu.ba/
- www.webopedia.com/
- www.wikipedia.org/wik
- http://seastorm.ncl.ac.uk/itti/create.html#create
- http://www.rspa.com

●●●