

CS403 Short Notes Mid term

Lec 1 to22

BY vuonlinehelp.blogspot.com

YOUTUBE LINK:

WEBSITE LINK:

[HTTPS://VUONLINEHELP.BLOGSPOT.COM/](https://vuonlinehelp.blogspot.com/)

IMPORTANT NOTE

You should never copy paste the same answer which we are providing. Please do make some changes in the solution file. If you appreciate our work, please share our website vuonlinehelp.blogspot.com with your friends.

آپ کو کبھی بھی وہی جواب کاپی پیسٹ نہیں کرنا چاہیے جو ہم فراہم کر رہے ہیں۔ براہ کرم حل فائل میں کچھ تبدیلیاں کریں۔ اگر آپ ہمارے کام کی تعریف کرتے ہیں تو براہ کرم ہماری ویب سائٹ vuonlinehelp.blogspot.com کو اپنے دوستوں کے ساتھ شیئر کریں

Lecture No. 01

Database definitions:

VERY Important

Definitions are important, especially in technical subjects because definition describes very comprehensively the purpose and the core idea behind the thing. Databases have been defined differently in literature.

Def 1: A shared collection of logically related data, designed to meet the information needs of multiple users in an organization. The term database is often erroneously referred to as a synonym for a “database management system (DBMS)”. They are not equivalent and it will be explained in the next section.

Def 2: A collection of data: part numbers, product codes, customer information, etc. It usually refers to data organized and stored on a computer that can be searched and retrieved by a computer program.

Def 3: A data structure that stores metadata, i.e. data about data. More generally we can say an organized collection of information.

Def 4: A collection of information organized and presented to serve a specific purpose. (A telephone book is a common database.) A computerized database is an updated, organized file of machine readable information that is rapidly searched and retrieved by computer.

Def 5: An organized collection of information in computerized format.

Def 6: A collection of related information about a subject organized in a useful manner that provides a base or foundation for procedures such as retrieving information, drawing conclusions, and making decisions.

Def 7: A Computerized representation of any organizations flow of information and storage of data.

Importance of the Databases

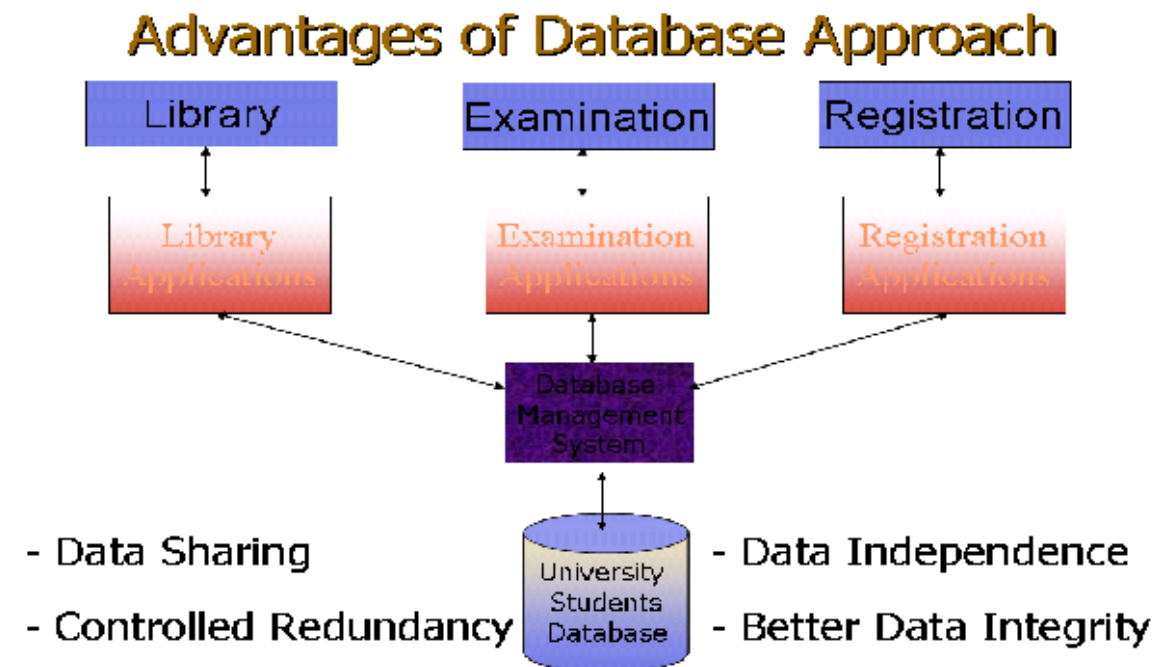
Traditionally computer applications are divided into commercial and scientific (or engineering) ones. Scientific applications involve more computations, that is, different type of calculations that vary from simple to very complex.

Databases and Traditional File Processing Systems

Traditional file processing system or simple file processing system refers to the first computer-based approach of handling the commercial or business applications. That is why it is also called a replacement of the manual file system.

Advantages of Databases

It will be helpful to reiterate our database definition here, that is, database is a shared collection of logically related data, designed to meet the information needs of multiple users in an organization. A typical database system environment.



Lecture No. 02

Difference between Data and Information

Data is the collection of raw facts collected from any specific environment for a specific purpose. Data in itself does not show anything about its environment, so to get desired types of results from the data we transform it into information by applying certain processing on it. Once we have processed data using different methods data is converted into meaningful form and that form of the Data is called information.

Database Application:

A database application is a computer program whose primary purpose is retrieving information from a computerized database. From here, information can be inserted, modified or deleted which is subsequently conveyed back into the database.

Database Management Systems:

Database management system is software of collection of small programs to perform certain operation on data and manage the data.

Two basic operations performed by the DBMS are:

- Management of Data in the Database
- Management of Users associated with the database.

Further Advantages of Database Systems:

Database management systems help users share data quickly, effectively, and securely across an organization. By providing quick solutions to database queries, a data management system enables faster access to more accurate data. Database systems are very much beneficent to enterprises and businesses, some of the advantages are listed below:

Data consistency

- Better data security
- Faster development of new applications
- Economy of scale
- Better concurrency control
- Better backup and recovery procedures

Cost Involved:

These charges may also be known as the disadvantages of the database system. Different types of costs (Financial and Personnel) which an organization faces in adopting a database system are listed below:

- High Cost:
- Conversion Cost:
- Difficult Recovery Procedures:

Levels in database

Three Level Architecture of Database

Real World Data

The real world level of data means that level of data at which entities or objects exist in reality,

Meta Data:

For storage of the data related to any entity or object existing at real world level we define the way the data will be stored in the database. This is called Meta data. Meta data is also known as schema for the real world data

Existence of Data:

Existence of the data level shows the actual data regarding the entities as real world level according to the rules define at the Meta Data level.

Users of Database Systems:

Application Programmers

End Users

Application programmers

Database users are the one who really use and take the benefits of database. There will be different types of users depending on their need and way of accessing the database. Application Programmers. They are the developers who interact with the database by means of DML queries.

End Users:

Second category of the Database users are the end users, this group of users contains the people who use the database application programs developed by the Application programmers. This category further contains two types of users:

- Naïve Users
- Sophisticated Users

Naïve Users

Naïve Users are unsophisticated users, who have no knowledge of the database. These users are like a layman, which has a little bit of knowledge of the database. They are the end-users of the database who work through the menu-driven applications.

Sophisticated Users:

Sophisticated users are those who have their own way of accessing the database. This means they do not use the application program provided in the system. Instead, they might define their own application or describe their need directly by using query languages.

Duties of the DBA

A Database administrator has some very precisely defined duties which need be performed by the DBA very religiously. A short account of these jobs is listed below:

- Schema definition
- Granting data access
- Routine Maintenance
 - Backups
 - Monitoring disk space
 - Monitoring jobs running

Schema Design

DBA in some organization is responsible for designing the database schema, which means that DBA is the person who creates all the meta data information for the organization on which the database is based.

Granting Access to Users:

More Files Join VU WhatsApp Group: <https://chat.whatsapp.com/BL4ok4jnM9iJiglhAA1SG3>

DBA is also responsible for grant of access rights to the database users. Along with granting and revoking (taking back) the rights the DBA continuously monitors and ensure the legal use of these rights.

Routine Maintenance

Database Maintenance is a term we use to describe a set of tasks that are all run with the intention to improve your database. There are routines meant to help performance, free up disk space, check for data errors, check for hardware faults, update internal statistics, and many other obscure (but important) things.

Lecture No. 03

Database Architecture:

A Database Architecture is a representation of DBMS design. It helps to design, develop, implement, and maintain the database management system. A DBMS architecture allows dividing the database system into individual components that can be independently modified, changed, replaced, and altered.

Database uses architecture

Database architecture uses programming languages to design a particular type of software for businesses or organizations. Database architecture focuses on the design, development, implementation and maintenance of computer programs that store and organize information for businesses, agencies and institutions.

The Architecture:

Internal schema or view defines the structures of the data and other data related activities in the database. For example it defines that for a student what data will be stored in terms of attributes of the student and it also defines how different values for these attributes will be stored, also it tells that who is allowed to make changes to the database and what changes he can make, etc.

Intention

These details give us the internal schema and are called the intention of the database. Intention for a database is almost permanent.

Extension

Extension of the database is performed on the bases of a complete intention, i-e once a database has been defined it is populated with the data of the organization for which the database is created. This population of the database is also called as the extension of the database. Extension

is always done according to the rules defined in the internal schema design or the intention of the database.

External View (Level, Schema or Model):

This level is explicitly an end user level and presents data as desired by the users of the database. As it is known that the database users are classified on two grounds.

- Section of the organization
- Nature of Job of the users

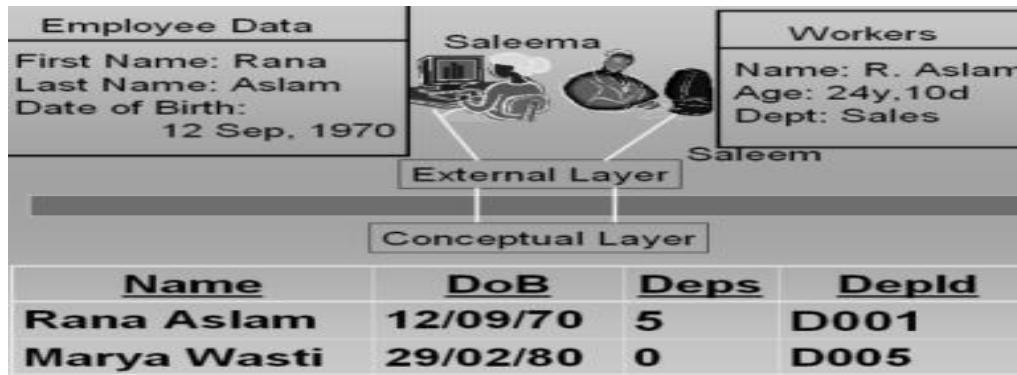
The external level of the database caters to the needs of all the database users. External level of the database might contain a large number of user views, each user view providing the desired features and fulfilling requirements for the user or user group for which it is intended. External views are also helpful when we want to display the data which is not place in the database or not stored at all.

SO External views are designed and create in way that they can be modified at a later stage without making any changes in the logical or internal views. We can see that the same data record is displayed in two entirely different ways.



Conceptual or Logical View:

A complete description of the information content of the database the entire information structure of the database, as seen by the DBA the community view of data all entities, attributes and their relationships are represented here Contains record types representing entities, data item types with their attributes, relationships and constraints on data. Contains Semantic information about the data meaning, security and integrity information *relatively constant* designed with the present as well as future needs of an organization.



Lecture No. 04

Internal or Physical View / Schema

The internal schema is also known as a physical schema. It uses the physical data model. It is used to define that how the data will be stored in a block. The physical level is used to describe complex low-level data structures in detail. Binary storage mechanism is always implemented by the Operating System of the Computer. DBMS to some extent decides the way data is to be stored on the disk. This decision of the DBMS is based on the requirements specified by the DBA when implementing the database.

Functions of DBMS

LONG or SHORT

- Data Processing
- A user accessible Catalog
- Transaction Support
- Concurrency Control Services
- Recovery Services
- Authorization Services
- Support for Data Communication
- Integrity Services

DBMS it is necessary that we understand the relation of database and the DBMS and the dissection of the set of functions the DBMS performs on the data stored in the database. Two important functions that the DBMS performs are:

- User management
- Data Management

DBMS Environments:

LONG

- Single User

- Multi-user
 - Teleprocessing
 - File Servers
 - Client-Server

Single User Database Environment

The DBMS might have a number of users but at a certain time only one user can log into the database system and use it. This type of DBMS systems are also called Desktop Database systems.

Multi-User Database systems

This is the type of DBMS which can support a number of users simultaneously interacting with the database in different ways. A number of environments exist for such DBMS.

- **Teleprocessing**
This type of Multi user database systems processes the user requests at a central computer, all requests are carried to the central computer where the database is residing, transactions are carried out and the results transported back to the terminals (literally dumb terminals).
- **File Servers**
This type of multi-user database environment assumes another approach for sharing of data for different users. A file server is used to maintain a connection between the users of the database system.
- **Client-Server**
This type of multi-user environment is the best implementation of the network and DBMS environments. It has a DBMS server machine which runs the DBMS and to this machine are connected the clients having application programs running for each user.

Lecture No. 05

Database Application Development Process

These three steps cannot always be considered as three independent steps performed in a sequence or one after another Database Application development Process includes the Following Stages or steps:

- Database Design
- Application Programs
- Implementation

Database Design:

Database design is the organization of data according to a database model. The designer determines what data must be stored and how the data elements interrelate. With this information, they can begin to fit the data to the database model. Database management system manages the data accordingly.

What is database application program and examples?

Application Programs

A database application can be used for storing or retrieving data, processing transactions, or various machine learning calculations. For example, Facebook has a user database with which it authenticates users when they log into their Facebook account.

Implementation

Database implementation is the process of installing database software, performing configuration and customization, running and testing the database and then integrating it with applications. Finally, the implementation process involves training the users.

Preliminary Study:

Preliminary Stage, which is based on the proper study of the system. It means that all the parts of the systems, or the section of the subject organization for which we intend to develop the system must be studied. We should find the relation or interaction of different section of the organization with each other and should understand the way information flows between different sections of the organization.

Tools Used for Database System Development:

Tools are used for describing the design process in standard ways. If there is no standardized tool available for designing a specific systems; Then everyone will have to use its own design notation, and a notation used by one designer may not be understandable to the another one.

Data Flow Diagrams:

The most common tool used for designing database systems is Data Flow Diagram. It is used to design systems graphically and expresses different system detail in different DFD levels.

Limitation of DFDs:

They do not provide us a way of expressing decision points. DFDs are focused on flow of information only.

Symbols used in DFD:

There are a limited number of symbols which are used for design process in DFDs.

DATAFLOW:

The purpose of the dataflow in a DFD is to express the flow of information from one entity to another entity in the system Data flows are pipelines through which packets of information flow.

Types of DFD

- Context diagram

More Files Join VU WhatsApp Group: <https://chat.whatsapp.com/BL4ok4jnM9iJiglhAA1SG3>

- Level 0 diagram
- Detailed diagram

Context Diagram

DFD Level 0 is also called a Context Diagram. It's a basic overview of the whole system or process being analyzed or modeled. It's designed to be an at-a-glance view, showing the system as a single high-level process, with its relationship to external entities.

Level 0 Data Flow Diagrams:

The level 0 Diagram in the DFD is used to describe the working of the whole system. Once a context DFD has been created the level zero diagram or level 'not' diagram is created. The level zero diagram contains all the apparent details of the system. It shows the interaction between a numbers of processes and may include a large number of external entities.

Detailed diagram

In the detailed Data Flow and is given a number. Numbering processes is based on a specific notation, in the level 0 diagrams only left half or the portion before the decimal point is valid but in the detailed diagram when a complex process is expressed further its sub processes are number like 1.0, 1.1, and 1.2 and so on.

Lecture No. 06

Detailed Data Flow Diagram:

This Type of the Data flow diagrams is used when we have to further explain the functionality of the processes that we showed briefly in the Level 0 Diagram.

- The special features associated with this diagram are that, one, it is optional, and that is, it is created for only those processes from the level 0 diagram for which we want to show the details.
- Second specific characteristic of the detailed DFD is its processes' numbering. Numbering of processes in the detailed DFD is done on the basis of numbering of the particular process in level 0 diagrams whose sub-processes are being included in the detailed DFD.

Data Dictionary

LONG

A database that containing data about all the databases in the database system. Data dictionaries store all the various schema and file specifications and their locations.

Types of data dictionary

There are two types of data dictionaries:

- Integrated
- Free Standing

Integrated

The first type of data dictionary in this context is the integrated data dictionary. Such a data dictionary is place embedded into the database system, and is created by the DBMS for its usage under the directions and requirements provided by the DBA.

Free Standing

Second type of data dictionary is free standing data dictionary create by any CASE tool and then attached to the database management systems.

Cross Reference Matrix

In the cross reference matrix, on the Y axis we specify the accessible components of the database such as transitions, reports, or database objects and on the x axis we specify the attributes that will be accessed in the corresponding accessed object.

Outcome of the Analysis Phase

In the preliminary study phase, database designers collect information about the existing system from the users of the system. Designers represent their understanding of the working of existing system in the form of DFDs and discuss it with the users to make it sure that they have understood all details of the existing system and the requirements of different users groups.

Database Design Phase

LONG

Database design phase follows the analysis phase. Before starting the discussion on the design activity, it will be wise if we clearly understand some basic concepts that are frequently used in this phase.

- Database Design /Database Mode
These terms can be used interchangeably for the logical structure of the database. The database design/model stores the structure of the data and the links/relationships between data that should be stored to meet the users' requirements. Database design is stored in the database schema, which is in turn stored in the data dictionary.
- Database modeling
The process of creating the logical structure of the database is called database modeling. If the database is not designed properly the implementation of the system cannot be done properly. Generally the design of the database is represented graphically.

Data Model

LONG

Data model is a set or collection of construct used for creating a database and producing designs for the databases. There are a few components of a data model:

- **Structure:**
What structures can be used to store the data is identified by the structures provided by the data model structures.
- **Manipulation Language**
For using a certain model certain data manipulations are performed using a specific language. This specific language is called data manipulation language.
- **Integrity Constraints**
These are the rules which ensure the correctness of data in the database and maintain the database in usable state so that correct information is portrayed in designing the database.

Significance of the Data Model

Data model is very important tool because it is something which is used for designing the database for a DBMS and no DBMS can exist independent of any data model, now if we use a specific DBMS but are not sure about the data model it uses for database usage, we cannot create a proper database.

Types of Data Model

The three primary data model types are relational, dimensional, and entity-relationship (E-R). There are also several others that are not in general use, including hierarchical, network, object-oriented, and multi-value.

Types of Database Design

LONG

- **Conceptual database design**
This design is implemented using a semantic data model, for example for creating a design for an organization database we can use and we do use the ER-Data model.
- **Logical Database design**
This design is performed using a data model for which we have a DBMS available and we are planning to run our database system that DBMS.
- **Physical Database Design:**
The Logical design created using a specific data model and created after the analysis of the organization, it needs to be implemented in a physical DBMS software so the Physical database design is performed and the design created so far in the logical form are implemented on that very DBMS.

Lecture No. 07

Entity-Relationship Data Model

The entity relationship (ER) data model has existed for over 35 years. It is well suited to data modeling for use with databases because it is fairly abstract and is easy to discuss and explain. ER models are readily translated to relations. ER models, also called an ER schema, are represented by ER diagrams.

ER modeling is based on two concepts:

- Entities, defined as tables that hold specific information (data)
- Relationships, defined as the associations or interactions between entities

Constructs in E-R Data Model

E-R data model supports following major constructs:

- ❖ Entity
- ❖ Attribute
- ❖ Relationship

The Entity

Entity is basic building block of the E-R data model. The term entity is used in three different meanings or for three different terms and that are:

- Entity type
- Entity instance
- Entity set

Entity Type

The entity type can be defined as a name/label assigned to items/objects that exist in an environment and that have similar properties. All these items are quite different from each other by their respective nature, still they may be considered the instances of the same entity type since they are all donated or purchased or manufactured. The process of identifying entity types, their properties and relationships between them is called abstraction.

Entity Instance

Whereas an entity type represents an abstract category, an entity instance is a manifestation of an entity within that category. For example, Cell could be the entity type, but Cell_1, Cell_2, and Cell_3 would represent the actual items within the network.

Entity Types	Properties	Instances
EMPLOYEE	Human being, has name, has father name, has a registration number, has qualification, designation	M. Sharif, Sh. Akmal and many others
FURNITURE	Used to sit or work on, different material, having legs, cost, purchased	Chair, table etc.
ELECTRIC APPLIANCES	Need electricity to work, purchased	Bulb, fan, AC
OFFICE EQUIPMENT	Used for office work, consumable or non-consumable,	Papers, pencil, paper weight etc.

Entity Set

A group of entity instances of a particular entity type is called an entity set. For example, all employees of an organization form an entity set. Like all students, all courses, all of them form entity set of different entity types.

Classification of entity types

The entity can be classified into two basic categories, strong entity set and weak entity set. An entity set that has a primary key is the strong entity set. An entity that doesn't have a primary key is a weak entity set. A weak entity set is dependent on one strong entity set to distinguish its entities whereas weak ETs are also called dependent ETs.

Weak Entity Types

An entity type whose instances cannot exist without being linked with instances of some other entity type, i.e., they cannot exist independently.

Strong Entity Type

- A strong/regular entity type is the one whose instances can exist independently, i.e., without being linked to other instances.
- Strong ETs have their own identity.

Naming Entity Types

Following are some recommendations for naming entity types.

- Singular noun recommended
- Organization specific names
- Write in capitals
- Abbreviations can be used, be consistent

Symbols for Entity Types

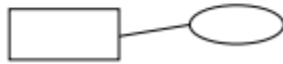
A rectangle is used to represent an entity type in E-R data model. For strong entity types rectangle with a single line is used whereas double lined rectangle is drawn to represent a weak entity type.

Attribute

In general, an attribute is a characteristic. In a database management system (DBMS), an attribute **refers to a database component, such as a table**. It also may refer to a database field. Attributes describe the instances in the column of a database.

Symbols for Attributes

Each represented as an oval, linked with an ET symbol



Types of Attributes

Attributes may be of different types. They may be:

- Simple or Composite
- Single valued or multi-valued
- Stored or Derived

Simple or Composite

Simple attributes are atomic values, which cannot be divided further. For example, a student's phone number is an atomic value of 10 digits. Composite attribute – Composite attributes are made of more than one simple attribute. For example, a student's complete name may have first_name and last_name.

Single valued or multi-valued Attributes:

The attributes that can have only one value for a given entity are called the single-valued attributes. For example, the attribute Book_title is a single-valued attribute as one book can have only one title. The attributes that can have multiple values for a given entity are called multi-valued attributes.

Stored or Derived Attributes:

Stored attributes are those attributes that are stored in the physical database for e.g date of birth. Derived attributes are those attributes that are not present in the physical database, but can be derived from the attributes present in the database. For e.g. age can be derived from date of birth.

Lecture No. 08

Simple or Composite Key

A key consisting of single attribute is called simple key, e.g., StudID, itemNo. A key consisting of more than one attribute is known as composite key, like {Program_Code, Course_Code}.

Key Attributes

A key is a set of attributes that can be used to identify or access a particular entity instance of an entity type (or out of an entity set). The concept of key is beautiful and very useful an attribute or set of attributes to identify an entity instance uniquely Types.

- Super key
- Candidate key
- Alternate Keys
- Primary key
- Secondary and Alternate keys

Super Key

A super key is a set of one or more attributes which taken collectively, allow us to identify uniquely an entity instance in the entity set. This definition is same as of a key, it means that the super key is the most general type of key.

Candidate key

A super key for which no subset is a super key is called a candidate key, or the minimal super key is the candidate key. It means that there are two conditions for the candidate key, one, it identifies the entity instances uniquely, as is required in case of super key, second, it should be minimum, that is, no proper subset of candidate key is a key.

Primary Key

A candidate key chosen by the database designer to act as key is the primary key. An entity type may have more than one candidate keys, in that case the database designer has to designate one of them as primary key, since there is always only a single primary key in an entity type. If there is just one candidate key then obviously the same will be declared as primary key.

Alternate Keys

Candidate keys which are not chosen as the primary key are known as alternate keys. if we select regNo as PK then the nIdNumber will be alternate key.

Secondary Key

Many times we need to access certain instances of an entity type using the value(s) of one or more attributes other than the PK. The difference in accessing instances using the value of a key or non-key attribute is that the search on the value of PK will always return a single instance (if it exists).

Lecture No. 09

Relationships

LONG

A relationship is any association, linkage, or connection between the entities of interest to the business; it is a two directional, significant association between two entities, or between an entity and itself. Each relationship has a name, an optionality (optional or mandatory), and a degree (how many). A relationship is described in real terms.

Relationship represents an association between two or more entities. An example of a relationship would be:

- Employees are assigned to projects
- Projects have subtasks
- Departments manage one or more projects

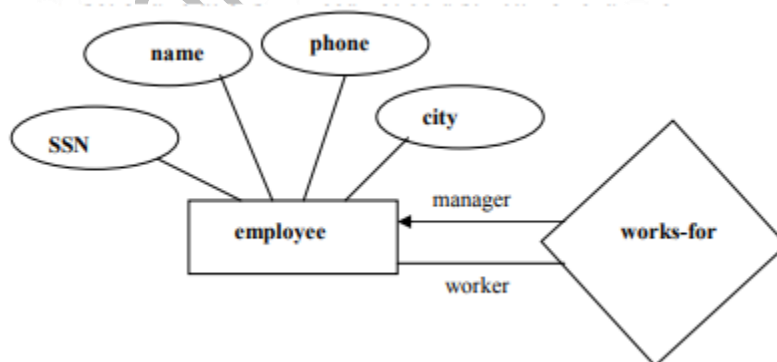
Relationship type

A relationship type is an abstraction of a relationship i.e. a set of relationships instances sharing common attributes. Entities enrolled in a relationship are called its participants.

The participation of an entity in a relationship is total when all entities of that set might be participant in the relationship otherwise it is partial.

Naming Relationships:

If there is no proper name of the association in the system then participants' names or abbreviations are used. STUDENT and CLASS have ENROLL relationship. However, it can also be named as STD_CLS.



The labels “manager” and “worker” are called “roles”. They specify how employee entities interact via the “works-for” relationship set.

Symbol for Relationships:

- Shown as a Diamond
- Diamond is doubled if one of the participant is dependent on the other
- Participants are connected by continuous lines, labeled to indicate cardinality.
- In partial relationships roles (if identifiable) are written on the line connecting the partially participating entity rectangle to the relationship diamond.
- Total participation is indicated by double lines

Types of Relationships

LONG

Unary Relationship

An ENTITY TYPE linked with itself, also called recursive relationship. Example Roommate, where STUDENT is linked with STUDENT.

Binary relationship

Binary relationship is the one that links two entities sets e.g. STUDENT-CLASS. Relationships can be formally described in an ordered pair form.

Ternary Relationship

A Ternary relationship is the one that involves three entities e.g. STUDENT-CLASS-FACULTY

N-ary Relationship

Most relationships in data model are binary or at most ternary but we could define a relationship set linking any number of entity sets i.e. n-ary relationship.

Relationship Cardinalities

LONG

The cardinality of a relationship is the number of entities to which another entity can map under that relationship. Symbols for maximum and minimum cardinalities are:

One-to-One mapping:

A mapping R from X to Y is one-to-one if each entity in X is associated with at most one entity in Y and vice versa.

Many-to-One mapping:

A mapping R from X to Y is many-to-one if each entity in X is associated with at most one entity in Y but each entity in Y is associated with many entities in X.

One-to-Many mapping:

A mapping R from X to Y is one-to-many if each entity in X is associated with many entities in Y but each entity in Y is associated with one entity in X.

Many-to-Many mapping:

A mapping R from X to Y is many-to-many if each entity from X is associated with many entities in Y and one entity in Y is associated with many entities in X.

Lecture No. 10

Roles in Relationships

Determine the role ETs play in a relationship. Most of the time is clear from the context, like in STD and COURSE relationship. Two situations when they should be expressed explicitly. A one to one relationship. Two ETs having more than one relationship.

Dependencies

Dependency is a type of constraint. There are a number of dependency types which are expressed below:

The Existence dependency:

This is the type of dependency which exists when one entity instance needs instance of another entity for its existence. Employees are dependent on projects, it means that if no project is assigned to an employee it cannot exist. In other words we can say that at a certain time an employee must be working on at least one project.

Identifier Dependency:

It means that the dependent entity in case of existence dependency does not have its own identifier and any external identifier is used to pick data for that entity.

Referential Dependency:

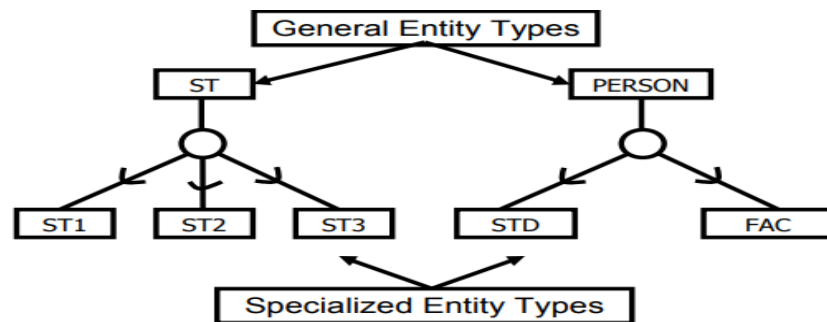
Referential integrity is the logical dependency of a foreign key on a primary key. The integrity of a row that contains a foreign key depends on the integrity of the row that it references—the row that contains the matching primary key.

Enhancements in E-R Data Model:

EER is a high-level data model that incorporates the extensions to the original ER model. Enhanced ERD are high level models that represent the requirements and complexities of complex database. Subclasses and Super classes. Specialization and Generalization.

Super-type and Subtypes

These are also relationships existing between entities, also referred to as generalized and specialized respectively.



A person supertype we can have further classify the person entity as Student (STD) and Teacher of Faculty member (FAC). Subtype entities are expressed with a link to the supertypes having an arc on the link

Lecture No. 11

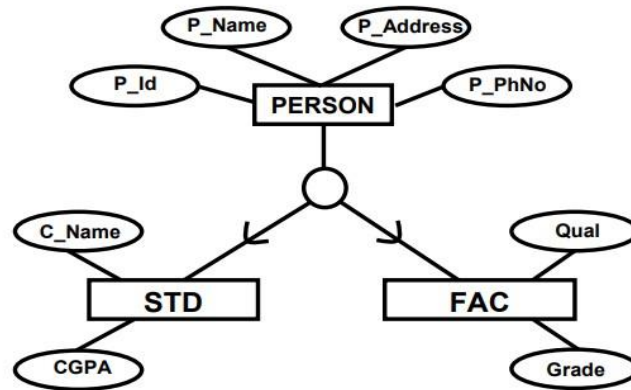
Inheritance

The transfer of the characteristics of a class in object-oriented programming to other classes derived from it. Inheritance in the paradigm of database systems we mean the transfer of properties of one entity to some derived entities, which have been derived from the same entities.

Super types and Subtypes

LONG

Subtypes hold all the properties of their corresponding super-types. Means all those subtypes which are connected to a specific supertype will have all the properties of their supertype.



Supertype / subtype Relationship:

The use of super type and subtype for the entities is very useful because it allows us to create hierarchy of the entities according to the attributes they have and we need not to write all the attributes again and again. We can group similar types of entities and the attributes associated with those entities at certain levels.

Specifying Constraints

Once there has been established a super/sub entity relationship there are a number of constraints which can be specified for this relationship for specifying further restrictions on the relationship.

Completeness Constraint

There is two types of completeness constraints, partial completeness constraints and total completeness constraints.

- Total Completeness:

Total Completeness constraint exist only if we have a super type and some subtypes associated with that supertype, and the following situation exists between the super type and subtype.
- Partial Completeness Constraint:

This type of completeness constraint exists when it is not necessary for any supertype entity to have its entire instance set to be associated with any of the subtype entity.

Disjointness Constraint

This rule or constraint defines the existence of a supertype entity in a subtype entity. There exist type types of disjoint rules.

- Disjointness rule
- Overlap rule

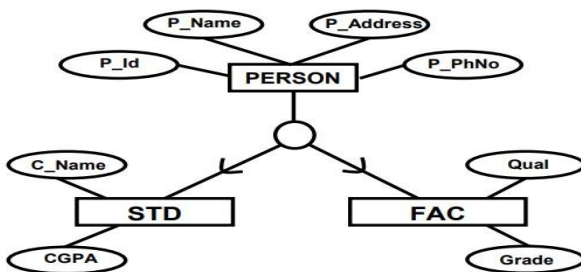
Disjointness rule

Describes the relationship between members of the subclasses and indicates whether it is possible for a member of a superclass to be a member of one, or more than one, subclass.

Overlap rule

The disjoint rule states an entity instance of a supertype can only be a member of one subtype. The overlap rule states an entity instance of a supertype can be a member of multiple subtypes. we can have four different types of combination existing for the supertype and subtype entities.

- Complete Disjoint
- Complete Overlapping
- Partial Disjoint
- Partial overlapping



Subtype Discriminator

This is a tool or a technique which provides us a methodology to determine that to which subtype one instance of a supertype belongs. To determine the relation we place an attribute in the entity supertype which can specify through its value, that to which entity subtype it belongs.

Lecture No. 12

Skip

Lecture No. 13

Identification of Entity Types of the Examination System

The exam controller section is the administrator of this whole examination system.
ANSWER:

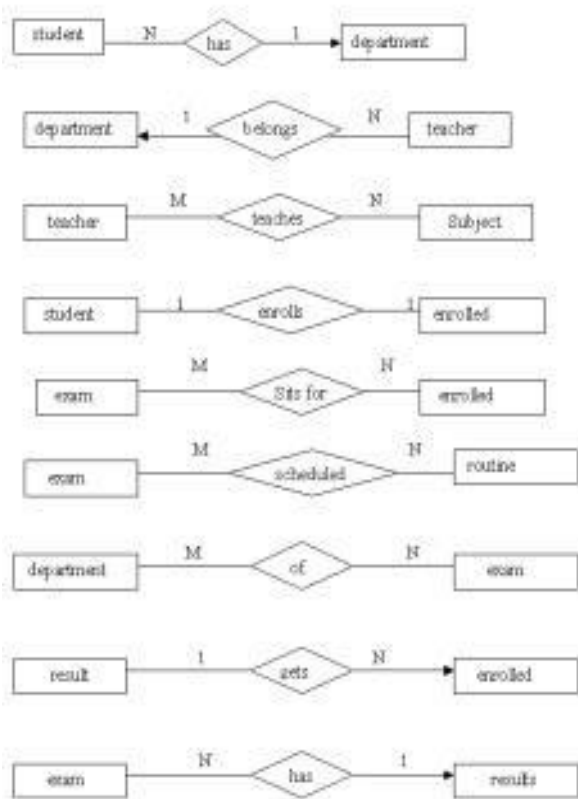
STEP 1: ENTITIES USED IN E-R DIAGRAM

1. Student
2. Department

3. Teacher
4. Subject
5. Exam
6. Exam Candidate
7. Result
8. Routine

STEP 2: FIND THE RELATIONSHIPS

1. This is the 1-to-many binary relationship set between department and student entity sets
2. Belongs to is the 1-to-many binary relationship set between DEPARTMENT and TEACHER sets.
Because- for a particular department there are many teacher but for a teacher only 1 department is fixed. And the participation of the teacher entity sets in this relationship sets is partial.
3. Teaches is the many-to-many binary relationship sets between teacher and subjects.
4. Enrolls is the one-to-one binary relationship sets between Student and Exam Candidate.
Component attributes are: date, month, and year.
5. Sits for is a many-to-many binary relationship sets between the Exam Candidate and the Examination.
6. Scheduled is the relationship that holds between the examination and the exam routine.
7. This is a many-to-many binary relationship set between the department and the examination.
8. This is the relationship sets between Exam Candidate and the Results.
9. Every exam has results. This is many to one relationship.



STEP 3:FIND THE KEY ATTRIBUTES

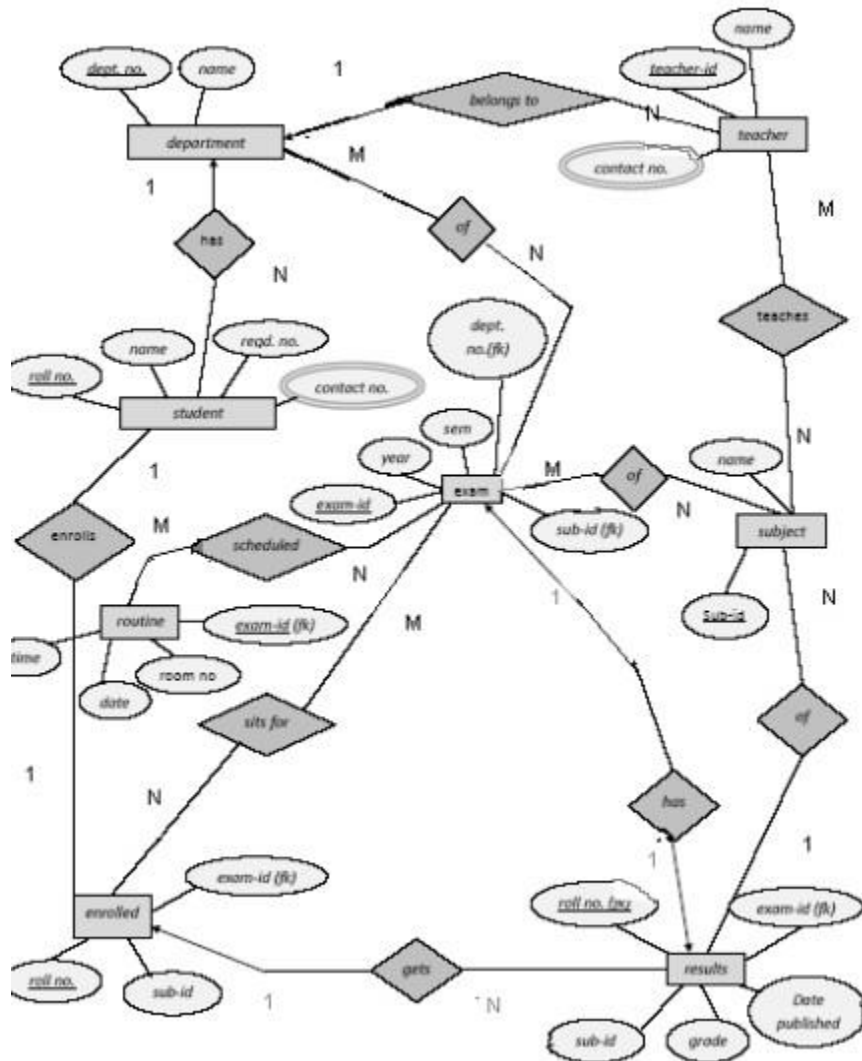
1. Student: rollno
2. Department: dept.no
3. Teacher: teacher_id
4. Subject: subject_id
5. Exam: exam_id
6. Exam Candidate: exam_id
7. Result: roll no
8. Routine: exam_id

STEP 4:FIND THE OTHER ATTRIBUTES

1. Student:
Name, Roll No, Reg. No, Contact No
2. Department:
Department_id, Name, Location
3. Teacher:
Teacher_id, Name, Contact No, Status

4. Subject:
Name, Subject_id ,Course, Dept
5. Exam:
Exam_id, Roll No, Sub_id, Exam roll no.
6. Exam Candidate:
Semester, Year, Exam_id, Sub_id
7. Result:
Roll no, Exam_id, Sub_id, Grade, Date_of_publish
8. Routine:
Date, Time, Room No, Exam_id
- 9.

STEP 5: COMPLETE THE ER-Diagrams



Lecture No. 14

Logical Database Design

Logical database design phase of database development process. Logical database design, like conceptual database design is our database design; it represents the structure of data that we need to store to fulfill the requirements of the users or organization for which we are developing the system. There are certain differences between the two that are presented.

Conceptual Database Design	Logical Database Design
Developed in a semantic data model (generally E-R data model)	In legacy data models (relational generally in current age)
Free of data model in which going to be implemented; many/any possible	Free of data model in which going to be implemented; many/any possible Results from Analysis Phase
Results from Analysis Phase	Obtained by translating the conceptual database design into another data model
Represented graphically	Descriptive
More expressive	Relatively less expressive
Going to be transformed and then implemented	Going to be implemented

Relational Data Model

The RDM is popular due to its two major strengths and they are:

- Simplicity
- Strong Mathematical Foundation

Presented by E. F. Codd in 1970, then of IBM before Relational Data Model, two older data models were in use; Hierarchical, Network First DBMS built on Relational Data Model (RDM) was system R. Another Relational DBMS built during those days was INGRES.

Basic of RDM

RDM used mainly for external, conceptual, and to some extent physical schema. Separation of conceptual and physical levels makes manipulation much easier, contrary to previous data models. RDM used for external, conceptual, and to some extent physical schema. The basic structure is relation both entities and relationships are modeled using tables/relations.

RDM used for external, conceptual, and to some extent physical schema. The basic structure is relation. Both entities and relationships are modeled using tables/relations.

Basic Properties of a Table

1. Each cell of a table contains atomic/single value
2. Each column has a distinct name; the name of the attribute it represents

3. The values of the attributes come from the same domain
4. The order of the columns is immaterial
5. The order of the rows is immaterial
6. Each row/tuple/record is distinct, no two rows can be same

A table

stId	stName	stAdres	doB
S001	Ali	Lahore	12/12/76
S002	A. Rehman	RWP	2/12/77

Components of RDM

As discussed in data model definition

1. Structure (relation/table)
2. Manipulation language (SQL)
3. Integrity constraints (Two)

Mathematical Relations

Consider two sets

$$A = \{x, y\} \quad B = \{2, 4, 6\}$$

Cartesian product of these sets

$$A \times B = \{(x,2), (x,4), (x,6), (y,2), (y,4), (y,6)\}$$

A relation is some subset of this Cartesian product, For example,

$$R1 = \{(x,2), (y,2), (x,6), (x,4)\}$$

$$R2 = \{(x,4), (y,6), (y,4)\}$$

The same notion of Cartesian product and relations can be applied to more than two sets, e.g. in case of three sets, we will have a relation of ordered triplets.

Database Relations

Thinking in some real world scenario

- Name = {Ali, Sana, Ahmed, Sara}

More Files Join VU WhatsApp Group: <https://chat.whatsapp.com/BL4ok4jnM9iJiglhAA1SG3>

➤ Age = {15,16,17,18,.....,25}

Cartesian product of Name & Age

Name X Age = {(Ali, 15), (Sana, 15), (Ahmed, 15), (Sara, 15)... (Ahmed, 25), (Sara, 25)}

CLASS = {(Ali, 18), (Sana, 17), (Ali, 20), (Ahmed, 19)}

Lecture No. 15

Skip

Lecture No. 16

Mapping Relationships

A Relationship Map is a picture of the relationships between individuals and organizations. Maps can help display the connections (or lack of connection) between important stakeholders in a community. These stakeholders can have power or influence over a community's ability to change.

Binary Relationships

Binary relationships are those, which are established between two entity types.

Following are the three types of cardinalities for binary relationships:

- One to One
- One to Many
- Many to Many

One to Many:

In this type of cardinality one instance of a relation or entity type is mapped with many instances of second entity type, and inversely one instance of second entity type is mapped with one instance of first entity type.

Many to Many Relationships:

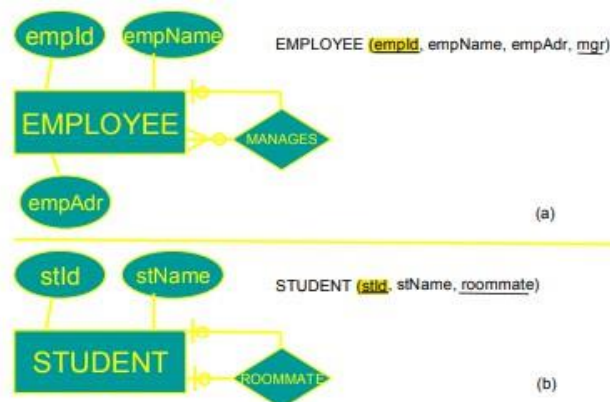
In this type of relationship one instance of first entity can be mapped with many instances of second entity. Similarly one instance of second entity can be mapped with many instances of first entity type.

One to One

This is a special form of one to many relationships, in which one instance of first entity type is mapped with one instance of second entity type and also the other way round. In this relationship primary key of one entity type has to be included on other as foreign key.

Unary Relationship

These are the relationships, which involve a single entity. These are also called recursive relationships. Unary relationships may have one to one, one to many and many to many cardinalities.



Supertype / Subtype Relationship:

A supertype is a generic entity type that has a relationship with one or more subtypes. A subtype is a sub-grouping of the entities in an entity type that is meaningful to the organization and that shares common attributes or relationships distinct from other subgroups.

Data Manipulation Languages

Data Manipulation Language or DML is a subset of operations used to insert, delete, and update data in a database. A DML is often a sublanguage of a more extensive language like SQL; DML comprises some of the operators in the language.

Following are the two types of languages:

- Procedural Languages

More Files Join VU WhatsApp Group: <https://chat.whatsapp.com/BL4ok4jnM9iJiglhAA1SG3>

➤ Non -Procedural Languages

Procedural Languages

These are called procedural languages (PL). In the case of a function or trigger procedure written in a procedural language, the database server has no built-in knowledge about how to interpret the function's source text. Instead, the task is passed to a special handler that knows the details of the language.

Non -Procedural Languages

A non-relational database is a database that does not use the tabular schema of rows and columns found in most traditional database systems. Instead, non-relational databases use a storage model that is optimized for the specific requirements of the type of data being stored.

Relational algebra

Relational Algebra is procedural query language, which takes Relation as input and generate relation as output. Relational algebra mainly provides theoretical foundation for relational databases and SQL. Projection is used to project required column data from a relation.

Five Basic Operators of Relational Algebra

Five basic operations in relational algebra:

- Selection
- Projection
- Cartesian product
- Union
- Set Difference.

Join Operation

Join is a special form of cross product of two tables. In Cartesian product we join a tuple of one table with the tuples of the second table. But in join there is a special requirement of relationship between tuples.

Lecture No. 18

Types of Joins

The joins operation forms a Cartesian product of its two arguments, performs a selection forcing equality on those attributes that appear in both relation schemas, and finally removes duplicate attributes. Following are the different types of joins:

- Theta Join
- Equi Join
- Semi Join
- Natural Join

Theta Join

Theta Join allows you to merge two tables based on the condition represented by theta. Theta joins work for all comparison operators. It is denoted by symbol θ . The general case of JOIN operation is called a Theta join. Syntax: $A \bowtie_{\theta} B$.

Equi Join

Equijoin($\bowtie$): Equijoin is a special case of conditional join where only equality condition holds between a pair of attributes. As values of two attributes will be equal in result of equijoin, only one attribute will be appeared in result.

Semi Join:

In semi join, first we take the natural join of two relations then we project the attributes of first table only. So after join and matching the common attribute of both relations only attributes of first relation are projected.

Natural Join

Natural join ($\bowtie$) is a binary operation that is written as $(r \bowtie s)$ where r and s are relations. The result of the natural join is the set of all combinations of tuples in r and s that are equal on their common attribute names.

Relational Algebra

The outer join merges the result of an inner join with the remaining tuples in one (or both) of the joined relations that do not share commonality with the tuples of the inner join result table.

Extension: the complement of the result of appl...

Inner_join: the result of applying the RESTRICT ...

Projection: the result of applying a PROJECT o...

U: the UNION ALL operator. Because UNION .

Lecture No. 19

Functional Dependency

Normalization is based on the concept of functional dependency. A functional dependency is a type of relationship between attributes. If A and B are attributes of a relation R, then B is functionally dependent on A if each value of A in R is associated with exactly one value of B; written as $A \longrightarrow B$.

It does not mean that A derives B, although it may be the case sometime. Means that if we know value of A then we can precisely determine a unique value of B.

Attribute of set of attributes on the left side are called determinant and on the right are called dependents

Like R (a, b, c, d, e)

$A \longrightarrow b, c, d, e$

$d \longrightarrow e$

Functional Dependencies and Keys:

The determinant of functional dependency that determines all attributes of that table is the super key. Super key is an attribute or a set of attributes that identifies an entity uniquely. In a table, a super key is any column or set of columns whose values can be used to distinguish one row from another.

Normal Forms

There are two goals of the normalization process: eliminate redundant data (for example, storing the same data in more than one table) and ensure data dependencies make sense (only storing related data in a table). Both of these are worthy goals as they reduce the amount of space a database consumes and ensure that data is logically stored.

Lecture No. 20

1st 2nd 3rd 4th 5th Normal Form

The goal of normalization is to create a set of relational tables that are free of redundant data and that can be consistently and correctly modified. This means that all tables in a relational database should be in the third normal form (3NF). A relational table is in 3NF if and only if all non-key columns are

- (a) mutually independent and
- (b) fully dependent upon the primary key

Mutual independence means that no non-key column is dependent upon any combination of the other columns. The first two normal forms are intermediate steps to achieve the goal of having all tables in 3NF. In order to better understand the 2NF and higher forms, it is necessary to understand the concepts of functional dependencies and loss less decomposition.

Lecture No. 21

Normalization Summary

Normalization is a step by step process to make DB design more efficient and accurate. A normalized database helps the DBA to maintain the consistency of the database. However, the normalization process is not a must, rather it is a strongly recommended activity performed after the logical DB design phase. Not a must means, that the consistency of the database can be maintained even with an un-normalized database design, however, it will make it difficult for the designer. Un-normalized relations are more prone to errors or inconsistencies.

Lecture No. 22

Data Volume and Usage Analysis

Statistics about the size and usage of data plays critical role in data processing efficiency. Final step of logical DB design or first step in physical DB design. Statistics collected during analysis phase from the users of the system May not be accurate.

DESIGNING FIELDS

Field is the smallest unit of application data recognized by system software, such as a programming language or any database management system. Data types are the structure defined for placing data in the attributes. Each data type is appropriate for use with certain type of data. 4 major objectives for using data types when specifying attributes in a database are given as under:

- Minimized usage of storage space
- Represent all possible values
- Improve data integrity
- Support all data manipulation

CODING AND COMPRESSION TECHNIQUES:

As the codes defined by the database administrator or the programmer consume less space so they are better for use in situations where we have large number of records and wastage of small amount of space in each record can lead to loss of huge amount of data storage space. Thus causing lowered database efficiency.

stId	stName	hobby
------	--------	-------

S1020	Sohail Dar	Reading
S1038	Shoaib Ali	Gardening
S1015	Tahira Ejaz	Reading
S1015	Tahira Ejaz	Movies
S1018	Arif Zia	Reading

Coding techniques are also useful for compression of data values appearing the data, by replacing those data values with the smaller sized codes we can further reduce the space needed by the data for storage in the database.