Data Base Management Systems (Introduction)

Data: Data is the raw material from which Useful Information is derived. Simply we can say that, data is a collection of unorganized facts, but can be made organized into useful information. Some examples of data are Prices, Weights, Costs, Number of items sold or purchased etc.

Information: The data, that have been processed in such a ways so as to increase the knowledge of the person who uses the data is known as information.

Data (or) Information Processing : The process of converting the facts into meaningful information is known as Data processing. It is also known as Information processing.

Meta Data: Simply we can say it as the Data about the Data. In other words, The Data that describes the properties of other data is known as Metadata. Meta Data keeps the information of the Other Data, i.e., it keeps the information HOW and WHERE the other data is stored.

Database: A database is a collection of data (information) of some given Organization (a company for example), that can be processed through one or more programs by multiple users. Some examples are A bank, A hospital, A company etc.....

Data Base Management Systems: A Data Base Management System (simply called as DBMS) is a System that manages the Data Base.

“A Software that manages the collection of Data"

A software that provides an efficient environment to help us in STORING and RETRIEVING the Data Base of an enterprise.

DBMS, Defines Data, Stores Data, Maintains data and of course, Deletes the Data.

Database is the CONTAINER for a collection of computerized data files. DBMS tells us n helps us in how to insert, maintain and delete these files.

Some advantages of Database Management Systems

- Reduction of Redundancies: Redundancy means duplicating (making the same copy of data again and again). Reduction of Redundancy means, avoiding the duplication of data (Remember the warning message "The file named XYZ.JPG is already exist. do you want to replace it?)
- Data Independence and Efficient Access: The files stored in Database are independent of their storage details. And the change in one file doesn’t effect the other (until and unless they related to each other).
- Data Integrity: Data Integrity means that the data values entered in the database must be checked to ensure that they fall within the correct format and range.

Let’s see an example. Suppose I am entering the student’s details of a School. What is I entered the age of the student as 55??? people will laugh at me. So the value should be less than 17. And of course, the Data Integrity also checks whether the referring field is existing or not. I mean, if i want to enter the marks of a person named shivani sharma, first of all it checks whether the name of that person exists in its database or not...

- Data Security
- Less Application Development Time: With the help of some Predefined functions like Concurrency control, Crash Recovery etc, DBMS helps us to develop applications in very less time.
- Conflict Resolution : It helps us in resolving the conflicts among various users to access the same data file
- Data Administration: DBMS provides maintenance and administration of data by providing a common base for the large collection of data being shared by several users.
- Concurrent Access: Number of users can access a single file Concurrently (I mean, at the same time).
- Crash Recovery: The DBMS maintains a continuous record for the changes made to the data, so , if there is any system crash by power failure or something, it can restore the Database...

The Transaction Management

Assume that you want to transfer some money (Say Rs. 1500) to your friend's account. You logged in into your bank's site and type your account number. Later you typed your friend's name and his account number. Later types the amount and clicks on TRANSFER button. Later 1500 /- TRANSFERS into your friend's account.

Now let’s see this process in DBMS's point of view.

You entered some money (Rs 1500). The DBMS checks whether the mentioned money is available in your account or not. Assume that you have some 3000 in your account. If available then it checks the recipient’s Account (Assume that he has Rs. 200 in his account).Later the DBMS reduces your amount to Rs 1500 and Adds that 1500 to your friend's account. And makes his account balance as Rs 1700 (200 + 1500).Then shows you a message, something like "TRANSACTION COMPLETED”. This is an example of Transaction.

A transaction is an execution of a user program and is seen by the DBMS as a series or list of actions.

“A transaction is nothing but a List of Actions". These actions include the reading and writing of database.

ACID Properties :

These are the properties that a transaction should possess in order to avoid failures during concurrent access to a database. The ACID is an acronym which stands for **A**tomicity, **C**onsistency, **I**solation, **and Durability**. Now let's have a look at these properties in detail. Of course, for beginners these all may look same and confusing, read those two or three times. Then you can get the exact meanings and differences among them.

- **Atomicity:** It ensures that the transaction either is executed **completely** or not at all. **I**ncomplete transaction consequences are not entertained, check an example
    - Assume that Shivani has Rs. 500/- in her account and Palvi has Rs. 200/- in her account. Now Shivani transfers an amount of Rs. 50/- to Palvi. A transaction debits the amount from shivani's account, but before it could be credited to palvi, if there is a failure, then transaction would stop. So finally Shivani loses Rs. 5o but palvi can't get the amount. This leaves the data in an inconsistent state. If there is a failure during transaction execution , then measures must be taken to get back the data in a form which was in, before transaction (I mean, the 50 shouldn't be deducted from Shivani's account in our case). This is taken care of by transaction management component.
- **Consistency:** The data in the database must always be in a consistent state. A transaction occurred on a Consistent data should end with the data with another Consistent stage after completion of that transaction. Take the above case, the total of the amounts of Shivani and Palvi are (500+200) is Rs 700/- So, after the Transaction completed, the total amount should be same. i.e., (450+250 = 700). Of course, in intermediate stage, where the amount is deducted from Shivani's account but not yet credited to Palvi, the total would not be same. It is the responsibility of DBMS.
- Durability: Durability ensures that the data remains in a consistent state even after the FAILURE. (This is ensured by keeping copy of the old data in the Disk, till the transaction is COMPLETED). I mean, if shivani is transferring money to Palvi. The money is deducted from Shivani's account and power gone (before adding the MONEY to palvi's account). Then our DBMS shouldn't save that transaction. This is called Durability.
- Isolation : All transaction must run in Isolation from one another. I mean each and every transaction should be kept unaware of other transactions and execute independently. The intermediate results shouldn't be available to other transactions.

Data Base Users and Administrator -

Depending on their degree of expertise or the mode of their interactions with the DBMS, The Data Base users (people who uses database) can be classified into several groups. They are,

- Naive Users
- Online Users
- Application Users
- Sophisticated Users
- Specialized Users

Naive Users: Naive means Lacking Experience; these are the users who need not be aware of the presence of the Data Base System. Example of these type of users is The user of an ATM machine. Because these users only responds to the instructions displayed on the screen (enter your pin number, click here, enter the required money etc). Obviously operations performed by these users are very limited.

Online Users: These are the users who may communicate with the Data Base directly via an online terminal or indirectly via a user interface and application program. These users are aware of the presence of the Data Base System and may have acquired a certain amount of expertise within the limited interaction they are permitted with a Data Base.

Application Programmers: Professional / Application programmers are those who are responsible for developing application programs or user interface. The application programs could be written in a general-purpose programming language or the commands available to manipulate a database.

Sophisticated Users: Simply we can say that these are the EXPERIENCED users. These people interact with the system without writing programs. Instead they from their requests in a database query language. They submit each such query to a query processor, whose function is to break down DML (Data Manipulation Language, the language which is used to MAINTAIN the data. we shall discuss about this later) statements into instructions that the storage manager understands. Analysts who submit queries to explore data in the Data Base fall in this category.

Specialized Users: These are the sophisticated users who write specialized database applications that do not fit into the traditional data-processing framework. Among these applications are computer - aided design systems, knowledge-based and expert systems, systems that store data with complex data types (Ex, Graphics Data and Audio Data) and environment-modeling systems.

DBA (Data Base Administrator): The person who controls both Data and the Programs that access that data in the Data Base is called the Data Base Administrator (DBA).

Functions of the DBA are,

- Defining Schemas (arrangement of Data)
- Creating Storage Structure and Access Methods (I mean how to store data and access that)
- Modifying the storage Data
- Granting Authorization Permissions
- Specifying the CONDITIONS of the data storage
- Periodically Updating the Data Base etc.,

Data Models

Data Model means, to give a SHAPE to the data. A Data Model Makes it easier to understand the Data. We can define the data model as "The Collection of High-Level data description that hide many low level storage details".

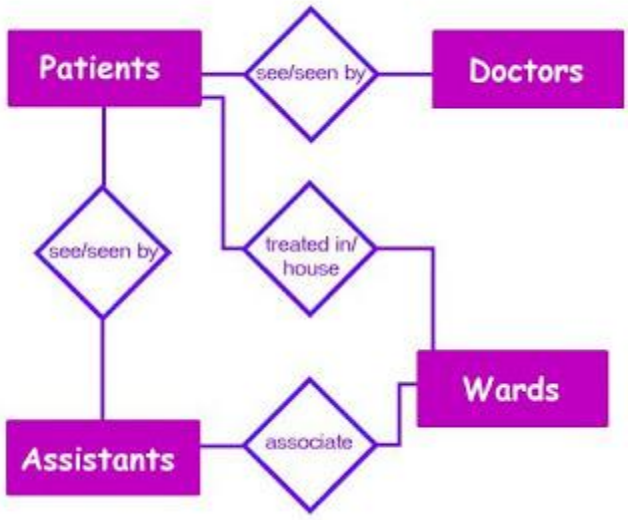
The Data Models are divided into THREE different groups. They are

1. Object - Based Logical Models
2. Record - Based Logical Models
3. Physical Data Models

1. Object Based Logical Models: These are used in describing data at Logical Level and View Level. (Logical level describes what data are stored in the database and what relationships exists among those Data. This logical level is used by the DBA (Data Base Administrator). He must decide what information is to be kept in the Data Base. The View level describes Only part of the entire database to be viewed by the user of the database hiding the details of the Information Stored.
The Object based logical models are described in the different following models.

- The E-R (Entity-Relationship) Model
- The Object-Based Logical Model
- The Semantic Data Model
- The Functional Data Model

E-R Model: The entity is a "Thing" or “Object" in the real world that is distinguishable from other objects. The E-R model is based on the collection of basic objects called Entities and the Relationship among them. Consider the following Diagram.



In the above diagram, RECTANGLES represents ENTITIES, DIAMONDS represents RELATIONSHIP among those ENTITIES. LINES represents links of Entities to Relationships.

Object - Oriented Model: The Object Oriented model based on a collection of OBJECTS. An object contains values stored in Instance Variables and Bodies of Code that operates on the Object.(These bodies of Code are called Methods).

Objects that contain the same types of values and the same methods are grouped together into classes. (A class is the definition of the object).

Semantic Data Model: A Semantic data model is a more high level data model that makes it easier for a user to give Starting Description of the data in an organization. (Semantic is nothing but the Meaning). These models contain a wide verity of relations that helps to describe a real application scenario. A DBMS cannot support all these relations directly. So it is build only with few relations known as relational model in DBMS. A widely used semantic data model is the Entity-Relationship (ER) data model which allows us to graphically denote entities and relationship between them.

Functional Data Model: The functional data model makes it easier to define functions and call them where ever necessary to process data.

2. Record - Based Logical Models: In this type of models, the data is kept in the form of RECORDS (documents). Thse models describe data at Logical and View Levels. When compared with Object Based Data Models, the record based logical models specify the overall logical structure of the database and provides higher-level implementation.

These are of 3 types. Those are,

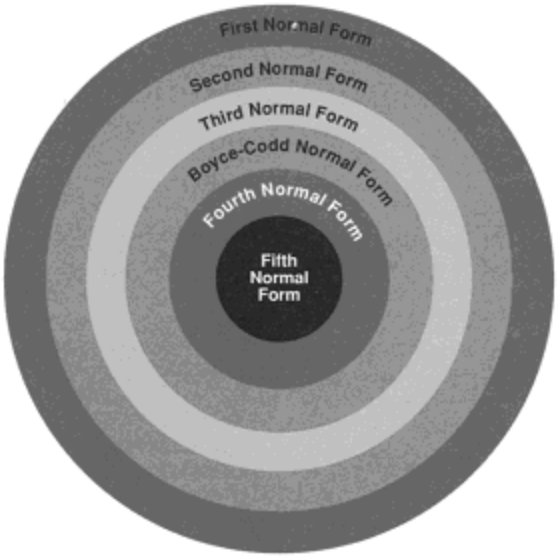
- Relational Model
- Network Model
- Hierarchical Model

Relational Model: The relational model represents both Data (entities) and Relationships among that data in the form of Tables. Each table contains multiple columns and each column contains a unique name. Have a look at the following table.

Normalization and Normal Forms -

The Process of making a table (or often we call it as Record) NORMAL is called Normalization We can say that the Normalization is the process of simplifying the Table.

The design method which helps us to MINIMIZE the redundancies of data (repetitions) and reduces the errors / mistakes in the table is called Normalization.



Note: To perform this Normalization Process, we should analyze the INTER DEPENDENCIES of the attributes in the tables and should take the subsets of the larger tables to form the Smaller tables.
In this process, we apply various Normal Forms to the given Database (Table). These Normal Forms helps us to break the BIG tables into smaller ones. Now lets have a look at these Normal Forms...

1st Normal Form :

* In 1st Normal Form we deal with the REDUNDANCY of ROWS.

In first Normal Form, each ATTRIBUTE must be ATOMIC. There shouldn't be any repetitions in the Rows. and There shouldn't be any multivalued columns. The advantage of the 1st Normal Form is "The queries become easier"

The 1NF eliminates GROUPS by keeping them into different (separate) Tables and connecting them with One to Many relationship.

2nd Normal Form:

* A table or entity should be in the first normal form, if you want to apply the 2nd Normal form on that.

Note: The first normal form deals with the repetition (redundancy) of data in Rows (Horizontal) , where as the 2nd Normal form deals with the redundancy of the data in Columns (Vertical).

A table should be in the First Normal form if you want to make sure that it is in 2nd normal form. Some more additional conditions are,

- It shouldn't have a COMPOSITE primary key. (I mean, the key SHOULD NOT be divided into sub parts (logical entities).
- In 2nd Normal Form, each attribute should FUNCTIONALLY Dependent on the primary key.
    - What is this Functionally Dependence?
        - If one or more attributes determines UNIQUELY the values of other attributes, then that property is called as Functionally Dependence.
- If you find any Non Dependent Attributes, you should move them into Smaller Tables.
- 2NF helps us to improve the data Integrity (Error free)

3rd Normal Form:

* A table or entity should be in the Second Normal Form (so obviously, it should be in the first normal form too :P) if you want to apply 3rd Normal form on that. Some more conditions are,

- If the table is in 3rd Normal Form, then it should be Transitive dependencies free.....
    - What is this Transitive Dependence?
        - If there exists TWO separate entities in ONE table. Obviously we should move them into smaller tables.
- 3nf also further improves the integrity of the data.

Boyce - Codd Normal Form:

- It is the ADVANCED version of the 3rd Normal Form (3NF).
- A row in a table is in BCNF, if every determinant of it is a Candidate Key. (Most of the entities of 3rd Normal form are already in Boyce - Codd Normal Form)
    - Why should we use BCNF if already 3NF is there?
        - The 3rd Normal Form misses the inter dependencies between Non Key attributes.
        - Simple we can say that, a 3NF is not in BCNF if,
            - There are number of Candidate Keys
            - The keys are composed of multiple attributes
            - There are more than one attribute between the Keys.....

4th Normal Form :

*Obviously it should be in the 3NF if you want to call it 4NF. In addition, we should eliminate Trivial Multivalued Dependencies... I mean, our table shouldn't have multiple sets of Multi Valued Dependencies.

5th Normal Form:

* It should be in the 4th Normal Form...

- In this, we should eliminate the dependencies, not determined by the Keys.
- I mean, every dependency should be the consequence of its Candidate keys...

Student

MatricNo	Name	Registered	CounsellorNo
s01	Bloggs	1993	4523
s02	Smith	1998	3412
s05	Jones	1997	4523
s07	Stewart	1996	4538
s09	MacDonald	1995	4523

Enrolment

MatricNo	CourseCode	TutorNo
s01	c4	4523
s05	c2	3412
s05	c7	3412
s07	c4	4538
s09	c4	4523
s09	c2	4538
s09	c7	4523

Course

CourseCode	Title	Credit
c2	C++ Programming	100
c4	Databases	100
c7	Logic	50

Network Model: Data in the network model are represented by collection of Records and Relationships among data are connected by LINKS. These links can be viewed as Pointers. Have a look at the following diagram.

Hierarchical Model : This is also same as Hierarchical model, the difference is the records in the database are represented in the form of TREES (in Hierarchical way)

Physical Data Models: Physical data models are used to describe data at the lowest level, which explains how the data is actually stored using complex low-level data structures. Actually, the physical data models are rarely used.There are two types in Physical Data Models.

- Unifying Model
- Frame-Memory Model

2.3 KEYS

- An important constraint on the entities of an entity type is the key or uniqueness constraint on attributes.
- A key is an attribute (also known as column or field) or a combination of attribute that is used to identify records.
- Sometimes we might have to retrieve data from more than one table, in those cases we require to join tables with the help of keys.
- The purpose of the key is to bind data together across tables without repeating all of the data in every table
- Such an attribute is called a key attribute, and its values can be used to identify each entity uniquely.
- For example, the Name attribute is a key of the COMPANY entity type because no two companies are allowed to have the same name.
- For the PERSON entity type, a typical key attribute is SocialSecurityNumber.
- Sometimes, several attributes together form a key, meaning that the combination of the attribute values must be distinct for each entity.
- If a set of attributes possesses this property, we can define a composite attribute that becomes a key attribute of the entity type.

The various types of key with e.g. in SQL are mentioned below, (For examples let suppose we have an Employee Table with attributes 'ID' , 'Name' ,'Address' , 'Department_ID' ,'Salary')

(I) Super Key – An attribute or a combination of attribute that is used to identify the records uniquely is known as Super Key. A table can have many Super Keys.

E.g. of Super Key

1 ID
2 ID, Name
3 ID, Address
4 ID, Department_ID
5 ID, Salary
6 Name, Address
7 Name, Address, Department_ID …… So on as any combination which can identify the records uniquely will be a Super Key.

(II) Candidate Key – It can be defined as minimal Super Key or irreducible Super Key. In other words an attribute or a combination of attribute that identifies the record uniquely but none of its proper subsets can identify the records uniquely.

E.g. of Candidate Key

1 Code
2 Name, Address

For above table we have only two Candidate Keys (i.e. Irreducible Super Key) used to identify the records from the table uniquely. Code Key can identify the record uniquely and similarly combination of Name and Address can identify the record uniquely, but

neither Name nor Address can be used to identify the records uniquely as it might be possible that we have two employees with similar name or two employees from the same house.

(III) Primary Key – A Candidate Key that is used by the database designer for unique identification of each row in a table is known as Primary Key. A Primary Key can consist of one or more attributes of a table.

E.g. of Primary Key - Database designer can use one of the Candidate Key as a Primary Key. In this case we have “Code” and “Name, Address” as Candidate Key, we will consider “Code” Key as a Primary Key as the other key is the combination of more than one attribute.

(IV) Foreign Key – A foreign key is an attribute or combination of attribute in one base table that points to the candidate key (generally it is the primary key) of another table. The purpose of the foreign key is to ensure referential integrity of the data i.e. only values that are supposed to appear in the database are permitted.

E.g. of Foreign Key – *Let consider we have another table i.e. Department Table with Attributes “Department_ID”, “Department_Name”, “Manager_ID”, ”Location_ID” with Department_ID as an Primary Key. Now the Department_ID attribute of Employee Table (dependent or child table) can be defined as the Foreign Key as it can reference to the Department_ID attribute of the Departments table (the referenced or parent table), a Foreign Key value must match an existing value in the parent table or be NULL.*

(V) Composite Key – If we use multiple attributes to create a Primary Key then that Primary Key is called Composite Key (also called a Compound Key or Concatenated Key).

E.g. of Composite Key, if we have used “Name, Address” as a Primary Key then it will be our Composite Key.

(VI) Alternate Key – Alternate Key can be any of the Candidate Keys except for the Primary Key.

E.g. of Alternate Key is “Name, Address” as it is the only other Candidate Key which is not a Primary Key.

(VII) Secondary Key – The attributes that are not even the Super Key but can be still used for identification of records (not unique) are known as Secondary Key.

E.g. of Secondary Key can be Name, Address, Salary, Department_ID etc. as they can identify the records but they might not be unique.

2.4 RELATION

- There are several implicit relationships among the various entity types.
- In fact, whenever an attribute of one entity type refers to another entity type, some relationship exists.
- For example, the attribute Manager of department refers to an employee who manages the department. In the ER model, these references should not be represented as

There are three types of relationships
1) One to one
2) One to many
3) Many to many

1 One to one: An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A.

2(A) One to many: An entity in A is associated with any number (zero or more) of entities in B. An entity in B, however, can be associated with at most one entity in A.

2(B) Many to one: An entity in A is associated with at most one entity in B. An entity in B, however, can be associated with any number (zero or more) of entities in A.

3. Many to many: An entity in A is associated with any number (zero or more) of entities in B, and an entity in B is associated with any number (zero or more) of entities in A.

relationships or **relation**. There is a relation “borrower” in the entities customer and account which can be shown as follows:

Specialization:

- An entity set may include sub groupings of entities that are distinct in some way from other entities in the set.
- For instance, a subset of entities within an entity set may have attributes that are not shared by all the entities in the entity set. The E-R model provides a means for representing these distinctive entity groupings.
  Consider an entity set person, with attributes name, street, and city. A person may be further classified as one of the following:
  • Customer
  • Employee
- Each of these person types is described by a set of attributes that includes all the attributes of entity set person plus possibly additional attributes.

- For example, customer entities may be described further by the attribute customer-id, whereas employee entities may be described further by the attributes employee-id and salary.
- The process of designating sub groupings within an entity set is called specialization.
- The specialization of person allows us to distinguish among persons according to whether they are employees or customers.
- As another example, suppose the bank wishes to divide
- accounts into two categories, checking account and savings account. Savings accounts need a minimum balance, but the bank may set interest rates differently for different customers, offering better rates to favored customers.
- Checking accounts have a fixed interest rate, but offer an overdraft facility; the overdraft amount on a checking account must be recorded.
- The bank could then create two specializations of account, namely savings-account and checking-account.
- As we saw earlier, account entities are described by the attributes account-number and balance.
- The entity set savings-account would have all the attributes of account and an additional attribute interest-rate.
- The entity set checking-account would have all the attributes of account, and an additional attribute overdraft-amount.
- We can apply specialization repeatedly to refine a design scheme. For instance, bank employees may be further classified as one of the following:
  • Officer
  • Teller
  • Secretary

- Each of these employee types is described by a set of attributes that includes all the attributes of entity set employee plus additional attributes. For example, officer entities may be described further by the attribute office-number, teller entities by the attributes station-number and hours-per-week, and secretary entities by the attribute hours-per-week. Further, secretary entities may participate in a relationship secretary-for, which identifies which employees are assisted by a secretary.
- An entity set may be specialized by more than one distinguishing feature. In our example, the distinguishing feature among employee entities is the job the employee performs. Another, coexistent, specialization could be based on whether the person is a temporary (limited-term) employee or a permanent employee, resulting in the entity sets temporary employee and permanent-employee. When more than one specialization is formed on an entity set, a particular entity may belong to multiple specializations. For instance, a given employee may be a temporary employee who is a secretary.
- In terms of an E-R diagram, specialization is depicted by a triangle component labeled ISA. The label ISA stands for “is a” and represents, for example, that a customer “is a” person. The ISA relationship may also be referred to as a super

class- subclass relationship. Higher- and lower-level entity sets are depicted as regular entity sets—that is, as rectangles containing the name of the entity set.

Generalization:

- The refinement from an initial entity set into successive levels of entity sub groupings represents a top-down design process in which distinctions are made explicit. The design process may also proceed in a bottom-up manner, in which multiple entity sets are synthesized into a higher-level entity set on the basis of common features. The database designer may have first identified a customer entity set with the attributes name, street, city, and customer-id, and an employee entity set with the attributes name, street, city, employee-id, and salary. There are similarities between the customer entity set and the employee entity set in the sense that they have several attributes in common. This commonality can be expressed by generalization, which is a containment relationship that exists between a higher-level entity set and one or more lower-level entity sets. In our example, person is the higher-level entity set and customer and employee are lower-level entity sets.
- Higher- and lower-level entity sets also may be designated by the terms super class and subclass, respectively. The person entity set is the super class of the customer and employee subclasses.
- For all practical purposes, generalization is a simple inversion of specialization. We will apply both processes, in combination, in the course of designing the E-R schema for an enterprise. In terms of the E-R diagram itself, we do not distinguish between specialization and generalization. New levels of entity representation will be distinguished (specialization) or synthesized (generalization) as the design schema comes to express fully the database application and the user requirements of the database.

Aggregation:

- One limitation of the E-R model is that it cannot express relationships among relationships.
- To illustrate the need for such a construct, consider the ternary relationship works-on, which we saw earlier, between a employee, branch,and job.
- Now, suppose we want to record managers for tasks performed by an employee at a branch; that is, we want to record managers for (employee, branch, job)combinations. Let us assume that there is an entity set manager.
- One alternative for representing this relationship is to create a quaternary relationship manages between employee, branch, job, and manager.

1. What is SQL?Answer: Structured Query Language (SQL) is a language that provides an interface to relational database systems.
   In common usage SQL also encompasses DML (Data Manipulation Language), for INSERTs, UPDATEs, DELETEs and DDL (Data Definition Language), used for creating and modifying tables and other database structures.
2. What is SELECT statement?Answer: The SELECT statement lets you select a set of values from a table in a database.The values selected from the database table would depend on the various conditions that are specified in the SQL query.
3. How can you compare a part of the name rather than the entire name?Answer: SELECT * FROM people WHERE empname LIKE '%ab%'Would return a recordset with records consisting empname the sequence 'ab' in empname.
4. What is the INSERT statement?Answer: The INSERT statement lets you insert information into a database.
5. How do you delete a record from a database?Answer: Use the DELETE statement to remove records or any particular column values from a database.

6. How could I get distinct entries from a table?Answer: The SELECT statement in conjunction with DISTINCT lets you select a set of distinct values from a table in a database.The values selected from the database table would of course depend on the various conditions that are specified in the SQL query.ExampleSELECT DISTINCT empname FROM emptable
7. How to get the results of a Query sorted in any order?Answer: You can sort the results and return the sorted results to your program by using ORDER BY keyword thus saving you the pain of carrying out the sorting yourself.The ORDER BY keyword is used for sorting.SELECT empname, age, city FROM emptable ORDER BY empname
8. How can I find the total number of records in a table?Answer: You could use the COUNT keyword , exampleSELECT COUNT(*) FROM emp WHERE age>40
9. What is GROUP BY?Answer: The GROUP BY keywords have been added to SQL because aggregate functions (like SUM) return the aggregate of all column values every time they are called.Without the GROUP BY functionality, finding the sum for each individual group of column values was not possible.
10. What is the difference among "dropping a table", "truncating a table" and "deleting all records" from a table.Answer: Dropping : (Table structure + Data are deleted), Invalidates the dependent objects ,Drops the indexes Truncating: (Data alone deleted), Performs an automatic commit, Faster than deleteDelete : (Data alone deleted), Doesn't perform automatic commit

What are the Large object types suported by Oracle?Answer: Blob and Clob.

11. Difference between a "where" clause and a "having" clause.Answer: Having clause is used only with group functions whereas Where is not used with.
12. What's the difference between a primary key and a unique key?Answer: Both primary key and unique enforce uniqueness of the column on which they are defined.But by default primary key

creates a clustered index on the column, where are unique creates a nonclustered index by default.Another major difference is that, primary key doesn't allow NULLs, but unique key allows one NULL only.

13. What are cursors? Explain different types of cursors.What are the disadvantages of cursors? How can you avoid cursors?Answer: Cursors allow row-by-row prcessing of the resultsets.Types of cursors: Static, Dynamic, Forward-only, Keyset-driven.See books online for more information.Disadvantages of cursors: Each time you fetch a row from the cursor, it results in a network roundtrip, where as a normal SELECT query makes only one rowundtrip, however large the resultset is.Cursors are also costly because they require more resources and temporary storage (results in more IO operations).Furthere, there are restrictions on the SELECT statements that can be used with some types of cursors.Most of the times, set based operations can be used instead of cursors.
14. What are triggers? How to invoke a trigger on demand?Answer: Triggers are special kind of stored procedures that get executed automatically when an INSERT, UPDATE or DELETE operation takes place on a table.Triggers can't be invoked on demand.They get triggered only when an associated action (INSERT, UPDATE, DELETE) happens on the table on which they are defined.Triggers are generally used to implement business rules, auditing.Triggers can also be used to extend the referential integrity checks, but wherever possible, use constraints for this purpose, instead of triggers, as constraints are much faster.
15. What is a join and explain different types of joins.Answer: Joins are used in queries to explain how different tables are related.Joins also let you select data from a table depending upon data from another table.Types of joins: INNER JOINs, OUTER JOINs, CROSS JOINs.OUTER JOINs are further classified as LEFT OUTER JOINS, RIGHT OUTER JOINS and FULL OUTER JOINS.
16. What is a self join?Answer: Self join is just like any other join, except that two instances of the same table will be joined in the query.
17. How can I eliminate duplicates values in a table?Answer: SQL> DELETE FROM table_name A WHERE ROWID > (SELECT min(rowid) FROM table_name B WHERE A.key_values = B.key_values);
18. How can one examine the exact content of a database column?Answer: SELECT DUMP(col1) FROM tab1
    WHERE cond1 = val1;
    DUMP(COL1)
    Typ=96 Len=4: 65,66,67,32
    For this example the type is 96, indicating CHAR, and the last byte in the column is 32, which is the ASCII code for a space.This tells us that this column is blank-padded.
19. How can I change my Oracle password?Answer: From SQL*Plus type: ALTER USER username IDENTIFIED BY new_password

21. What are the most important DDL statements in SQL?Answer: The most important DDL statements in SQL are:

- create table-creates a new database table
- alter table-alters (changes) a database table
- drop table-deletes a database table
- create index-creates an index (search key)
- drop index-deletes an index

22. What are the different SELECT statements?Answer: The SELECT statements are:

- SELECT column_name(s) FROM table_nameSELECT DISTINCT column_name(s) FROM table_name
- SELECT column FROM table WHERE column operator value
- SELECT column FROM table WHERE column LIKE pattern
- SELECT column,SUM(column) FROM table GROUP BY column
- SELECT column,SUM(column) FROM table GROUP BY column HAVING SUM(column) condition value

23. What are the INSERT INTO Statements?Answer: The INSERT INTO statements are:

- INSERT INTO table_name VALUES (value1, value2,....)
- INSERT INTO table_name (column1, column2,...) VALUES (value1, value2,....)

24. Write theUpdate Statement?Answer: UPDATE table_name SET column_name = new_value WHERE column_name = some_value

25. What are the Delete Statements?Answer: These are:

- DELETE FROM table_name WHERE column_name = some_value

Delete All Rows:

- DELETE FROM table_name
- DELETE * FROM table_name

26. how we can sort the Rows?Answer: There are mainly three types:

- SELECT column1, column2,...FROM table_name ORDER BY columnX, columnY,..
- SELECT column1, column2,...FROM table_name ORDER BY columnX DESC
- SELECT column1, column2,...FROM table_name ORDER BY columnX DESC, columnY ASC

27. Explain IN operator.Answer: The IN operator may be used if you know the exact value you want to return for at least one of the columns.
SELECT column_name FROM table_name WHERE column_name IN (value1,value2,..)

28. Explain BETWEEN...ANDAnswer: SELECT column_name FROM table_name WHERE column_name BETWEEN value1 AND value2 The values can be numbers, text, or dates.

1. What Are Networks Used For?

A simplified but worthwhile description of the uses of computer networks might be as follows:

- Sharing of hardware: For example, several PCs might be networked together in a wired or wireless local area network (LAN) to share a printer.
- Sharing of information: Distributed databases, e-mail, the World Wide Web and so on are examples of this. Here the sharing involves both LANs and wide area networks (WANs), especially the latter.

2. Overview of the Layers

The layers collectively are often referred to as the protocol stack.

a. Physical Layer

- This is concerned with the nature of the physical media (metal or optical cable, free-space microwave, etc.) used to send signals, the nature of the signals themselves, and so on.
- There is also the question of signal form; the signals themselves may be in the form of pure 0-1 bits, or may be in the form of certain frequencies. In addition there are questions concerning how a receiver distinguishes two bits which are adjacent in time.
- A major issue is the form of the medium, both in terms of the materials it uses and its topology. A basic wired Ethernet, for example, consists of cable conducting electrical signals; the connections could also be wireless. More complicated networks, including Ethernets, may consist of more than one cable, with all of them connected via a hub. The latter has become common even at the household level.

b. Data Link Layer

- For example, in an Ethernet, this layer is concerned with ensuring that two network stations connected to the same cable do not try to access the line at the same time. For this reason the Ethernet operation is an example of what is called a Medium Access Control (MAC) Protocol.
- Here is an overview of how the Ethernet MAC protocol works, using a "listen before talk" approach. When a network node has a message ready to send, it first senses the cable to see if any node is currently sending. If so, it generates a random backoff time, waiting this amount of time before trying again. If the node does not "hear" any other node sending, it will go ahead and send.
- There is a small chance that another node actually had been sending but due to signal propagation delay the transmission had not yet reached the first node. In that case a collision will occur, destroying both messages. Both nodes will sense the collision, and again wait random amounts of time before trying again.
- This layer also does the setting up of frames of bits (i.e. sets of consecutive bits sent along the wire), which not only include the message itself but also information such as (say, in the Ethernet case) the Ethernet ID number of the destination machine.
- Messages may be broken up into pieces before being sent. This may be handled at the transport level (see below), but may also be done at the data link level.

c. Network Layer

- This is the routing layer. Questions addressed in this layer include: If in our example above saturn wants to send a message to holstein, how is that accomplished? Obviously its first step is to send the message to mars; how does saturn know this? How can alternate routes be found if traffic congestion occurs?

d. Transport Layer

- Suppose saturn's message to holstein consists of a large file transfer, say 100 megabytes. This transfer will take a long time (by network standards), and we certainly don't want it to monopolize the network during that time. We also must deal with the fact that the buffer space at holstein won't be large enough to deal with a 100-megabyte message. Also, one 100-megabyte message would have a sizable probability of having at least one bit in error, and if so, we would have to retransmit the entire message!
- So, the file transfer must be done in pieces. But we don't want to burden the user at saturn with the task of breaking up the 100 megabytes into pieces, nor do we want to burden the user at holstein with the reassembly of the messages. Instead, the network software (again, typically in the OS) should provide these services, which it does at the transport layer, as for example is the case with TCP.

e. Session Layer

- This layer is concerned with management of a session, i.e. the duration of a connection between two network nodes. The word connection here does not mean something physical, but rather refers to an agreement between two nodes that some chunks of data with some relation to each other will be exchanged for some time. Actually, TCP does this in some senses, as does the socket interface to TCP, which is very much like the interfaces for reading or writing a file (described in more detail later).

f. Presentation Layer

- This layer deals with such matters as translating between character codes, if the source uses one and the destination the other. In the old days, this could mean ASCII at one end and EBCDIC on the other end. Today, though, it could mean for example two different coding systems for Chinese characters, Big 5 and GB.

g. Application Layer

- You can write programs at the application layer yourself, and of course you use many programs written by others, such as ftp, Web browsers, e-mail utilities, and so on.

3. Network Security

- System and network technology is a key technology for a wide variety of applications. Security is crucial to networks and applications. Although, network security is a critical requirement in emerging networks, there is a significant lack of security methods that can be easily implemented.
- There exists a "communication gap" between the developers of security technology and developers of networks. Network design is a well-developed process that is based on the Open Systems Interface (OSI) model. The OSI model has several advantages when designing networks.

- It offers modularity, flexibility, ease-of-use, and standardization of protocols. The protocols of different layers can be easily combined to create stacks which allow modular development. The implementation of individual layers can be changed later without making other adjustments, allowing flexibility in development. In contrast to network design, secure network design is not a well-developed process.
- When considering network security, it must be emphasized that the whole network is secure. Network security does not only concern the security in the computers at each end of the communication chain. When transmitting data the communication channel should not be vulnerable to attack.
- A possible hacker could target the communication channel, obtain the data, decrypt it and re-insert a false message. Securing the network is just as important as securing the computers and encrypting the message.

4. Internet Attack Methods

- Internet attacks methods are broken down into categories. Some attacks gain system knowledge or personal information, such as eavesdropping and phishing. Attacks can also interfere with the system's intended function, such as viruses, worms and trojans. The other form of attack is when the system's resources are consumes uselessly, these can be caused by denial of service (DoS) attack. Other forms of network intrusions also exist, such as land attacks, smurf attacks, and teardrop attacks. These attacks are not as well known as DoS attacks, but they are used in some form or another even if they aren't mentioned by name.

a. Eavesdropping

- Interception of communications by an unauthorized party is called eavesdropping. Passive eavesdropping is when the person only secretly listens to the networked messages. On the other hand, active eavesdropping is when the intruder listens and inserts something into the communication stream. This can lead to the messages being distorted. Sensitive information can be stolen this way.

b. Viruses

- Viruses are self-replication programs that use files to infect and propagate [8]. Once a file is opened, the virus will activate within the system.

c. Worms

A worm is similar to a virus because they both are self-replicating, but the worm does not require a file to allow it to propagate . There are two main types of worms, mass-mailing worms and network-aware worms. Mass mailing worms use email as a means to infect other computers. Network-aware worms are a major problem for the Internet.

A network-aware worm selects a target and once the worm accesses the target host, it can infect it by means of a Trojan or otherwise.

d. Worms

A worm is similar to a virus because they both are self-replicating, but the worm does not require a file to allow it to propagate [8]. There are two main types of worms, mass-mailing worms and network-aware worms. Mass mailing worms use email as a means to infect other computers. Network-aware worms are a major problem for the Internet. A network-aware worm selects a target and once the worm accesses the target host, it can infect it by means of a Trojan or otherwise.

e. Trojans

- Trojans appear to be benign programs to the user, but will actually have some malicious purpose. Trojans usually carry some payload such as a virus.

f. Phishing

- Phishing is an attempt to obtain confidential information from an individual, group, or organization. Phishers trick users into disclosing personal data, such as credit card numbers, online banking credentials, and other sensitive information.

g. IP Spoofing Attacks

- Spoofing means to have the address of the computer mirror the address of a trusted computer in order to gain access to other computers. The identity of the intruder is hidden by different means making detection and prevention difficult. With the current IP protocol technology, IP-spoofed packets cannot be eliminated.

h. Denial of Service

- Denial of Service is an attack when the system receiving too many requests cannot return communication with the requestors. The system then consumes resources waiting for the handshake to complete. Eventually, the system cannot respond to any more requests rendering it without service.