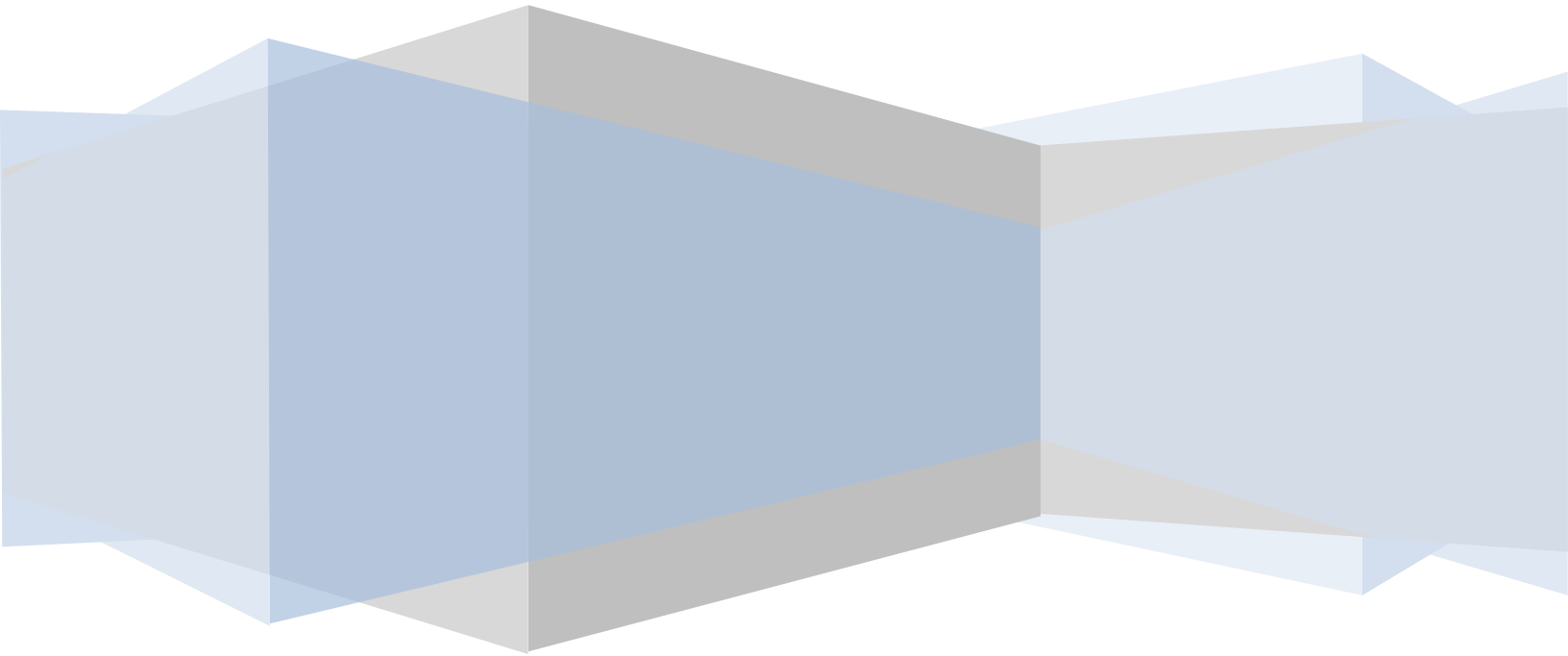


DATABASE MANAGEMENT SYSTEMS

B.Sc (CS) – 2020

Compiled by

Mr. M. Riyaz Mohammed M.C.A., M.Phil.,
Assistant Professor
Jamal Mohamed College (Autonomous)
Tiruchirappalli-620020.



UNIT –I

1. What is Information?

Information is nothing but redefined data.

2. What do you mean by quality of information?

Information with accurate, timely and relevant is meant by quality of information.

3. What are the attributes of information?

Accuracy, timeliness and relevancy are the three key attributes of information.

4. What is information processing?

Information processing is the acquisition, storage, organization, retrieval, display and dissemination of information.

5. What do you mean by instance of a database?

The collection of information stored in a database at a particular moment is called an instance of the database.

6. What are information islands?

The organizational structure promotes data ownership, thus promoting the storage of the same basic data in different locations. Database professionals use the term information islands or information compartmentalization.

7. What is structural Dependency?

In any file's structure addition or deletion of a field requires the modification of all programs using that file. Such modifications are required because the file system exhibits structural dependency.

8. What is data dependency?

All data access programs are subject to change when any of the file's data characteristics change. This is because the file system is dependent on the data or in other words the file system exhibits data dependency.

9. Define: Database

A collection of data designed to be used by different people is called a database.

10. What is data independence?

The structure of data files is stored in the DBMS catalog separately from the access programs. This property is called program – data independence.

11. What is data abstraction?

The characteristic that allow data independence is called data abstraction.

12. What is a multiuser DBMS?

A multiuser DBMS must allow multiple users to access the database at the same time. A fundamental role of multiuser DBMS software is to ensure that concurrent transactions operate correctly.

13. What are the components of a database?

Data items, Relationships, Constraints and Schema are the components of database.

14. What do you mean by data?

Data are binary computer representations of stored logical entities.

15. What is data dictionary?

Data dictionary is a file that defines the basic organization of a database. A data dictionary contains a list of all files in the database, the number of records in each file and the names and types of each field.

16. What do you meant by metadata?

The meta data provide a description of the data characteristics and the set of relationship that link the data found within the database.

17. What are relationships in a database?

Relationship represents a correspondence between the various data elements such as constraints and schema.

18. What are database constraints?

Constraints are predicates that define correct database states.

19. What is a database schema?

The schema defines various views of the database for the use of the components of the database management systems and for the application's security.

20. What is the difference between internal and external schema?

The internal schema defines how and where data are organized in physical data storage. The external schema defines a view or views of the database for particular users.

21. What is a data model?

A database model is a collection of logical constructs used to represent the data structure and the data relationships found within a database. The data model provides the necessary means to achieve data abstraction.

22. What are the different types of data model?

Data models can be grouped by the types of concepts they use to describe the database structure as conceptual (high level) model, physical (low level) and implementation (representational)models.

23. Name a few database models.

The following are the few database models:

1. Entity Relationship (ER) Model
2. Hierarchical Model
3. Network Model
4. Relational Model.
5. Object Oriented Model.

24. What do you mean by the conceptual or high level data model?

The conceptual model focuses on the logical nature of the data representation. They provide concepts that are close to the way many users perceive data.

25. What are physical or low level data model?

The physical data model is purely meant for computer specialist to describe how data is stored as files in the computer by representing information such as record formats, record orderings, and access paths.

26. What are the two types of physical data model?

The two most popular physical data models are

1. Unifying Model
2. Frame Memory Model

27. What do you mean by implementation or representational data model?

Implementational data model describes on how data is represented in the database. This model represent data by using record structures and are sometimes called record-based data model.

28. Why do we model data?

The two main purpose of modeling the data are

1. To assist in the understanding of the meaning- semantics of the data.
2. To facilitate communication about the information requirements.

29. What are the different types of database relationships?

The three different types of database relationships are

1. 1:n Relationship
2. n:1 Relationship
3. n:n relationship

30. Define: Distributed Database

A distributed database is one that can be dispersed or replicated among different points in a network.

31. Define: Object Oriented Database

An object oriented programming database is one that is congruent(similar) with the data defined in object classes and subclasses.

32. Differentiate flat file and relational database

Flat file database:

- i) A flat file consists of only one file or table.
- ii) The flat file has the demerits of data redundancy.

Relational database:

- i) Relational database contain one type of data.
- ii) It contain information as to how to retrieve data from other files.
- iii) Here data redundancy is mostly avoided and is easy for updating also.

33. Mention some of the advantages of database.

The advantages of having a centralized control of data are

- i) Redundancy can be minimized
- ii) Inconsistency can be avoided
- iii) A data can be shared.
- iv) Standard can be enforced.
- v) Security can be enforced.
- vi) Integrity can be kept intact.
- vii) Data independence.

34. Explain briefly about the architecture of database systems.

The architecture of database systems consists of three views

- i) Internal view
- ii) Conceptual view
- iii) External view

INTERNAL VIEW:

The internal components are the one near the physical storage i.e, it concern about how the actual data is stored. It is a very low level representation of entire database.

CONCEPTUAL VIEW:

The conceptual view level establishes mapping between the external level and the internal level. Also it is meant for community of users from where individual view can be defined.

EXTERNAL VIEW:

The external view level is the one, which indicates how easy users view the data.

35. Mention the role of database administrator

The database administrator has the important role to play. It is a very senior position. The roles are follows:

- i) Defining the content of the database.
- ii) Defining the storage structure and access mechanism.
- iii) Co-ordinating the users.
- iv) Defining the authorization checks and validation procedures.
- v) Lay down policy for backup and recovery.
- vi)

36. Explain how database management system has implemented in modern database packages

- A database management system (DBMS) is a software package that controls all access to database.
- It allows database administrator, **system administrator (SA)** and **system security operators officer (SSO)** to do their respective roles.
- When a request to access database comes, it examines
 1. The request interprets it using some data manipulation languages.
 2. External schema, converts external-conceptual mapping.
 3. Conceptual schema, converts conceptual-internal mapping.
 4. Internal schema, converts conceptual-internal storage mapping and
 5. Internal storage and performs necessary operations to retrieve and stored data.

37. Mention some of the features of DBMS.

DBMS offers the following features like

- i) Data definition language(DDL)
- ii) Data manipulation language(DML)
- iii) Concurrency control
- iv) Automatic recovery and backup
- v) Replication
- vi) High availability
- vii) Auditing

38. Mention the roles of System Administrator and System Security officers.

Role of System Administrator

- Installing DBMS if Sybase then adapting server (AS) and backup server.
- Creating and managing login accounts of adapting server(AS).
- Granting rolls and permission for adapting server (AS) users.
- Managing and monitoring the use of disk space, memory and connection.
- Backup and restoring data base.

-
- Configuring parameters of adapting servers (AS) to advice lost performance.

Role of System Security Officers

- Creating server login account and assigning passwords.
- Changing the password of any account.
- Granting and revoking the SSO and operator roles.
- Creating,granting and revoking user defined roles.
- Granting the capability of another user through out the server.
- Setting the password and expiration interval.
- Managing the audit system.

39. Mention the two kinds of object owners?

The two types of ownerships are:

- Database owner
- Database object owner.

40. Mention the role of database owner.

The database owner is the one who had created database or to whom database ownership is transferred.

The system administrator grants users the authority to create database with the GRANT command.

The database owners can:

- ➔ Use system procedures. **SP-add users** to allow DBMS user access to the database..
- ➔ Use **GRANT** command to give other user permission to create objects and execute the commands within to database.
- ➔ A database owner login to (AS) using his logging name and password
- ➔ In other DBMS the owner is known as owner but he is known by his known by his regular user name.

41. Mention the role of database object owner.

The database object owner is a user who creates database objects

1. Tables
2. Indexes
3. Views
4. Defaults
5. Triggers
6. Rules and Procedures

There is no special login name or password for database object owner. The database object owner can create an object using **CREATE** statement and then give permission to other users.

The database owner cannot use an object directly until the **DBO** does not grant him the permission. But, he can always use the **SETUSER** command to camouflage any other user in the database including the object owner.

42. Mention the various names of database (optional databases)

When DBMS is installed, it contains the following system databases

- The master database
- Model database
- The temporary database(**TEMPDB**)
- The system procedure database

The other optional database are

- Auditing database
- The two-face commit transaction database and
- Database consisting checks databases
-

43.Explain the various databases in detail.

The various databases are:

- Master database
- Model database
- Temporary database
- System procedure database
- SYBDIAG database
- SYB security database
- SYB system DB database
- DBCCDB database
- SYS database

Master database

The master database keeps system table. The tables keep track of information about DBMS as a goal. Each user database is created with a subset of the system table. The system tables are also known as data dictionary or system catalogue. A master database and its table are created as soon as adapter server is installed

Model database

Adaptive server has the model database. It acts as a template for new user database. Whenever a user executes the create database command, As makes a copy of the model database. It contains the required system tables for each user database. The size of new created database cannot be less than the model database, and the size of model database cannot be larger than the size of temporary databases

Temporary database

This database keeps the temporary tables and caters to the temporary working storage requirements. The storage space defined in temporary database is shared by all users of all databases on the server. The default size of temporary database depends on logical page size of the server which can be 2,4,8,16,k.

The size of temporary database is increased if:

- You want to keep large temporary tables
- Large number of simultaneously, sub queries and aggregate with group clause are taking place.

System Procedure Database

The system procedures are stored in a DBMS supplied database. When the user executes the stored procedure from any database, the database server first tries to find it in current users database. If it is not found there, then it looks for it in syvsysdemprocs when users executes the stored procedures the corresponding procedure will ne activated.If it does not find there also, then it looks for it in master database

SYBDIAG database : This database is created for debugging process.

SYB Security database: This contains audit system, for Sybase database.

SYB System DB database:It stores information about distributed transaction. The sys coordination table of the DBMS keeps information about remote server participation in distributed transaction.

DBCCDB database:This database keeps configuration information of the target database, operation activity and outcome of the operation in **DBCCDB** database

Sys database: The sys database table has one entry for each database created on the system. When SQL server is installed, master, temporary and model are present

44. Explain actionable of database administrator.

Some of the responsibilities related to database administrator is described follows:

If **SYBASE** is the data server then **DBA** tasks are:

1. installing data server
2. creating devices
3. creating database
4. viewing device information

1. Installing data server

- i) space required for all types of databases:113MB
- ii) install issues SVRBUILT and do the following task:
 - a) install license manager
 - b) create ASE and install the name for ASE.
 - c) Install master dot and give its size
 - d) Give path of error dot log file
 - e) Give the connection information
 - f) Install backup server and specific error log file name

2. Creating devices

It is a process that prepares the devices for storage and makes it known to the server (DBMS). Disk init is the command used for disk initialized. The name is listed in Master...sysdevices.

UNIX example:

```
Raw partition
disk init
name=dev_date_2";
phy name="/dev/rxyID";
vdev no=2,size=5120
(vdevno=virtual device number.)
```

3. Creating Databases

The syntax for creating database is

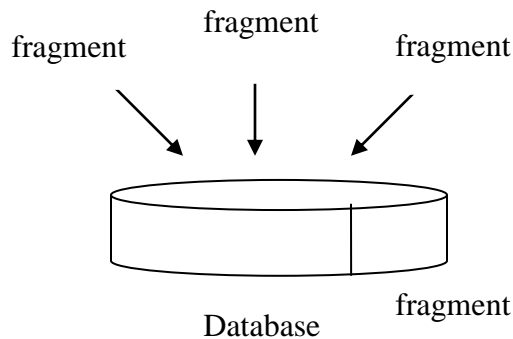
```
Create database database_name
[ on { default| database_device } [ = size[K|M|G]]
[ ,.database_device [=size[K|M|G]]
.....
]...]
[ with override]
[ for load]
File System
Disk init
Name="dev_date_2",
Physname = "/dev/dev_date_2.dat",
Vdevno=2,size=5120
```

4. Viewing Device Information

```
Sp_helpdevice dev_data_2.
```

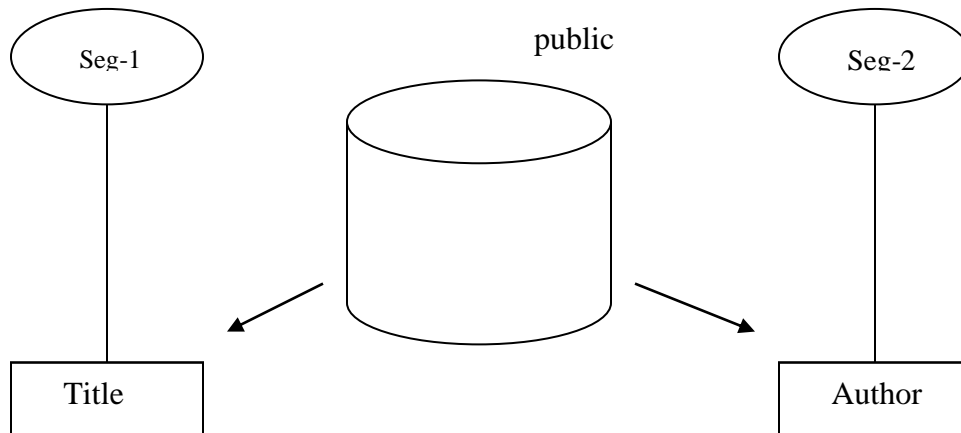
45. What is database fragment?

Whenever a database is created, the server act one entry in sys usages for each device on which is created. Each entry in sys usages is known as database fragment



46. Write a short note on Segments.

A segment is a named collection of one or more devices assigned to a given database. When segments are created, high underscore use tables can be placed on one segment and another high underscore use tables .on another segment for letter performance.



By default, three segments are created when a database is create

- 1. System Segment:** Store system tables of master database.
- 2. Default Segment:** Store database objects such as tables and indexes.
- 3. Log segment:** Stores database transaction logs(sys logs)

47. Mention some of the scope of E-R model.

The Entity-Relationship [E-R] model is used as an information model to develop the conceptual structure.

The scope of E-R model includes:

- Entity and entity sets and reducing E-R diagram to tables
- Relationship and relationship sets
- Attributes
- Mapping constraints
- Keys
- Entity Relationship diagrams
- Representing Strong Entity Sets, Weak Entity Sets & Relationship Sets.
- Generalization
- Aggregation

48. Mention the components of E-R model.

- E-R model forms the basis of the E-R diagrams
- E-R diagrams represents the conceptual view of database
- E-R diagram represents the main components:

- i) **Entities**
- ii) **Attributes**
- iii) **Relationship**

49. What is an Entity?

The fundamental item in any data model is the entity. An Entity is viewed as the atomic real world item. An entity is a instance of an entity type that is uniquely identifiable.

A database normally contains many different entity types. Although an entity type has a distinct set of attributes, each entity has its own values for each attributes.

50. Explain about attributes.

An attribute is a single atomic unit of information that describes about something its named entity for example, the attributes of a entity **BOOK** can be **ISBN**, title, author, publisher, price, year of publication etc.

This attribute provide additional information about entity. We can classify the attributes as follows:

- i) Simple attribute
- ii) Composite attribute
- iii) Single valued attribute
- iv) Multi valued attribute
- v) Derived attribute

i) Simple attribute:

A simple attribute is a attribute composed of an single components with an independent existence simple attributes cannot be further sub divided

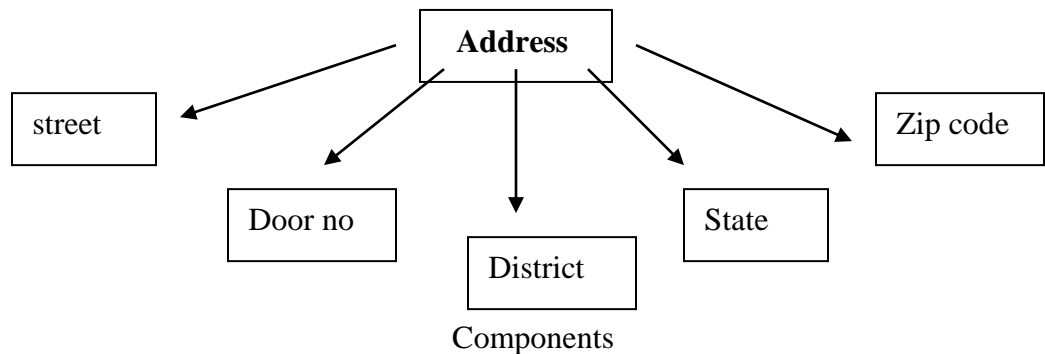
Example: Age, Salary, Sex etc..,

Simple attributes are sometimes called as atomic attributes.

ii) Composite attributes:

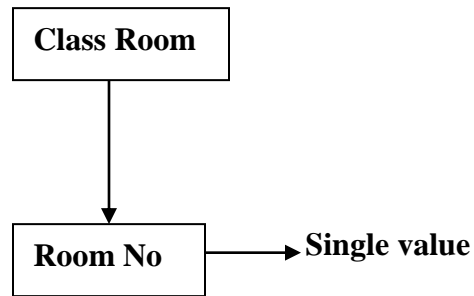
An attributes composed of multiple components, each with an independent existence is called a composite attribute.

Some attributes can be further divided to yield smaller components with an independence existence of their own.



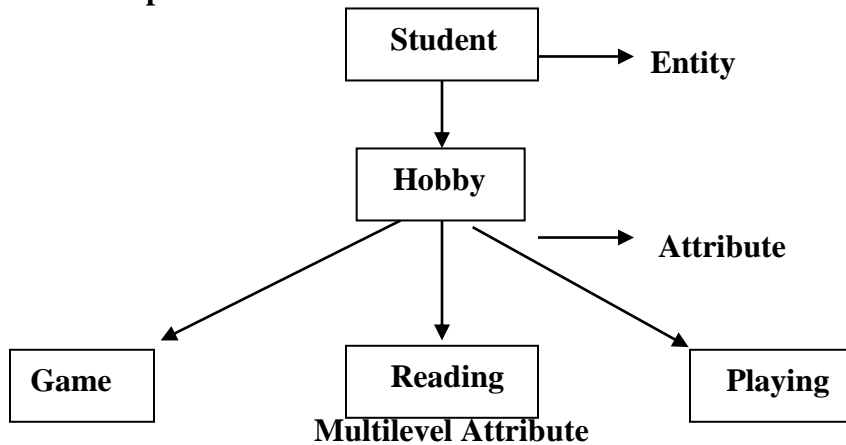
iii) Single valued attribute: The single values attribute is one that holed a single valued for a single entity. The majority of attributes are single valued for a particular entity.

Example:



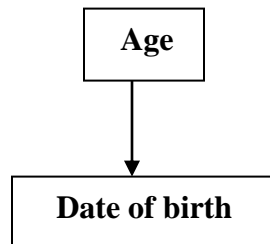
iv) **Multi valued attribute:** A multi valued attribute is one that holds multiple values for a single entity. Some attributes have multiple values for particular entity.

Example:



iv) **Derived attribute:** A derived attribute is one that represents a value that is derived from the value of a related attribute or set of attributes, not necessarily some entity.

Example



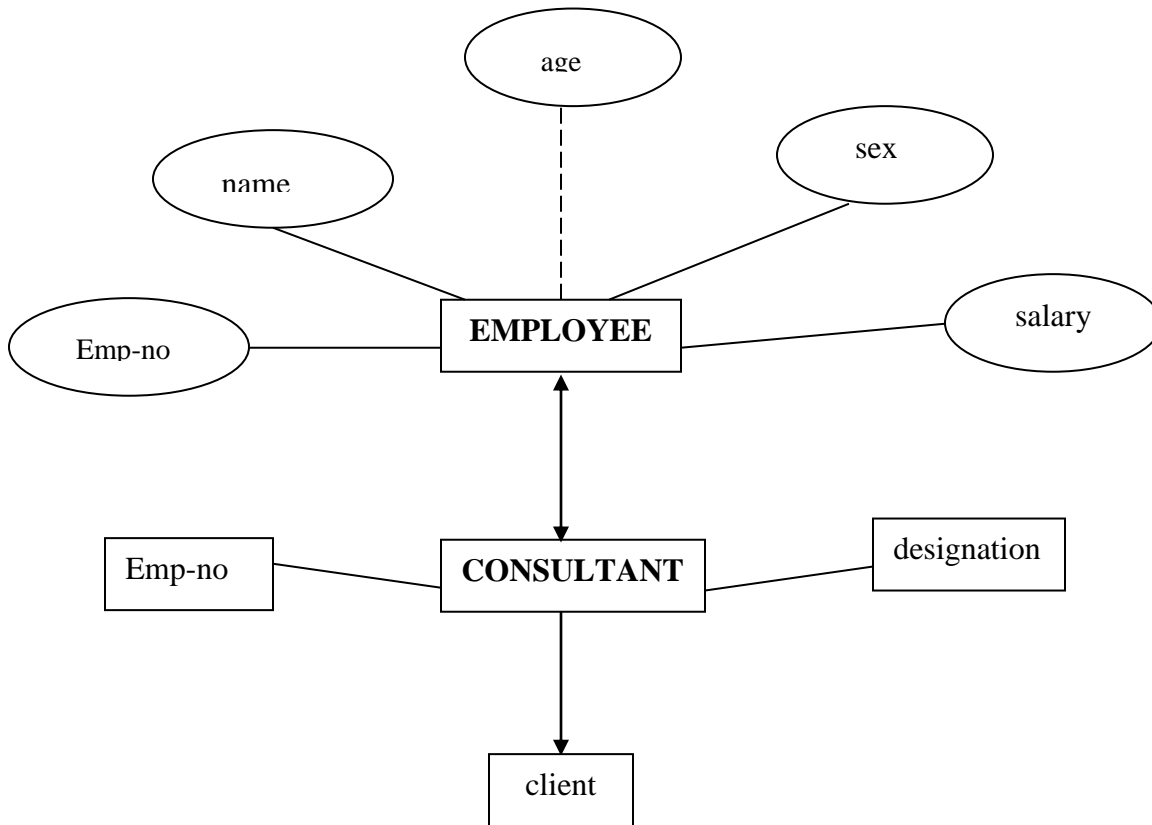
51. Explain about Relationship in detail.

A relationship is an association between entities.

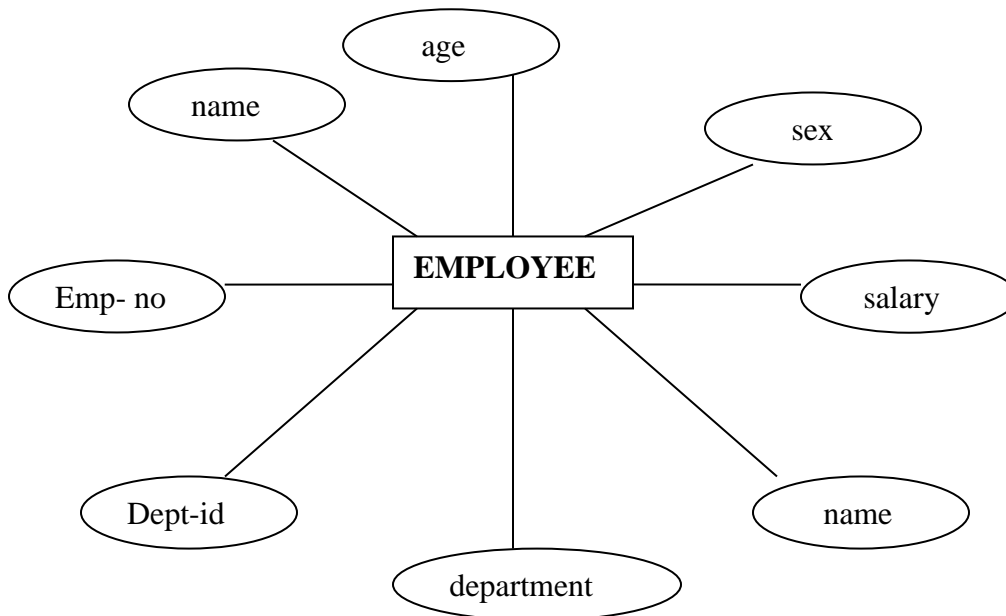
The various classification of relationships are

- **One-to-one relationship**
- **One-to-many relationship**
- **Many-to-many relationship**

ONE-TO-ONE RELATIONSHIP:

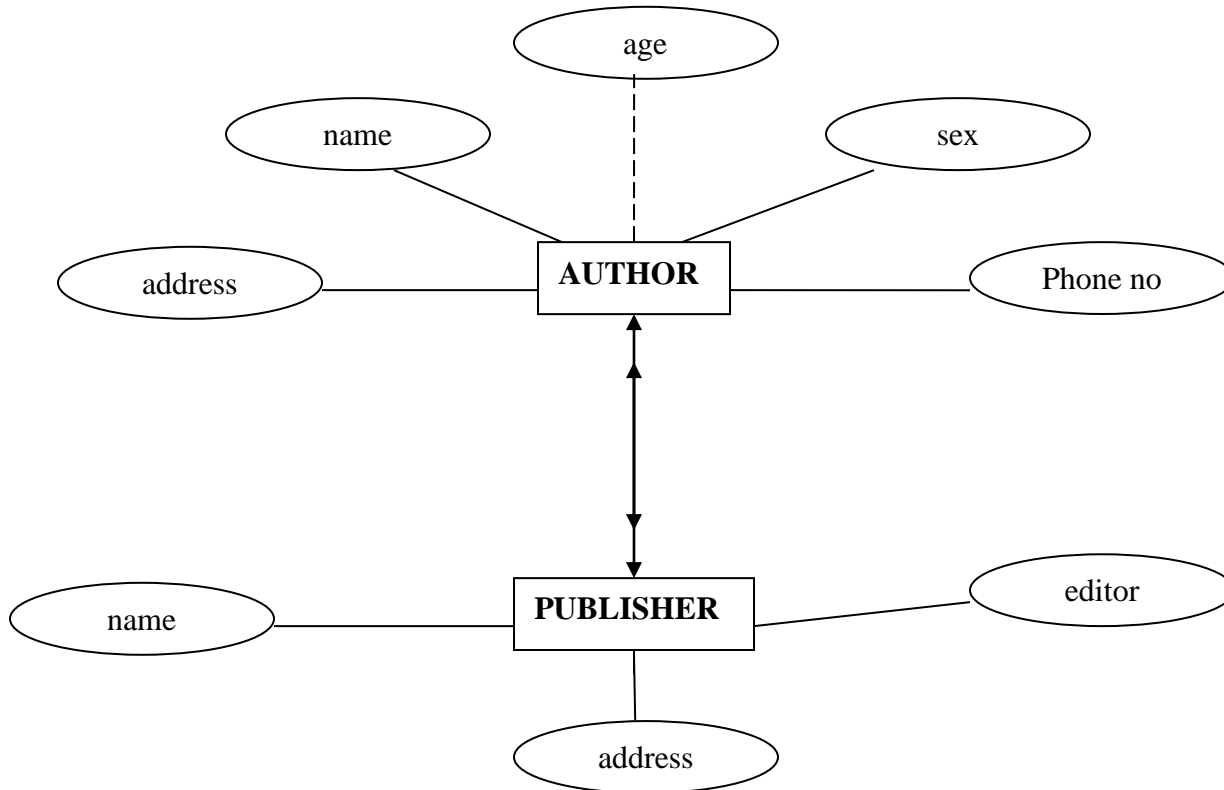


2. ONE-TO-MANY RELATIONSHIP:



(Assuming that an employee works for only one department and a department can have more than one employees)

3. MANY-TO-MANY RELATIONSHIP:



DATA MODELS

52. *Mention the names of various datamodels..*

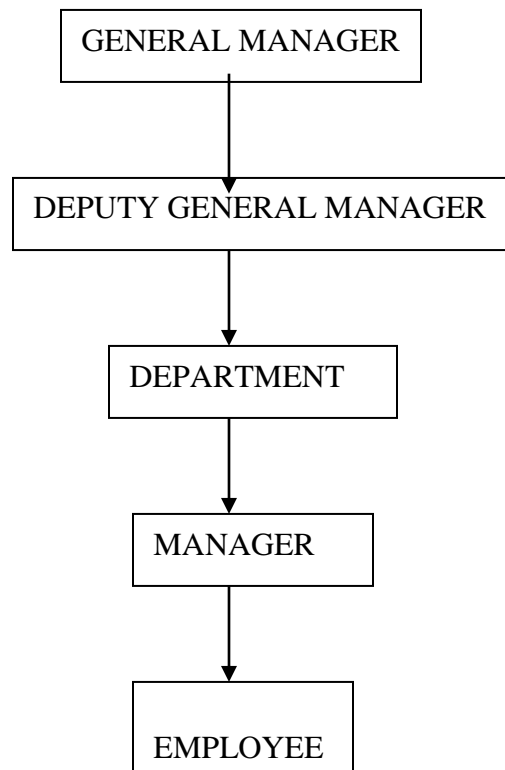
The various data models are

- E-R model [Entity Relationship model]
- Hierarchical model
- Network model
- Relational model
- Object oriented model

53. *Write a short not on hierarchical model.*

Hierarchical database model is one of the oldest data model, dating from 1950's. The hierarchical model assumes a tree structure is the most frequent acquiring relationships. The hierarchical model organizes data element as table rows, one for each instance of an entity

Example



54. Mention the various advantages and disadvantages of hierarchical models.

Advantages:

1. Simplicity: Since the database is based on the hierarchical structure, the relationship between the various layers is logically simple.

2. Data Security: Hierarchical data model is the first database model that enforced security by the DBMS

3. Data integrity: Since the model is based on parent/child relationship, there is always a link between the parent segment and the child segment under it.

4. Efficiency: A hierarchical database model is very efficient one. When a database contain a large number of 1:n relationships and when the users require a large number of transaction using data whose relationship are fixed.

Disadvantages:

1. Implementation complexity: Although this model is conceptually simple and easy to design, it is quite complex to implement the database designers should have a very good knowledge of the physical data storage characteristics.

2. Database management problems: If you make any changes in database structure of a hierarchical structures, then you need to make necessary changes in all the application programs

that access the database. Thus, maintaining the database and the application can become very difficult.

3.Lack of structural independence: Structural independence exist when the changes to the database structure do not affect the DBMS's ability to access data. Hierarchical database system use physical storage part to navigate to the different data segments. So, the application programmers should have good knowledge of the relevant access paths to access the data. So, if the physical structures is changed the application will also to be modified. Thus, in a hierarchical database the benefits of data independent is limited by structural independence.

4. Programming complexity: Due to the structural dependence and the navigational structure, the application programmers and the end user must know precisely how the data is distributed physically in the database in order to access data. This requires knowledge of complex pointer system, which is often beyond the grasp of ordinary users.

5.Implementation limitation: Many of the common relationships do not confirm to the 1:n format required by the hierarchical model. The many-to –many (m:m) relationships, which are more common in real life are very difficult to implement in a hierarchical model.

55. Define:Network model

The network model replaces the hierarchical tree with a graph

Tree with a graph:

- 1) Thus allowing more general connections among the nodes.
- 2) The main difference of the network model from the hierarchical model is its ability to handle many-to –many relationships (n:n)
- 3) In network database terminology, a relationship is a set.

56. Explain the advantages and disadvantages of Network Model.

Advantages:

1. Conceptual Simplicity: Just like hierarchical model, the network model is also conceptually simple and easy to design.

2. Capability to handle more relationship types: This model can handle one-to-many relationship and many-to-many relationship, which is a real help in modeling the real life situation.

3.Easy of data access: The data access is easier and flexible than in the hierarchical model, and an application can access an owner record and all the member records within a set and if one member in the set has two owner, then one can move from one owners to another.

4. Data integrity: The network model does not allow a member to exist without a owner. Thus, the user must e first define the owner record and then the member record. This ensure the data integrity

5. Data independence: The network model is better than the hierarchical model in isolating the programs from the complex physical storage details.

6.Database standards:One of the major drawback of the hierarchical model was the availability of universal standards for database design and modeling, the network model is based on the standards formulated by the DBDG (Database task group of POTASOL comity) and augmented *ANSI / SPARC (American national standard institute/ standard planning and requirement community in 1970's)*

Disadvantages:

1.System Complexity: The navigational data access mechanism makes the system implementation very complex and conqently the database administrators, database designs, programmers and even the end user should be familiar with the internal data structure in order to access data.

2.Absence of structural independence:Since the access method is the navigational system, making structural changes to the database is very difficult in most cases and is impossible in some cases.If the changes are made to the database structure then all the application programs need to be modify before they can access data.

57. What is Relational data model?

Relational model stores data in the form of a table. Relational database are powerful because they require few assumption about how data is related and are how it will be extracted from the database. As a result, the same databases can be viewed from the different ways. Another feature is single database can be spread across several tables. This differs from flat file databases is self contained in a single table.

58. Explain the Advantages and disadvantages of Relational Model.

1.Structural independence:

Changes in the database structure do not affect the database access. Hence the database model has been achieved by the structural independence.

2.Conceptual simplicity:

It is more simple than network and hierarchical database model at the conceptual level. Since this model freeze the designers from the physical data storage details. The designers can concentrate on the logical view of the database.

3. Design, implementation, maintenance and easy usage:

The relational database model achieves both the data independence and structural independence, design administration and the usages much easier than the other model.

4. ADHOC Query capability:

The presence of very powerful, flexible and easy to use query capability is one of the main reasons for the immense popularity of the relational database model. The SQL – Structured Query Language makes ADHOC Queries a reality. SQL is the 4th generation language [4GL].

Disadvantages

1. Hardware overheads.
2. Easy of design can lead to bad design
- 3.Information Island phenomenon policy.

UNIT –II

1. Define: File

A file is nothing but a collection of records.

2. Define: Attribute

An attribute is a characteristics of interest about an entity.

3. Define: Instance

An instance of the entity is represented by a set of specific value for each of the attributes.

4. Define: Data item

Each attribute of an entity is represented in storage by a data item.

5. Mention the various types of files.

Master file:

This is a file relating the permanent information about entities. Eg. The accounts master file in a bank will contain details like a) account number, b) type of accounts, c) name of the account holder and address, d) data code wording the a./c, e) name of the nominee, f) balance in the a/c, etc.,

Transaction file:

This is a collection of record, describing activities or transactions by organization. Eg. The transaction file of the bank will contain a) records of day to day activities of the bank, b) deposits, c) withdrawals, d) fund transfer and so on.

Report file:

This is a file created by extracted data to prepare a report a report of all accounts shorted by the account number and containing the details like account holders name and account balance, etc.,

6. Explain the operation on the file.

The two main file operations are,

i. Retrieval operation:

It do not change the content of the file. It only locates records in the file matching certain specified criteria.

ii. Update operation:

It is change the file by modifying the records, editing the records and inserting the new records. The actual operation for location modifying detecting and inserting recorded will vary from system to system, there are several representative operation that are used in most system. Some of them are give below.

Find (locate)

The goal of this operating is to locate the record or records that satisfies the search criteria.

Read

The contents of the records are copied from the memory to a program variable or work area.

Read next (or) Get next

Searches for the next records that matches the selection condition and when found the content of the record are copied to the programme variable or work area.

Modify

Also known as update. This command modifies the field value of the current record and the writes the modified record back to the disk.

Insert

Insert a new record to the file. It involves many process like

- a) finding a place to insert.
- b) writing a new to the disk.
- c) updating the index entries.
- d) updating the record headers and so on depending on the file type and the insertiion mechanism.

Delete

Delete the current record and updates the file on the disk to reflect the deletion.

7. What are the different types of files?

The different types of files are Master files, Transaction files and report files.

8. What is a primary key and what is its importance?

A primary key is a field or set of fields whose contents is unique to one record and its importance is it is used to identify that particular record.

9. What is direct file organization?

In a direct file the data may be organized in such a way that they are scattered throughout the disk in what may appear to be a random order.

10. What is direct file processing?

Processing data using direct access is referred to as direct file processing.

11. What are secondary indexes and why it is used?

Secondary indexes can be used to search a file on the basis of secondary keys. If a user need to access the data by another key (not by primary key), in this situation secondary index could be constructed with the another field name as the secondary key.

12. What do you mean by index sequential file processing?

Index sequential is another method of direct processing that combines the features of both sequential and direct access. In addition to a primary index, index sequential access establishes a secondary index in which all the records are arranged by ascending or descending primary key. When the file needs to be processed sequentially, the index is followed from the first entry to the last, thereby processing the files sequentially.

13. What is ISAM?

ISAM is Index Sequential Access Method.

14. What is VSAM?

VSAM is Virtual Storage Access Method.

15. What do you mean by the term file volatility?

When data are frequently added to or deleted from a file, the file is said to have a high volatility.

16. What is file activity?

The percentage of record in a file that is actually accessed during any one run is the file activity rate.

17. What is File Query?

When information must be retrieved very quickly, then some form of direct organization must be used. Railway reservation system, inventory systems and ATM systems all falls into this category.

18. What is data currency?

Data currency refers to the timeliness of data. If the data need to be up to the minute, then direct organization and processing will be required. Reservation systems and on-line shopping systems all depend on timely data and therefore depend on direct systems.

19. What is buffering?

The address of the area in the main memory reserved to hold the contents of the block called buffer.

20. What is double buffering?

The technique of parallel data processing by the CPU and data transfer by the I/O processor is called double buffering.

21. What is record format definition?

A collection of field names and their corresponding data types constitutes a record type definition or record format definition.

22. What are the different types of records?

Fixed Length Records and Variable Length Records are the two types of records.

23. What are dangling pointers?

The incorrect pointers are called dangling pointers.

24. What are pinned records?

To avoid dangling pointers, the records that contain pointers should not be moved or deleted. Such records are termed as pinned records.

25. What are variable length records?

Files could also be made of records which are of different sizes. These records are called variable length records.

26. What is a pointer and how are they used in file organization?

A pointer is a field of data that can be used to locate a related field or record. In most cases , a pointer contains the address or locations of the associated data.

27. What is a file organization?

A file organization is a technique for physically arranging the records of a file on secondary storage devices.

28. What is sequential file organization?

In a sequential file organization, the records in the file are stored in sequence according to a primary key value. To locate a particular record, a program must scan the file from the beginning until the desired record is located.

29. What is Indexed-sequential file organization?

In an indexed file sequential , the records are stored either sequentially or non-sequentially and an index is created that allows the applications to locate the individual records using the index .In the indexed organization , if the records are stored sequentially based on the primary key value , then that file organization is called an indexed sequential organization.

30. What are root nodes and leaf nodes in an index hierarchy?

The general structure of a hierarchical index is called a tree. This tree will have the root node (the top most index file) at the top and the leaf nodes (the records) at the bottom.

31. What is a B-tree index?

The most common type of tree used by access methods and database management systems to store indexes is a balanced tree or B-tree.

32. What is B⁺- tree?

The most popular form of B-tree is the B⁺- tree. One of the index structures that maintain its efficiency even with the insertion and deletion of data is the B⁺- tree index structure. A B⁺- tree index takes the form of a balanced tree in which every path from the root to the tree leaf is of the same length. Each non-leaf node in the tree has between 'n/2'and 'n' children, where 'n' is fixed for a particular tree.

33. What is a hashed file organization?

In a hashed file organization, the address of each record is determined using a hashing algorithm. The hashing algorithm is a routine that converts a primary key value into a record address.

34. What is indexing?

An index for a file system works in the same way as the catalog in a library.

35. What is an ordered index?

An ordered index is based on a sorted ordering of the values.

36. What is hashed index?

A hashed index is based on the values being uniformly distributed using a mathematical function called hash function.

37. What are access type?

Access type could include finding records with a specific attribute value or finding records whose attribute values fall in a specified range.

38. What is access time?

Access time is the time taken to find a particular data item or a set of items using the given technique.

39. What are Insertion time and deletion time for index structures?

Insertion time is the time takes to insert a new record. This is the total time taken to find the correct place to insert the new records and the time taken to update the index structure. Deletion time is the time taken to delete a record. It is the sum of the time taken to find the record for deletion, the time for deleting the record and the time taken to update the index structure.

40. What is primary index?

If the files containing the records is sequentially ordered , the index whose search key specifies the sequential order of the file is called the primary index.

41. What is clustering index?

Primary index are also known as clustering index.

42. What is a binary search?

A binary search is performed by starting at the middle entry in the index and comparing the key in the index with the desired key. If the desired key is in the second half of the index, the search continues only in that half. The middle record of that half is examined and the search continues on the upper or lower half of that segment. This halving and comparison procedure continues until the desired key is located.

43. What is a secondary index?

Indexes whose search key specifies an order that is different from the sequential order of the file are called secondary indexes or nonclustering indexes.

44. What is hashing?

Hashing is a technique which allow us to avoid accessing an index structure.

45. What are hash files and hash fields?

Files using hashing techniques are called hash files. In the case of hashing the search condition must be an equality condition on a single field called the hash field of the file.

46. What is hash function or randomizing function?

The idea behind hashing is to provide a function called hash function or randomizing function that is applied to the hash field value of a record to yield the address of the disk block in which the record is stored.

47. What is hash table?

In the case of internal files, hashing is implemented as a hash table using an array of records.

48. What is mod hashing?

One common hashing technique is called mod hashing where the hash field value, which is an integer, is divided by 'm' and the remainder is returned. This value is used for the record address.

49. What is folding?

Folding is a hashing technique that involves applying an arithmetic function such as addition or logical function such as exclusive or (XOR) to different portions of the hash field value to calculate the hash address.

50. What are collisions in hashing and why do they occur?

A collision occur when the hash field value of a record that is being inserted hashes to an address that already contains a different record. In this situation the new record must be inserted to some other position because its hash address is already occupied.

51. What is collision resolution?

The process of finding another address position is called collision resolution.

52. Why are buckets used in hashing?

The collision problem is less severe when using buckets because more records can be assigned to a bucket without causing problem.

53. What do you mean by global depth and local depth in the case of extensible hashing?

In the extensible hashing scheme, a type of directory (an array of 2^d bucket addresses) is maintained. 'd' is called the global depth of the directory. The integer value corresponding to the first 'd' bits of the hash value is used as an index to the array to determine a directory entry and the address in that entry determines the bucket in which the corresponding records are stored. However, there is no need to have a distinct bucket for each of the 2^d directory locations. Several directory locations with the same first 'l' bits for their hash values may contain the same bucket address if all the records that hash to these locations fit in a single bucket. In this case 'l' is called the local depth.

54. Write a short note on file storage organization.

When data are stored on secondary storage devices, the method of file organization chosen will determine how the data are accessed, the time required to access the record and transfer the data to the primary storage.

The following five operations are required for the processing of records in a file.

- i) File creation
- ii) Record location (finding the record).
- iii) Record creation.
- iv) Record deletion.
- v) Record modification.

55. Mention the two forms of file organization

The two forms of file organization are

- i) Sequential file organization
- ii) Direct file organization.

56. Mention the advantages and disadvantages of sequential file organization or processing.

Advantages

- i) Magnetic tape, the least expensive method of secondary storage.
- ii) Efficient form of organization when the entire file must be processed at once.
- iii) Transaction file and old master file together act as a backup and can be used to create a new master file, if it is damaged or destroyed.

Disadvantages

- i) The time it takes to access a particular record may be too long.

-
- ii) The entire file must be processed and a new master file is created, even if only one record is required maintenance is required.

57. Explain sequential file organization.

In sequential file organization records are stored in some predetermined one after the other one field refer to as the primary key. Usually, determines the sequence are order.

A primary key is a field or set of fields whose content is unique is to one record and can therefore the used to identify then record. Primary keys usually include a student id, employee id, part no., and so on.

A company's payroll records, for eg., Might be stored sequentially by the employee number. This file would start with the record that has the lowest number, and the remaining records could be arranged in ascending order of employee numbers.

Sequential file organization is very common. Because, it makes effective use of the least expensive secondary storage device. A magnetic tape Another reason for its popularity is that the sequential processing at periodic intervals use a batch of transactions, is very efficient in many applications like payroll calculations, monthly billing etc.,

A disk storage device can be used much like a magnetic tape with data records stored and accessed in sequence.

58. What do you mean by indexed sequential organization?

When a files are stored on a disk, direct access of a desired record is possible. Indexed sequential file organization provides the facility of accessing records both sequentially and directly.

An Index of records stored on the disk. If large updates are required this organization allow records to be accessed sequentially, as well as provides the ability to access records directly for queries.

59. Mention the pit falls in a relational database design?

- i) Spreadsheet design.
- ii) Too much data.
- iii) Compound fields.
- iv) Missing keys.
- v) Bad keys.
- vi) Missing relations.
- vii) unnecessary relationship.
- viii) Incorrect relation.
- ix) Duplicate field names.
- x) Missing or incorrect business rules.
- xi) Missing or incorrect constraints.
- xii) Referential integrity.
- xiii) Database security.
- xiv) International issues.

60. Explain Direct file organization.

In a direct file, unlike a sequential form of organization, the data may be organized in such a way that they are scattered throughout the disk in what may appear to be a random order.

Direct file organization that supports direct access in which records can be accessed nearly instantaneously and in any order. Once, accessed a record can be read or updated, and when this process is completed, the system is free to respond to another request.

When using direct access, an application such as an online transaction processing system for inventory control can be designed so that centralized data are not only instantly accessible but one always up-to date.

Processing data using direct access is referred to as direct file processing. Most applications do not use transaction files, and there is no reason to create a new master file when a single record is updated or when maintenance must be performed. Direct processing requires either magnetic disk or optical disk and cannot use magnetic tape.

Advantages:

- i) data can be accessed directly and quickly.
- ii) Primary and secondary indexes can be used to search for data in many different ways.
- iii) Files can still be processed sequentially using a secondary index.
- iv) Centrally maintained data can be kept up to date.

Disadvantage:

- i) The use of an index lowers the computer system's efficiency
- ii) Because all data must be stored on disks, the hardware for these systems can be expensive.
- iii) Because the files are updated directly and no transaction files are maintained, there may be no backup if a file is destroyed.
- iv) Procedures must be established to ensure the regular creation of backup files.

61. Write a short note on Hashing.

Hashing also known as randomizing is a method for determining the physical location of record, in this method record key [such as employee number, part number, a patient number] is processed mathematically and another number is computed that represents the location where the record will be stored.

The task is to take a set of record keys and file a formula to map them in to a set of disk storage location identified.

If the record keys can sequentially with no gaps, then this would have been a fairly simple process, but such a situation almost never exists.

So the record keys are transformed in to storage addresses by using an arithmetic procedure called randomizing or hashing algorithm. The records all physical address.

Eg. Payroll system with 1000 employees. The employee number is in the range 1000 to 9999. Take a record whose record key is 4555 a prime number close to number of storage location.

We will divide 4555 by 999 a prime number close to 1000 and use the remainders so as the storage address of the record with remainder is 567 which is used as address of the record with record key 4555.

A major difficulty with randomizing procedure is that some addresses will never get generated while two or more records keep procedure identical disk addresses or synonyms or collision.

In such situation, one of the record is stored at the generated location and a mechanism is provided to store the synonym is an overflow.

Subsequently when a record is retrieve there must be a method of finding it in the overflow area.

Storage of records in the overflow are reduces the efficiency of the retrieval process, because the search for right record becomes more complex through the use of overflow areas and thus because more time consuming.

Another difficulty with hashed organization is that overflow becomes severe problems if most of storage locations are used.

Eg. 40% of no problematic, 90% of overflow is likely to become problematic.

Hashing address may refer to a bucket, or group of records instead of a single record location.

Group of record stored in the bucket is searched sequentially to find the desired record.

When a file has many keys that lead to the same physical storage location thus hashing may be inappropriate.

It is occasionally necessary to process the file sequentially in ascending or descending thought the disk.

Performance of hashed file organization:

File creation:

Each record is stored in same way as additions to an existing file the storage location is calculated using the key and the hashing algorithm and an overflow operation is performed if necessary.

Record location / modification:

Locating a particular record for inquiry or modification is efficient relative to sequential access if transactions cannot be put in sequence or the number of records to be modified is low.

Record creation / Deletion:

Becomes more inefficient as more and more records are added and storage begins to fill.

62. What is Indexing?

Common procedure for locating a record in a file is for system to store records randomly throughout the disk, but to provide one or more indexes to locate a particular record.

Primary index: It Associates a primary key with the physical location in which a record is stored.

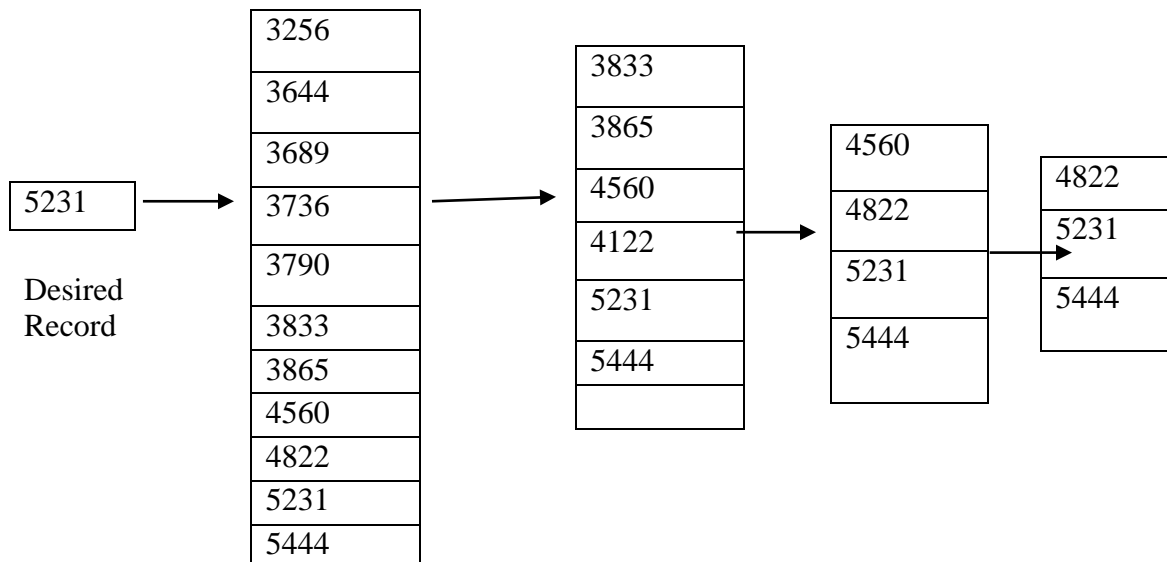
When a user requests a record the disk operating system first loads primary index in to computer CPU and then searches the index sequentially for the key.

When it finds entry for key, it then reads the address in which the record is stored. The Disk system then proceeds to this address and reads records contents.

Advantage of primary index is that it contains only two pairs of information the records key and the physical address of the record. The record includes many more fields of information. So the search of a small index and then direct access to the record is much faster than a sequential search of the data file itself.

If the index is in record key sequence, a binary search is an efficient procedure for locating a key. A binary search performed by starting at the middle entry in the index and comparing the key in index with desired key.

If the desired key is in second half of the index, the search continues only in that half. The middle record of that half is examined and the search continues on the upper or lower half of that segment.



The efficiency of a binary search procedure is shown by the fact that to locate a particular entry in an index of 2000 entries requires the examination of not more than 11 entries.

In addition to primary index, it is also possible to develop secondary indexes, which can be used to search a file on basis of secondary indexes. Index sequential is another method of direct processing that combines the features of both sequential and direct access. In addition to primary index, index sequential access establishes a secondary index in which all the records are arranged by ascending or descending primary key.

But processing a file sequentially in this way is inefficient, because index must be used to locate each record in file.

Index is also constructed so that a single record can be accessed directly. One variation of this approach developed by IBM, is called is called ISAM {Index Sequential Access Method}.

Performance of indexed file organization:

File creation

Requires extra procedures to create indexes.

Record location/Modification

Very efficient locating a particular record for online inquiry or updating is very efficient because of the use of index.

Record creation/Deletion

Addition requires adding a record to storage, setting up points and modifying indexes. When the overflow area fills up an ISAM file can be reorganize merging records in the overflow area with the records in the prime area to produce a new file with all the records in the proper sequences.

A more recent version of ISAM is known as VSAM [Virtual Storage Access Method].

The name virtual storage is given to this access method because the software for VSAM is so large that it may require virtual storage methods to enter the instructions. This is a more efficient method for adding records than the overflow method used in ISAM systems.

63. Describe about relational data structure?

The smallest unit of data in the relational model is the individual data values such values are assumed to be atomic which means that they have no interval structure as far as the model is concerned. A domain is a set of all possible data values. For eg, in a supplier parts the relation the domain is supplier in the no. of set of all valid supplied number. Thus, domains are pools of values, from which the actual values appearing in the attributes (columns) are drawn. The domain function is a very important an integral part of the relational model, because it has implications for comparisons and hence for operations such as joins and unions that directly or indirectly involves such comparison. If two attributes draw the values from the same domain, then comparisons involving those two attributes make sense because, we are like with like.

Table:

Emp-no	Name	Age	Department
E09898	Imran	60	Consultancy
E19765	Yogesh	50	Consultancy
E20193	Palanivel	60	Production
E21989	Prem	75	Operation
E25678	vasanth	80	Maintenance

Domains are conceptual in nature. They may or may not be explicitly stored in a database as actual set of values. But, they should be specified as part of Database definition and then each attribute definition should include a reference to the corresponding domain. Now, let us take a look at the relations. A relation on domain say $D_1, D_2, D_3, \dots, D_n$ consists of a heading and a body. The heading consists of a fixed set of distinct attributes, say $A_1, A_2, A_3, \dots, A_n$, such that

each attribute “a;” corresponds to exactly one of the underlying domains “D;”. The body consists of a time varying set of tuples, where each tuple is in turn consist of a set attribute value pairs (A_i, B_i), one such pair of each attribute A_i in the heading for any given attribute value pair (A_i,B_i), V_i is a value from the unique domain from D_i. That is associated with the attribute A_i.

64. Explain about indexing methods

An index for a file system works in the same way as the catalog in a library. Basically, there are two types of indexes.

1. Ordered indexes.
2. Hashed index.

Ordered indexes: An ordered index is based on a sorted ordering of the values.

Hashed indexes: A hashed index is based on values being uniformly distributed using mathematical function called hashing function. There are several techniques for implementing ordered indexing and hashing and each technique is best suited for a particular database application.

The following factors are:

- Access type
- Access time
- Insertion time
- Deletion time
- Space overhead

Access type: The types of access that are supported efficiently. These types could include finding records with a specific attribute value [like emp-no=12] or finding records whose attribute value fall in a specified range like price between 200 and 400.

Access time: Access time is the time taken to find a particular data item or a set of items using the given technique.

Insertion time: This is the time it takes to insert a new record. This is the total time taken to find the correct place to insert the new records and the time taken to update the index structure.

Deletion time: The time it takes to delete a record is deletion time. Deletion time is the sum of the time taken to find the record for deletion, time for deleting the record and the time take to update the index structure

Space Overhead: By space overhead we mean the addition space occupied by the index structure.

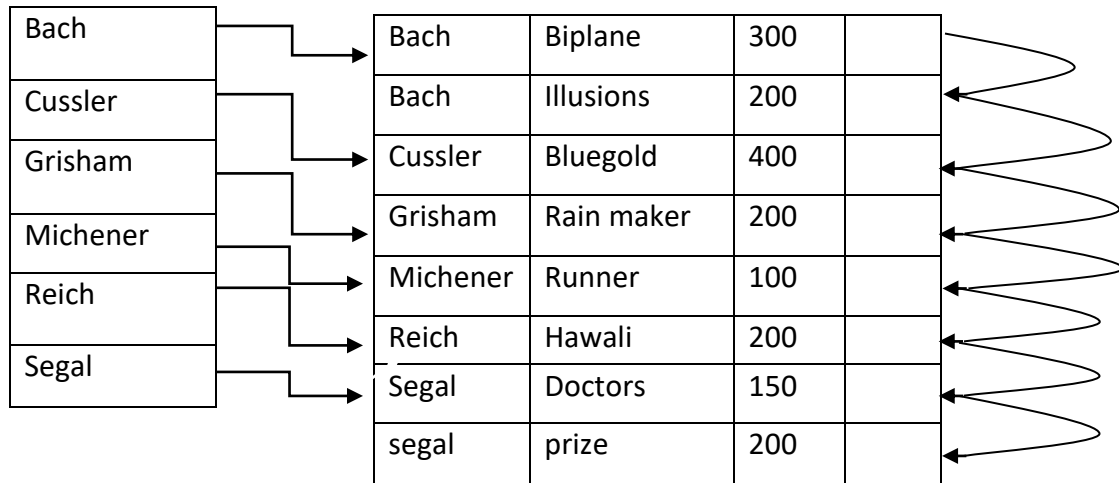
Ordered index: Ordered indexes are classified to two types:

- 1. Primary indexes:** If the file containing the records is sequentially ordered, the index whose search key specifies the sequential order of the file is called primary index or clustering indexes.
- 2. Secondary indexes:** The search key of a primary index is usually the primary key. Indexes whose search key specifies an order that is different from the sequential order of the file are called sequential indexes or non-clustering indexes.

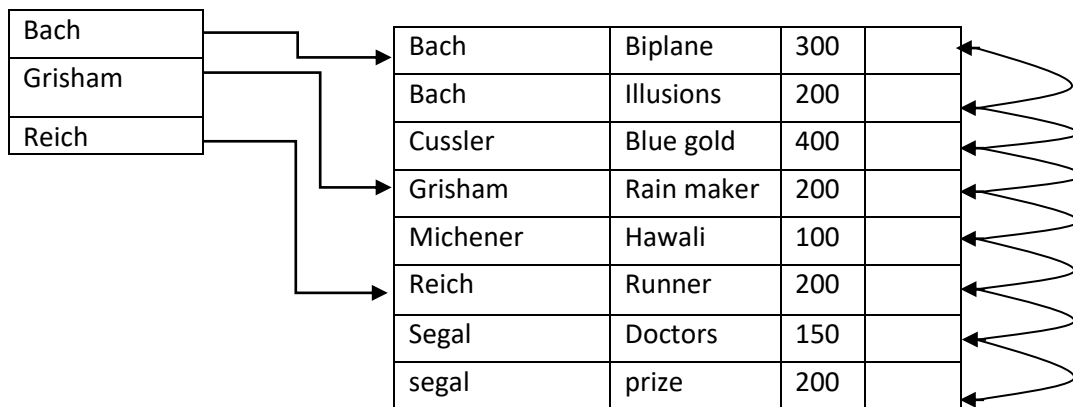
Primary Index : One of the oldest indexing scheme used in database systems are files with primary index on search key. These files are called index sequential file. It is designed for applications that require both sequential processing of the entire file and also random access to individual records. In case of a dense index, an index record appears for every search key value in the file. The index record contains search key value and pointer to the first data record with the search key values.

Bach	Biplane	300
Bach	Illusion	300
Cussler	Blue gold	300
Grisham	A pointed house	200
Michner	hawali	100
Reich	The runner	200
Segal	Doctor	150
Segal	The class	150

Bach	Biplane	300	→
Bach	Illusions	200	→
Cussler	Blue Gold	400	→
Grisham	Rainmaker	200	→
Michener	Hawaii	100	→
Reich	Runner	200	→
Segal	Doctors	150	→
Segal	Prize	200	→



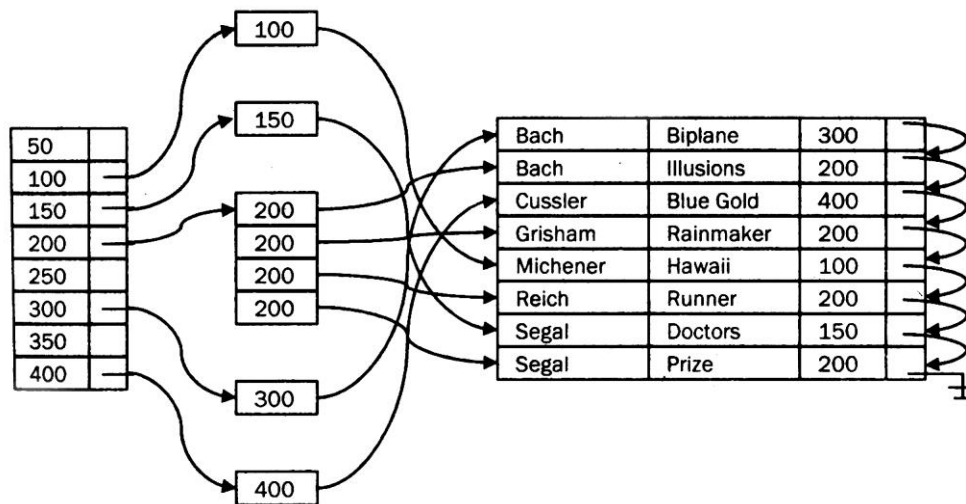
In case of sparse index, the index record is created only for some search key values. Each such index record contains a pointer to first record with that search key value. To locate a record, we find the entry with largest search key value that is less than or equal to the search key value for which we are looking.



Secondary index: Secondary index on a candidate key is similar to a dense primary index, except that the records pointed to by successive values in the index are not stored sequentially.

Secondary indexes can be structured differently from a primary index. If the search key of the primary index is not a candidate key and if the index points to first record with a particular value for search key, other records can be fetched by a sequential scan of the file

Search key of a secondary index is not a candidate key, then it is not enough to point to just the first record with each search key value, because remaining records with the same search key value could be anywhere in the file since the records are ordered by search key of primary index, rather than by search key of the secondary index. Pointers of the secondary index do not point directly to file, but a bucket that contains the points to the file.



Scanning the primary index is very efficient because the records in the file are stored physically in same order as the index order. We can use a sparse index for the primary index, where we store only some values of the search key. But it is not possible in secondary index.

Secondary index improve the performance of queries that use keep other than the search key of primary index. But they impose a considerable overhead on the database maintenance. So, the database designer should decide which of the secondary indexes are required based on the frequency of the queries and modification.

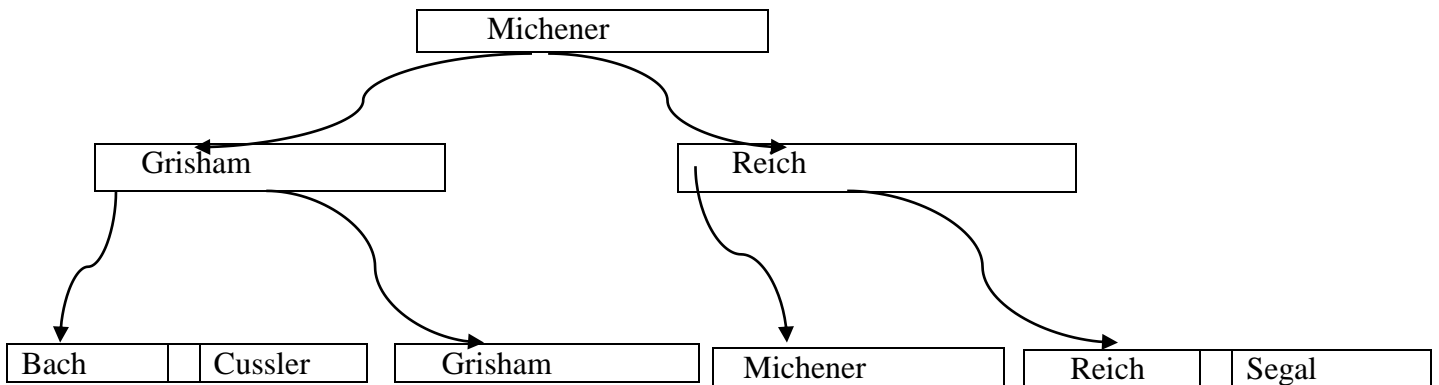
65. Explain B+- tree indexes

In the case of index sequential file, the performance degrades as the file grows. This is because the index lookups and the sequential scans take more time as more records are therein the file. Although this performance degradation can be overcome (to a certain extent) by reorganizing the file, frequent file reorganizations are undesirable and will add to the maintenance overheads. One of the index structure that maintain its efficiency even with the insertion and deletion of data is B+- tree index structure. A B+- tree index takes the form of a balanced tree in which every path from root to the tree leaf is of the same length. Each non-leaf node in the tree has between 'r/2' and 'n' children ($n/2 \leq c \leq n$), where 'n' is fixed for a particular tree. The B+ tree structure creates performance overheads on insertions and deletion and add space overheads. This performance overheads is acceptable even for files with high modification frequency, because the cost of file

organization is eliminated. Also there will be some amount of wasted space as some nodes will be half empty (nodes with $n/2$ children). It contains up to $n-1$ search key values k_1, k_2, \dots, k_{n-1} and n pointers $p_1, p_2, p_3, \dots, p_n$. The search key values within a node are kept in sorted order.

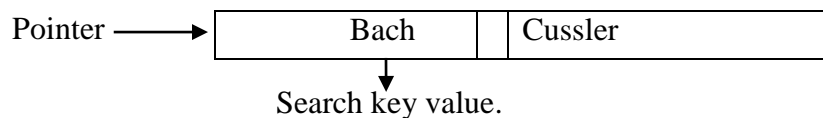
P_1	K_1	P_2	K_2	P_{n-1}	K_{n-1}	p_n
-------	-------	-------	-------	-------	-----------	-----------	-------

Pointers (p_1, p_2, \dots, p_{n-1}) points either to a file record with search key values (k_1, k_2, \dots, k_{n-1}) or to a bucket of pointers each of which pointers to a file record with search key value (k_1, k_2, \dots, k_{n-1}). The Bucket structure is used only if the search key does not form primary key and if the file is not stored in the search key value order. Pointer p_n is used to chain together the leaf nodes in the search key order thus allowing efficient sequential processing of the file.

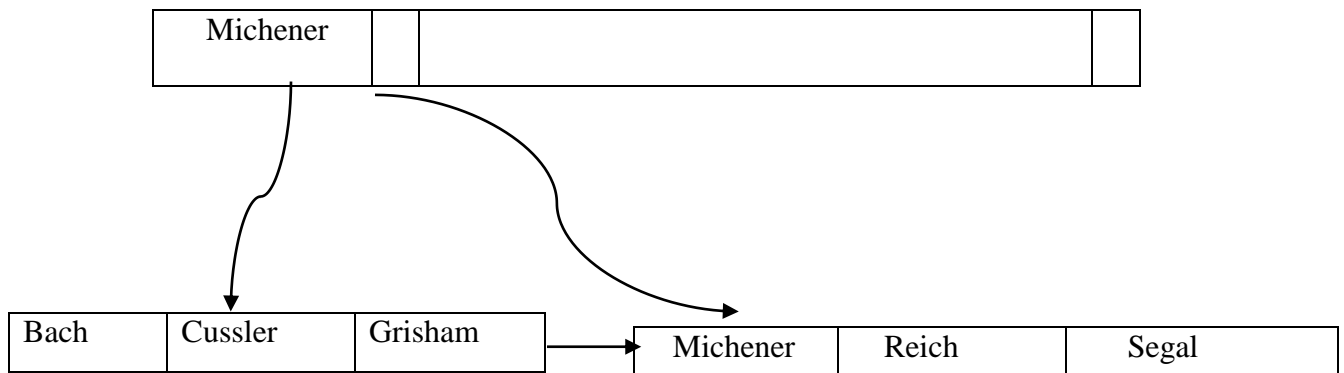


B+- tree for the book file (n=3)

For simplicity, we have omitted both the pointers to the file itself and the null pointers. Since the book file is ordered by the another name, the pointers in the leaf node point directly to the records in the file. The pointers in the bucket point, to a file record with the search key value of the pointer. Each leaf can hold up to $n-1$ values. The minimum value that a leaf can contains is $(n-1)/2$. The ranges of values in each leaf do not overlap. The last pointer is used for a special purpose. Since there is a linear order on the leaves based on the search key values that they contain we use the last pointer in each leaf node to chain together the leaf node to chain together the leaf node in the search key order.



The non-leaf nodes of the B+- tree form a multilevel sparse index on the leaf nodes. Structure of non-leaf node is the same as the leaf node, except that all pointers are pointers to tree nodes. A non-leaf node may hold up to 'n' pointers but most hold at least $n/2$ pointers. The number of pointers in a node is called the fan out of the node.



‘B’ in the B+- tree stands for “BALANCED”. It is the balanced property of B+- trees that ensure good performance for backup (queuing), insertion and deletion of records.

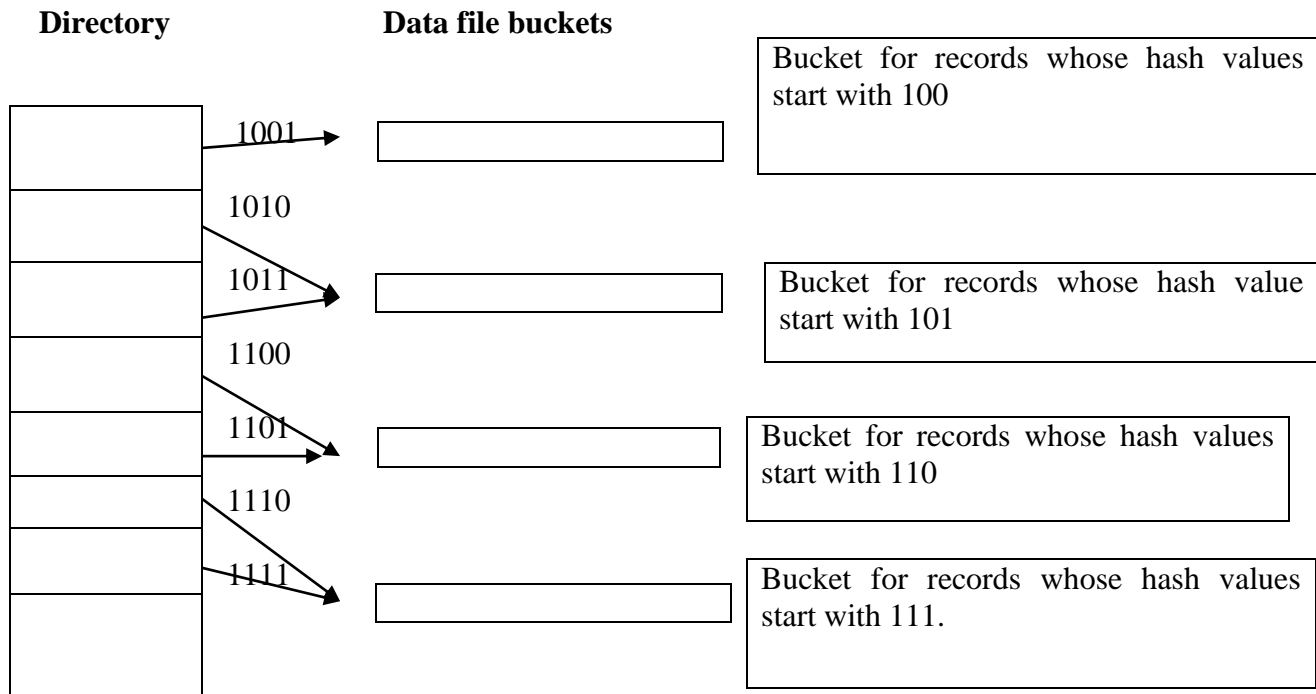
66. Explain Dynamic Hashing

A major drawback of static hashing is that the hash address space is fixed. Hence, it is difficult to dynamically expand or shrink the file depending on the number of records. There are many hashing schemes that allow the dynamic expansions and shrinking of files. Some of them are extensible hashing, linear hashing and so on. Extensible hashing stores an access structures in addition to the file and hence are somewhat similar to indexing. Main difference is that the access structure is based on the values that result after application of hash function to the search field. The extensible hashing schemes take advantage of the fact that the result of applying a hashing function is a non-negative integer and hence can be represented and a binary number. The access structure is built on the binary representation of the hashing function result, which is a string of bits. Linear hashing does not require additional access structure.

67. Explain Extensible Hashing?

In this hashing schemes type of directory an array of 2^d bucket addresses is maintained. d is called the global depth of the directory. The integer value corresponding to the first d bits of the hash value is used as an index to the array to determine a directory entry and the address in that entry determines that bucket in which the corresponding records are stored. There is no need for having a distinct bucket for each of the 2^d directory locations. Several directory locations with the same first ‘ d ’ bits for their hash values may contain the same buck address. If all the records that hash to these locations fit in a single bucket. ‘ d ’ is called the local depth. The value of global depth (d) can be increased or decreased by one at time thus doubling or having the number of entries in the directory away.

Doubling is needed if a bucket whose local is equal to the global depth ($d=1$) overflow.



Most record retrieval require two block accesses one to the directory and one to the bucket. The main advantage is extensible hashing is that the performance of the file does not degrade as the file grows. Another advantage is that no space is allocated for future growth, but additional buckets can be allocated dynamically as and when needed.

The space overhead for the directory table is negligible. Maximum directory size is 2^k , k is the number of bits in the hash value. This involves two block accesses instead of one in the case of static hashing.

68. Explain Linear Hashing.

The objective of this hashing technique is to allow a hash files to expand and shrink its number of buckets dynamically without needing a directory.

Suppose a file has ‘ m ’ buckets $(0,1,2,3,4,\dots,m-1)$ and users mode hashing function $h(k)=k \bmod m$. This hash function is called the initial hash function h ;

Overflow because of collisions handled by maintaining individual overflow chains for each bucket. However, when a collision leads to an overflow record in any file bucket, the first bucket in the file (bucket 0) is split in to two bucket the original bucket 0 and a new bucket ‘ m ’ at the end of the file.

Records originally in bucket 0 are distributed between the two bucket based on a different hashing functions.

$$H_{i+1}(k) = k \bmod 2m$$

This key property of the two hash functions h_i and h_{i+1} is that any record that hashed to bucket 0 based on h_i will hash to either bucket 0 or bucket m based on h_{i+1} .

If enough overflow occurs, all the original buckets 0,1,2,3,...m-1 will have been split and the file will now have 2m buckets instead of 'm' buckets and all buckets was the hash functions 'k' mod 2m. Hence, the records in overflow are eventually distributed in to regular buckets using hash function 'k' mod '2m' and a delayed split of their buckets.

After all the original buckets have been split and then hash functions k mod 2m has been applied to all records in the file, if a new collision that cause overflow occurs, we use a new hash functions.

$$H_i + 2(k) = k \text{ mode } 4m$$

This process is continued and a sequence of hashing function.

$$H_i + 3(k) = k \text{ mode } 8m,$$

$$H_i + 4(k) = k \text{ mode } 16m,$$

and so on are used.

Linear hashing is better than extensible hashing in the sense that it does not require a directory for implementation.

Main advantage of linear hashing is the complexity in implementing the splitting and organization of the buckets and the application of the new hashing function after each split.

69.Explain Interface Indexing

Suppose there is a sample data as shown in Table 4.1. It consists of information about four titles. Each title has a TitleId, Tname, Price, Discount, TextCat. Each title has a unique identification.

Table 4.1 A Sample Data

TitleId	Tname	Price	Discount	TextCat
T1	Oracle	399	15%	RDBMS
T2	C	299	10%	PL
T3	C++	199	10%	PL
T4	Sybase	100	10%	RDBMS

We shall consider how the data can be represented in several ways in the storage at the stored record interface. The simple data structure consists of a single stored file with all the records at one place. Suppose we have 1,000 records as shown in Table 4.1 (actually 4 records shown). If each TextCat field take 5 bytes, we would store 5000 bytes for TextCat. On the other hand we assume that TextCat is a pointer taking one byte, then we can store this pointer along with 1000 records. It will take 1000 bytes. Suppose we create another table Text Category, which has 2 records with structures as shown in Table 4.2(a) and 4.2(b).

Table 4.2 (a) Text category table

Table 4.2 (b) Title table

TextCate Pointer	TextCate	TitleId	Tname	Price	Discount	Textcat	Pointer
1	PL	T1	Oracle	399	15%		2
2	RDBMS	T1	C	299	10%		1
-	-	T3	C++	199	10%		1
-	-	T4	Sybase	100	10%		2

The size of text category table would be $(4+1)*2=10$ whereas size taken by Textcat pointer in table title is $1*1000=1000$ bytes. So overall space required will be $1000+10=1010$ bytes. If the single table is split into two tables you save in storage.

On the other hand if someone wants to know all the characteristics of a particular TitleId, say T1 then he needs to access two tables. Therefore, the number of accesses increase of data is split into two tables. Suppose the query is to find “All the TitleIds with Textcat Pointer = 1”, then the number of access will be 2.

Suppose, there is a query “find all TitleIds under category” ‘Programming Language’ (PL). Here too, we continue to use two files: one a Text Category file, and the other, a title file, but the pointers are in reverse direction, i.e. from text Category Table to title table. In other words, the text category table contains pointers to all the titleIds.

This representation is better for answering queries like “Find all TitleIds under a given Text category”. But it is not good to answer all attributes of a given TitleId.

Consider Tables 4.3(a) and 4.3(b), where the storage requirement is same. The Text Category File acts as an index to the Text File. The aim of index file is to define an access path for the Title File.

Table 4.3(a) Text Category File

<i>TextCat</i>	<i>Title Pointers</i>
PL	T2,T3
RDBMS	T1,T4

Table 4.3(b) Title File

<i>TitleId</i>	<i>Tname</i>	<i>Price</i>	<i>Discount</i>
T1	Oracle	399	15
T2	C	299	10
T3	C++	199	10
T4	Sybase	100	10

An index file has a record that contains a data value (in this case data value for text category field) and a pointer (i.e. a pointer for TitleId). The fields of index file are: TextCat (known as indexed field) and the pointers, pointing towards the records of the file it is indexing, i.e. pointing to the record of indexing (Title File). An indexed file (i.e. Text Category) can be used in the two ways: firstly, the user can access it sequentially until the matching TextCat, say, PL is not found. Secondly, user can directly access the individual record, say, TextCat = PL, in the indexed files.

The advantage of indexing is that it gives fast retrieval; the disadvantage is that it slows down update.

The TextCat is an index, which is managed by the DBMS. It is called as a dense secondary index. The term ‘dense’ indicates that the indexing file, Text Category, contains an entry for each and every record stored in the indexed file, Title. Moreover, the indexed file, does not include the field on which indexing is done, i.e. Title does not include field, TextCat(refer Tables

4.3(a) and 4.3(b)). The term “Secondary” means that the index is on a field other than the primary key.

Here we have done indexing on just one field. In an extreme case, we can do indexing on each field of file. Title, viz. Tname, TextCat, and Discount. Such an indexing is called *inverted file organization*. (See Table 4.4(a), (b), (c) and (d).)

Table (a)		Table (b)		Table (c)		Table (d)
Tname	Index	TextCat	Index	Discount	Index	Title
Tname	Pointers	TextCat	Pointers	Discount	Pointers	TitleId
Oracle	T1	PL	T2,T3	15	T1	T1
C	T2	RDBMS	T1,T4	10	T2,T3,T4	T2
C++	T3					T3
Sybase	T4					T4

Inverted file organization

This is good for queries where requests like “Find all TitleId’s for a given individual property” are to be answered. It is not good where all the properties of a given TitleId is to be answered. In real life, the normal organization as shown in Tables 4.3(a) and 4.3(b) along with secondary indexes on some fields is used.

Hierarchical organization

Under this file organization, we shall have a file consisting of two records—one for each TextCat. The number of records under each TextCat can vary. In this case under TextCat, PL, there are 2 records and under RDBMS also, there are two records. (See figure 4.1)

PL				RDBMS			
T2	C	299	10	T1	Oracle	399	15
T3	C++	199	10	T4	Sybase	100	10

Hierarchical organization.

A list of records below each TextCat (e.g. under PL) is known as repeated group. The representation like this where each table record is broken into two parts: TextCat and rest of the attributes of record title represents association between a TextCat and its titles. If we look back at Tables 4.3(a) and 4.3(b) where TextCat indicates the secondary index for title file, we find it is nothing, but hierarchical file.

70. Write a short note on Hashing Scheme of File Organization.

Each record is placed at a location whose address is computed as some function of a value. Therefore, DBMS calculates the stored record address (SRA) and places the record at that location. Similarly, when this record is to be retrieved, the same calculation using hash function is performed.

Suppose the TitleIds are: T20, T20, T30, T40, and T50. Let the hash function be:
Stored Record Address (SRA)= Remainder on dividing numeric part of TitleId by 7
The SRAs are: 3, 6, 2, 5, 1
It means record T10 will be stored at SRA 3, T20 will be stored at SRA 6, and so on.

Theoretically, it is possible to use the “identity” hash function. The “identity” generates a sequential number consisting of desired number of digits to be used as primary key. Suppose the primary keys are: 100, 150, 200, 500, 700, 900. Although there are only six primary key values it will reserve the entire SRA ranging from 0-999. There would be a wastage of 994 SRA (1000 – 6 = 994). However, if we use some hash function, i.e. divide the generated “identity” number by 7 and use its remainder as SRA, we will always get a number in the range of 0-6. If a cushion of about 50% is to be kept for future growth then we will generate, say, about 9 primary key values. We can instead use another hash function where we shall divide the generated “identity” by 11 and use its remainder as SRA. By choosing the divisor as 11 we can generate 0-10 numbers. Generally divisor is selected as prime number.

The above example shows how hash function works and why it is necessary.

The main disadvantage of hashing scheme is that as the size of the data file increases, so does the number of collisions, meaning that the average access time also increases. The solution to the problem is given by R. Fagin, et al., in their paper.

The ratio of the space actually required by the records of a file to the actual space allocated for the file is known as *load factor*.

Suppose the number of records needed is 11 blocks of storage space; but we earmark (1.5) * 11 = 16.5 = 17 blocks to reduce the number of collisions, then, the load factor will be (11/17) * 100 = 65%.

A strategy for direct storage of records should have low search time and a few collisions. This can be done by choosing low load factors (large blocking factors- 1.5 times in the example we have taken) and by selecting a hashing algorithm that distributes record uniformly over the storage area.

Extended hashing technique which is a variation of basic hashing technique guarantees that no more than two secondary storage accesses are even needed to locate the record for a given key. Its may be described as follows:

Suppose the basic function is h, and the primary key value of some record R is P. Hashing P, i.e. calculating h(p) gives P' which is called the pseudokey of P. Pseudokeys are not interpreted as addresses as they are in the case of hashing technique. Instead they are used to store records in some indirect methods as described.

The data blocks are addresses, which contain the actual stored records. A data block contains multiple records. One of the shortcomings of hash-addressing scheme is the problem of collisions. A collision occurs when the storage location for more than one record generated via hash function is same. Suppose there are titleIds T80 and T157 then hash function dividing the numeric part of TitleId by 11 and picking up its remainder will give same address location 3 for both of these records. Suppose all the address location from 0-10 are full then we move forward to find the first free address location. This will be stored at location 3 and to retrieve it back, the same process is repeated. But already a record at location 3. So how to store this record? Dynamic hashing scheme can be used to resolve this problem.

UNIT – III

1. *What is a tuple?*

In Relational Data structure terminology tuple is nothing but record.

2. *What do you mean by the degree of a relation?*

The number of attributes in a relation is called the degree of the relation.

3. *What is the cardinality of the relation?*

The number of tuples or rows in a relation is called the cardinality of the relation.

4. *What is a candidate key?*

A candidate key is an attribute that can uniquely identify a row in a table. Every relation has atleast one candidate key, because at least the combination of all its attributes has the uniqueness property.

5. *What is a Primary key?*

A Primary key is a column in the table whose purpose is to uniquely identify the record from the same table. One candidate key is designated as the primary key.

6. *What is an alternate key?*

Remaining candidate key except the primary key in a table are called alternate key.

7. *What do you mean by foreign key?*

A Primary key is a column in the table whose purpose is to uniquely identify the record from a different table. OR Foreign key is an attribute or combination of attribute of one relation (table).Also the foreign key and the primary key should be defined on the same underlying domain.

8. *What is the uniqueness property?*

At any given time , no two distinct tuples (rows) of R have the same value for A_i ,the same value for A_j and the same value for A_n .

9. *What are the constraints included in a relational model?*

Relational data model includes several types of constraints whose purpose is to maintain the accuracy and integrity of the data in the database. The major type of integrity constraints are

1. Domain Constraints
2. Entity Integrity
3. Referential Integrity
4. Operational constraints

10. What are domain constraints?

A domain is a set of values that may be assigned to an attribute. A domain definition usually consists of the following components.

1. Domain name
2. Meaning
3. Data Type
4. Size or length
5. Allowable values or allowable range

11. What is entity integrity?

The entity integrity rule is designed to assure that every relation has a primary key, and that the data values for that primary key are all valid. Entity integrity guarantees that every primary key attribute is non-null.

12. What is referential integrity?

A referential integrity constraint is a rule that maintains consistency among the rows of two tables (relation). The rule states that if there is a foreign key in one relation, either each foreign key value must match a primary key value in the other table or else the foreign key value must be null.

13. What is normalization?

Normalization is a process in which we analyze and decompose the complex relations and transform them into smaller, simpler and well –structured relations for validating and improving the logical design , so that the logical design satisfies certain constraints and avoid unnecessary duplication of data.

14. What are the two problematic issues in the design of relational database?

Two most problematic issues in the design of relational databases are

1. Repetition of Information (redundancy)
2. Inability to represent certain information.

15. What do you mean by determinant?

Determinant refers to the attribute or group of attributes on the left-hand side of the arrow of a functional dependency.

$Emp_no \rightarrow emp_name$

16. What is a trivial functional dependency?

A dependency is trivial , if and only if , the right hand side is a subset of the left-hand side(determinant).

Example: $emp_no, emp_name \rightarrow emp_name$

17. Explain the terms reflexivity, augmentation and transitivity.

Reflexivity: It states that a set of attributes always determines any of its subsets or itself. If B is a subset of A, then $A \rightarrow B$

Augmentation: It states that adding the same set of attributes to both the left and right hand sides of a dependency results in another valid dependency. If $A \rightarrow B$, then $A, C \rightarrow B, C$

Transitivity: It states that functional dependencies are transitive. If $A \rightarrow B$ and $B \rightarrow C$, then $A \rightarrow C$

18. Define Self Determination

Self determination states that a set of attributes always determines any of its subsets or itself. i.e $A \rightarrow A$

19. Define Decomposition.

Decomposition states that we can remove attributes from the right hand side of a dependency. If $A \rightarrow B, C$, then $A \rightarrow B$ and $A \rightarrow C$

20. Define Union

Union can combine a set of dependencies into single functional dependencies. If $A \rightarrow B$ and $A \rightarrow C$ then $A \rightarrow B, C$

21. What is a key?

A key uniquely identifies a row in a table.

22. What is intelligent key?

An intelligent key is based upon data values such as a date, a last name or a combination of values.

23. What is non-intelligent key?

A non-intelligent key is completely arbitrary, having no function or meaning other than identification of the row.

24. What are the different types of relations between the entities in a table?

There are three possible relationships between related entities or tables.

1. One-to-one relationship
2. One-to-many relationship (its mirror image is many-to-one)
3. Many-to-many relationship

25. What are intersection tables?

An intersection table contains two or more foreign keys, relating the primary key values of two or more tables to each other. The role of an intersection table is to convert the many-to many relationship into two one-to-many relationships that can be easily handled by the database.

26. What is transitive dependency?

A transitive dependency in a relation is a functional dependency between two or more non-key attributes.

27. What is first normal form (1NF)?

The multi-valued attributes called repeating groups should be removed, i.e elimination of repeating groups.

28. What is Second normal form (2NF)?

The partial functional dependencies have to be removed ,i.e elimination of redundant data.

29. What is third normal form (3NF)?

The transitive dependencies have to be removed, i.e elimination of columns not dependent on the key.

30. What is Boyce-Codd Normal form (BCNF)?

The remaining anomalies that results from functional dependencies are removed.

31. Write a short note on Normalization

The concept of relational database was first talked about by Dr .Edger F.CODD around 1970's, In IBM research publication are system R Relational around this time, he gave the concept of normalization as basic for data design.

He defined first, second and third Normal form depending upon the set of constraints which each Normal satisfies. Normalization is used to avoid redundancy and the is problems arising out of redundancy.

Codd's original definition of third normal form was found to be "weak" and needed "strength" . A strong revised formulation of 3NF was derived by "BOYCE" and "CODD". It was stronger than the originally defined 3NF, meaning any relation that was in 3NF as per in old definition was not necessary in 3NF as per new definition. The new form is known as BOYCE/CODD Normal form and is abbreviated as BCNF.

The concept of normalization created so much of interest among the contemporary computer researches that FAGIN name out with a new fourth normalized form.

Later, he went on to define, another normal form which is known as projection join Normal form (PJ/NF) OR fifth Normal form.

32. Mention the purpose of Normalization

- i) Minimize redundancy in data.
- ii) Remove insert, delete, update anomaly during database activities.
- iii)Reduce the need to reorganize data when it is modified or enhanced.

33. Write a short note on function dependency

Author:

A.NO	A.NAME	A.QUALI	A.STATUS
1	Alexis leon	MCA ,Ph. D	Software consultant
2	Mathew leon	M.Sc ,Ph. D	Software developer
3	Rajesh Narang	M.Sc, Ph. D	Project manager
4	Balagurusamy	MCA,M.Tech,Ph. D	Database administrator

A.A → A.A(NAME,QUALIFICATION,STATUS)

A.NO ← DEPENDENT



A → NUMBER

B → NAME , QUALIFICATION , STATUS.



Determinant Dependent.

34. Explain first normal form with suitable example(1)

A relation is called is first normal form 1NF if it contains no repeating groups.

DATA IN UNNORMALIZED FORM

A#	A.NAME	A.QUAL	A.STATUS	T.ID	T.NAME	PRICE	DISCOUNT	COPIES SOLD
90	Kapoor	Ph. D	10	T5	TCP/IP	299	20	700
100	Arora	Ph. D	10	T1	ORACLE	399	15	800
100	Arora	Ph. D	10	T2	C	299	10	900
110	Arora	M. Tech	20	T3	C++	199	10	700
120	Sharma	MCA	30	T4	SYBASE	100	10	600
120	Basu	MCA	30	T1	ORACLE	399	15	700

The above table its not in first normal form as it contains repeated groups. The repeating columns on attribute or removed from the above table and put in to a new table. It is called us

parent table. The rest of the attributes not covered in parent table or put in another table. It is known as derived table.

PARENT TABLE (AUTHOR)

A#	A.NAME	A.QUALIFICATION	A.STATUS
90	Kapoor	Ph. D	10
100	Arora	Ph. D	10
110	Sharma	MCA	20
120	Basu	MCA	20

Definition:-1NF

Repeated group of elements should be removed.

A#	A.NAME	A.QUAL	A.STATUS	T.ID	T.NAME	PRICE	DISCOUNT	COPIES SOLD
90	Kapoor	Ph. D	10	T5	TCP/IP	299	20	700
100	Arora	Ph. D	10	T1	ORACLE	399	15	800
100	Arora	Ph. D	10	T2	C	299	10	900
110	Arora	M. Tech	20	T3	C++	199	10	700
120	Sharma	MCA	30	T4	SYBASE	100	10	600
120	Basu	MCA	30	T1	ORACLE	399	15	700

A relation capital are is in first normal form(1NF) if and only if all underlying domains contain atomic values only.

First normal form (1NF) eliminates repeating group by putting each into a separate table and connecting them with a one-to-many relationship.

Table author is in first normal form since A# in table author appears only once whereas in table author underscore title. A# comes repeatedly A#=100, A#=120, each comes twice as a result of this repetition following anomalies comes up.

ANOMALIES

a)Insert anomaly :-

Suppose A#=130 has written a title Id=T6, T.Name is equal to DBMS, price=599, discount=10%but it cannot be inserted in Author-title table unless one copy of T.name is equal to DBMS is sold.

b>Delete anomaly:-

If we delete a tuple in author-title table which has only one occurrence we case two pieces of information, one about title and another about number of copies sold for a title Id.

Table the case of 4th tuple in Author-title table, if we delete A#=110 then we loose complete information Title Id=T3 along with number of copies sold for it.

c)Update anomaly:-

There is redundancy as for as T.name is concerned in table Author-Title.

Example:-

T.name for Title Id=T1 which is oracle occurs at two places. If for Title Id=T1. The discount is increased from fifteen to 20%,we have to search for each and every occurrence of title Id=T1 in table Author-title.

* Even if a single occurrence is remains unchanged for title Id=T1 in either tuple we will have a discount has 15 at one place and 20 at another. This will lead to inconsistency.

*The solution to this problem is to split the relation Author-Title into two relations.

- i) Title (Title Id, Title name, Price, Discount)
- ii) Author-Title(A#, Title Id, copies sold)

We will do it in second normal form to redress this anomalies.

First Normal Form definition(2nd example)

A database is in first normal form if it satisfies the following conditions:

- * Contains only atomic values.
- * There are no repeating groups.

An atomic value is a value that cannot be divided.

Example:

In the table shown below the values in the [color] column in the first row can be divided into “red” and “green”, hence table-product is not in 1NF.

A repeating group means that a table contains two or more columns that are closely related.

Example:

A tables that records data on a book and its author(s) with the following columns:

[Books ID, Author 1, Author 2 and Author3 is not in 1NF] because Author1, Author2 and Author3 are all repeating the same attribute

First Normal Form Example(2):

Table-Product

Product Id	Color	Price
1	Red	15.99
2	Yellow	23.99
3	Green	17.50
4	Yellow, Blue	9.99
5	Red	29.99

The above table is not in first normal form because the [color] column can contain multiple values.

Example:

The first row includes values “red” and “green”.

To bring this table to first normal form, we split the table into two tables and now we have the resulting tables.

Table- Product – Price

Product Id	Price
1	15.99
2	23.99
3	17.50
4	9.99
5	29.99

Table – Product – Color

Product Id	Color
1	Red
2	Yellow
3	Green
4	Yellow, Blue
5	Red

Now first normal form is satisfied, as the columns on each table all hold just one value.

35. Explain Second Normal Form with suitable example.

Definition

A database is in second normal form if it satisfies the following condition.

* It is in first normal form.

* All non-key attributes are fully functionally dependent on the primary key.

In a table if attribute B is functionally dependent on A but is not functionally dependent on a proper subset on A, then B is considered fully functionally dependent on A. Hence, in a 2NF table, all non-key attributes cannot be dependent on a subset of the primary key.

Note that if the primary is not a composite key, all non key attributes are always fully functional dependent on the primary key. A table that is in first normal form and contains only a single key as the primary key is automatically in second normal form.

Example

Consider the following example:-

Table- Purchase _ detail

Customer Id	Store Id	Purchase Location
1	1	Los Angles
1	3	San Francisco
2	1	Los Angles
3	2	New York
4	3	San Francisco

The above table has a composite primary key [customer id, store id]. The non-key attribute is [purchase Location]. In this case, [purchase location] only depends on [store id], which is only part of the primary key. Therefore this table does not satisfy second normal form.

To bring this table to second normal form, we break the table into two tables and now we have the following.

Table – Purchase

Customer Id	Store Id
1	1
1	3
2	1
3	2
4	3

Table – Store

Store Id	Purchase
1	Los Angles
2	New York
3	San Francisco

What we have done is to remove the partial functional dependency that we initially have. Now, in the table [table- store], the column [purchase location] is fully dependent on the primary key of that table, which is store Id.

36. Explain Third Normal Form Definition

A database is in third normal form if it satisfies the following condition.

- i) It is in second normal form.
- ii) There is no transitive functional independency.

By transitive functional dependency we have the following relationship in the table A is functionally dependent on B, and B is functional dependent on C. In this case C is transitively dependent on A via B.

Example:

Consider the following example

Table – Book – Detail

Book-Id	Gender-Id	Gender- Type	Price
1	1	Gardening	25.99
2	3	Sports	14.99
3	1	Gardening	10.00
4	2	Travel	12.99
5	3	Sports	17.99

In the above table [book-id] determines [gender-id] and [gender] determines [gender-type] via [gender-id] and we have transitive functional dependency and this structure does not satisfies the third normal form.

To bring this table to the third normal form, we split the table into two as follows.

Table – Book

Book-Id	Gender-Id	Price
1	1	25.99
2	3	14.99
3	1	10.00
4	2	12.99
5	3	17.99

Table – Gen

Gender-Id	Gender- type
1	Gardening
2	Sports
3	Gardening

Now all non-key attributes are fully functionally dependent only on the primary key In [table-book], both [gender-id] and [price] are only dependent on [book-id].

In table [table-gender], [gender- type] is only dependent on [gender id].


37. Explain Boyce-Codd Normal Form(BCNF) with suitable example.

A database table is set to see in BCNF if in is in 3NF and contains each and every determinant as a candidate key. The process of converting the table into BCNF is as follows:

- * Remove the nontrivial functional dependency.
- * Make separate table for the determinates.

BCNF of the below table is as follows:

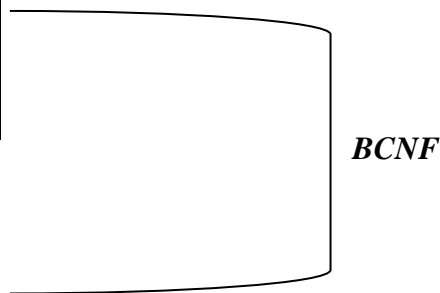
<i>Supplier ID</i>	<i>Supplier Name</i>	<i>Product ID</i>	<i>Quality</i>
<i>S001</i>	<i>Mr.X</i>	<i>P001</i>	<i>120</i>
<i>S002</i>	<i>Mr.Y</i>	<i>P002</i>	<i>102</i>
<i>S003</i>	<i>Mr.Z</i>	<i>P001</i>	<i>100</i>



Determinants

Supplier-Id	Product-Id	Quantity
S001	P001	120
S002	P002	102
S003	P003	100

Supplier Id	Supplier Name
S001	Mr. X
S002	Mr. Y
S003	Mr. Z



UNIT-IV

1. *What is SQL?*

Structured Query Language is the standard command set used to communicate with the relational database management systems.

2. *Why is it important to have a good understanding of the SQL?*

SQL code, irrespective of whether it is written or generated, should be carefully examined, verified, tuned, and optimized. So a fundamental and thorough understanding of SQL is an absolute necessity for software professionals and end users who are in the business of creating, maintaining, and using data and database management systems.

3. *What are the characteristics of SQL?*

SQL usage is extremely flexible. It uses a free form syntax that gives the user the ability to structure SQL statements in a way best suited to him. Each SQL is parsed by the RDBMS before execution, to check for proper syntax and to optimize the request. Unlike certain programming languages, there is no need to start SQL statements in a particular column or be finished in a single line. The same SQL request can be written in a variety of ways.

4. *What are SQL operators?*

Operators and conditions are used to perform operations such as addition, subtraction or comparison on the data items in an SQL statement. Operators are represented by single characters or reserved words. A condition is an expression of several operators or expressions that evaluates to True, False or Unknown.

5. *What is the difference between unary and binary arithmetic operators?*

The unary operator operates on only one operand. Example -10.
The binary operator operates on two operands. Example 1+2

6. *What are Data Definition Language (DDL) commands? Explain with examples*

Data Definition Language is used to create, alter, and delete database objects. The commands used are CREATE, ALTER, and DROP. The principal data definition statements are CREATE TABLE, CREATE VIEW, CREATE INDEX, ALTER TABLE, DROP TABLE, DROP VIEW, and DROP INDEX.

7. *What are Data Manipulation Language (DML) commands? Explain with examples*

Data Manipulation language commands let users insert, modify and delete the data in the database. SQL provides three data manipulation statements - INSERT, UPDATE, DELETE.

Example 1: *INSERT*

```
INTO BOOKV ( ID, TITLE, PUBLISHER, YEAR, PRICE)  
VALUES ( "CO2", "COUNTDOWN 2000", "TMH", 1997, 295);
```

Example 2: *UPDATE BOOK*
*SET PRICE = PRICE * 1.12*
WHERE PUBLISHER = " McGraw-Hill"
AND YEAR > 1995;

Example 3: *DELETE*
FROM BOOKV
WHERE PUBLISHER = " McGraw-Hill"

8. ***What are Data Control Language Command? Explain with example.***

The data Control language consists of commands that control the user access to the database objects. Thus DCL is mainly related to the security issues ,i.e, determining who has access to the database objects and what operations that can perform on them. The DCL commands are GRANT and REVOKE. The user should get permission from the DBA explicitly to access the database objects using the Grant Command. Revoke command is used to take away a privilege that was granted.

Example 1: *GRANT SELECT,DELETE,UPDATE*
ON CATALOG
TO RIYAZ;

Example 2: *REVOKE SELECT*
ON CATALOG
FROM RIYAZ;

REVOKE UPDATE
ON CATALOG
FROM PUBLIC;

REVOKE DELETE
ON CATALOG
FROM RIYAZ;

9. ***What are data query language (DQL) command? Explain with examples***

This is one of the most commonly used SQL statements. This SQL statement enables the users to query one or more tables to get the information they want. SQL has only one data query statement- SELECT.

Example: *SELECT AUTHOR*
FROM CATALOG
WHERE PRICE= 40;

10. What are data administration statements(DAS)?

Data administration commands allow the user to perform audits and analysis on operations within the database. They are also used to analyze the performance of the system. Two data administration commands are START AUDIT and STOP AUDIT.

11. What are Transaction Control statements (TCS) ? Explain with examples.

Transaction Control statements are statements , which manage all the changes made by the DML statements. Some of the Transaction control statements are COMMIT, ROLLBACK, SAVEPOINT, SET TRANSACTION.

Example 1: *COMMIT [WORK];*

Example 2: *ROLLBACK [WORK];*

Example 3: *SAVEPOINT SP1*

```
DELETE FROM EMPLOYEE WHERE EMPNO = 100;  
SAVEPOINT SP2  
DELETE FROM EMPLOYEE WHERE EMPNO = 110;  
SAVEPOINT SP3  
DELETE FROM EMPLOYEE WHERE EMPNO = 121;  
SAVEPOINT SP4  
DELETE FROM EMPLOYEE WHERE EMPNO = 101;  
SAVEPOINT SP5  
DELETE FROM EMPLOYEE WHERE EMPNO = 111;
```

12. What is a database table?

Tables are the basic building blocks in any relational database management systems. They contain rows and columns of data.

13. What are views?

A view is a named table that is represented not by its own physically separate stored data, but by its definition in terms of other named tables (base tables or views).

14. How are views updated? What are the restrictions in updating the view?

All views are not updateable. That is INSERT, UPDATE and DELETE operations cannot be performed on all views. Some of the restrictions in updating the views are:

1. It must be derived from a single base table.
2. If a column of a view is derived from an aggregate function , then the view is not updateable.
3. A view defined on a non-updateable view is not updateable.
4. If the definition of the views involves either a GROUP BY or a HAVING clause and DISTINCT at the outermost level, then the view is not updateable.
5. If the FROM clause in the view definition involves multiple range variables, then it is not updateable.

15. How are views deleted?

If you want to delete or remove an existing view you can use the DROP VIEW statement. The syntax of DROP VIEW statement is

DROP VIEW view-name { RESTRICT/CASCADE}

16. What is the syntax of the CREATE TABLE command?

Syntax 1:

*CREATE TABLE base-table-name
([Column -1-definition], [Column -2-definition], [Column - n – definition],
[Primary-key-definition] , [Alternate-key-definition], [Foreign-key-definition]);*

Syntax 2:

CREATE TABLE base-table-name LIKE table-name;

17. How are tables deleted?

An existing base table can be deleted at any time by using the DROP TABLE statement. The syntax of this statement is

DROP TABLE base -table-name;

18. How are views different from base tables?

Base tables are autonomous tables. Views are not autonomous. View do not contain any data , instead view is a virtual table , deriving its data from base tables.

19. What is the use of the WITH CHECK OPTION clause?

If the clause ‘WITH CHECK OPTION ‘is included in the view-definition, then all inserts and updates on that view will be checked to ensure that the newly inserted or updated .Rows do not violate the view-defining conditions.

20. What is the use of SELECT statement?

The SELECT statement lists the items – column names, computed values, the aggregate functions etc – to be retrieved. The FROM clause specifies the table or tables from where the data has to be retrieved.

Example: *SELECT TITLE,AUTHOR,PUBLISHER
FROM BOOK;*

21. What do you mean by closure property?

The result of a SELECT statement is another table is referred to as the closure property of the relational systems.

22. *What do you mean by SELECT *? Explain with the help of an example.*

“ * “ is used to get all the columns of a particular table. For example the SQL, SELECT * FROM BOOK will give an entire copy of the table BOOK.

```
SELECT *  
FROM BOOK;  is the same as
```

```
SELECT ID, TITLE , AUTHOR, PUBLISHER, YEAR, PRICE  
FROM BOOK;
```

23. *How will you eliminate duplicates from the result of a SELECT statement?*

When you give a SELECT statement the RDBMS does not eliminate the duplicates from the result. To avoid this , you have to use the keyword DISTINCT. DISTINCT eliminates the duplicates from the result set.

```
SELECT DISTINCT PUBLISHER  
FROM BOOK;
```

24. *How can we get computed values using the SELECT statement? Explain with example.*

The SQL statements can be used for retrieving the computed values without any problems. For example, consider the following query ‘ For all books get the price in US dollars (the price given in the table being in Indian rupees and 1 dollar = 47 rupees). We can do this as follows.

```
SELECT TITLE, AUTHOR , PRICE / 47 AS PRICE_D  
FROM BOOK  
WHERE PRICE > 400;
```

25. *How is it possible to select null values? Give example*

For retrieving rows where some of the columns have been defined as NULLS, there is a special comparison operator of the form IS [NOT]NULL. For example, suppose the PRICE of book is not known and hence , kept as NULL. The SQL is

```
SELECT *  
FROM BOOK  
WHERE PRICE IS NULL;
```

26. *What is the purpose of the GROUP BY clause in the SELECT statement?*

The GROUP BY clause specifies a summary query. This is usually used with aggregate functions like SUM, AVG, MAX, MIN etc. For example , the statement ‘...GROUP BY PUBLISHER...’ will group the result set based on the publisher’s name.

27. *What is the purpose of the ORDER BY clause in the SELECT statement?*

The ORDER BY clause sorts or orders the results based on the data in one or more columns in the ascending or descending order. For example, ‘.....ORDER BY PRICE

DESC.....'will produce a result set where the items are shown in the descending order of price. If the ORDER BY clause is omitted, the result set is not sorted.

28. What is the purpose of the HAVING BY clause in the SELECT statement?

The HAVING clause tells SQL to include only certain groups produced by the GROUP BY clause in the query result set. HAVING clause is the equivalent of the WHERE clause and is used to specify the search criteria or search condition when GROUP BY clause is specified.

29. What is the difference between the IN and BETWEEN clauses?

The IN clause is used to get the certain value and BETWEEN clause is used to get those items that fall within the range.

Example 1:

```
SELECT TITLE, AUTHOR, PUBLISHER, YEAR
FROM BOOK
WHERE YEAR IN ( 1999, 2000);
```

Example 2:

```
SELECT TITLE, AUTHOR, PUBLISHER, PRICE
FROM BOOK
WHERE PRICE BETWEEN 400 AND 600;
```

30. What is the use of the LIKE clause in a SELECT statement? Explain with example

LIKE is a very powerful clause and also very useful. For example, if you want to get all the details of the books whose publisher's name starts with 'S', use LIKE as follows

```
SELECT TITLE, AUTHOR, PUBLISHER
FROM BOOK
WHERE PUBLISHER LIKE "S%";
```

Here, % stands for any sequence of n characters.

31. What is the use of the ESCAPE clause? Explain with example

If the ESCAPE clause and a character is specified it means that, the special interpretation given to the literal characters '_' and '%' can be disabled. In the following example, the backslash '\' is specified as the ESCAPE character, which means that the special interpretation given to '_' and '%' can be disabled by preceding such characters with a backslash. So the query

```
SELECT TITLE, AUTHOR, PUBLISHER
FROM BOOK
WHERE PUBLISHER LIKE "\_%" ESCAPE "\";
```

Will return any publisher name with an underscore(_) in it.

32. What is a Subquery?

Subqueries are queries that appear within the WHERE or HAVING clause of another SQL statement. Subqueries provide an easy and efficient way to handle queries that require the results from another query. Subqueries allow you to combine these two queries so that you can get the results using a single SQL statement as follows:

```
SELECT *  
FROM MEMBER  
WHERE MEMBER_ID IN ( SELECT DISTINCT  
MEMBER_ID FROM ORDER_SUMMARY);
```

Note that the keyword DISTINCT is used in the subquery to eliminate the duplicate distributor's IDs. Subqueries always appear as part of the WHERE clause or HAVING clause. The subqueries are almost identical with the ordinary SELECT statement, but there are a few differences.

1. A subquery must produce a single column of data as its result. The subquery can have only a single select item in its SELECT clause. So you cannot use a 'SELECT *' in a subquery unless the table you are referring has only one column.
2. The ORDER BY clause cannot be specified in a subquery. Since the results of the subquery are used internally and are not displayed to the user, ordering does not make much sense. You can specify the ORDER BY clause in the main query to order the results.
3. A subquery cannot be a UNION, only a single SELECT statement is allowed.

33. What are aggregate functions or column functions?

Aggregate functions allow you to summarize the data from the tables. An aggregate function takes an entire column of data as its argument and produces a single data item that summarizes the column.

34. What are aggregate functions used for?

Aggregate functions are used to examine the following information requests:

1. What is the total number of employees in a particular department and in the organization?
2. What is the average salary of an employee and also for the company?
3. What is the minimum salary in the organization?
4. What is the name of the employee who draws the maximum salary in the company?
5. What is the name of the employee who draws the minimum salary in the company?
6. What are the names of the employees whose salary is greater than the average?

35. Name the most commonly used aggregate functions.

The most commonly used aggregate functions provided by SQL are COUNT(), COUNT(*), SUM(), AVG(), MAX() and MIN().

36. What are the rules to be followed when using aggregate functions?

The rules to be followed when using aggregate functions are

1. For SUM and AVG the argument must be of type numeric.
2. Except for the special case COUNT (*), the argument may be preceded by the key word DISTINCT to eliminate the duplicate rows before the functions is applied to a column. The alternative to DISTINCT is ALL , which is the result. The DISTINCT is legal for MAX and MIN but meaningless.
3. The special function COUNT (*)which is used to all rows without any duplicate elimination and so the keyword DISTINCT is not allowed for this function.
4. The argument cannot involve any aggregate function references or table expressions at any level of nesting. For example , the SQL'SELECT AVG (MIN (QTY))AS AVERAGE' is illegal.
5. Any NULL in the column is eliminated before the function is applied , regardless of whether DISTINCT is specified or not , except in the case of COUNT(*), where nulls are handled like normal values.

37. What is the difference between COUNT() and COUNT(*)?

COUNT() is used to count the number of values in a column. COUNT(*)is used to count the number of rows of the query results.

38. What is the use of the SUM()?Explain with examples

SUM() is used to find the sum of the values in a column.

Find the total basic pay for all the employees in the organization.

```
SELECT SUM (BASIC)
FROM EMPLOYEE;
```

39. What is the use of the AVG()?Explain with examples

AVG() is used to find the average of the values in a column.

Find the average HRA of an employee.

```
SELECT AVG (HRA)
FROM EMPLOYEE;
```

40. What is the use of the MAX() and MIN()?Explain with examples

MAX() is used for finding the maximum value in a column.MIN () is used for finding the minimum value in a column.

Get the department ID, the average, maximum and minimum basic pay of all the departments.

```
SELECT DEPTID, AVG(BASIC), MAX(BASIC),MIN(BASIC)
FROM EMPLOYEE
GROUP BY DEPTID;
```

TABLES USED IN THIS CHAPTER

MEMBER TABLE

ID	NAME	ADDRESS	CITY	STATE	PIN	PHONE	E MAIL
C01	M..Leon	Noel Heritage,Thrikkakara,	cochin	kerala	682021	423981	mathews@ini.net
C02	P.Ram	Nelson chambers,Aminjikarai	Madras	Tamil Nadu	600029	3743062	pram@vsnl.com
C03	S.Gupta	Ramon House,Churchgate	Mumbai	Maharashtra	400020	4998101	sgupta@usa.net
C04	A.Verma	500,Kala Mansion,S.D,Road	Secunderabad	Andhra Pradesh	500003	2461255	av@hotmail.com
C05	R.Mishra	Ansari Road,Daryaganj	New Delhi	New Delhi	110002	5819378	rm@yahoo.com

CATEGORY TABLE

Category ID	Description
BS	Business
CS	Computer science
FN	Fiction
GR	General

ORDER_DETAILS_TABLE

ORDER_NO	BOOK_ID	QUANTITY
11096733	P1	1
11096733	B2	1
11096733	C1	1
11364223	A1	1
11364223	D3	2
11364223	E1	1
11364223	H1	1
13976453	I3	1
26970379	E1	10
26970379	P2	2
26970379	T1	1
26970379	Z1	1
26970379	E1	3
26971454	E1	15
26971454	I3	10
36532264	M3	1
36532264	M4	1
36532264	B5	1

36532264	B6	1
36552341	J1	3
36552341	K1	4
55946511	J2	2
55946511	E1	2
55946511	D4	2
86464430	L1	1
86464430	E3	1
86464430	C5	1
86464430	E1	1

PUBLISHER TABLE

Publisher_ID	Name	City	Country
AP	Abbeville press	Paris	France
AW	Addison-Wesley	Massachusetts	USA
AH	Artech house	Boston	USA
BB	Bantam books	London	UK
BP	Blackwell publisher	New York	USA
CP	Century	London	UK
CN	Clarkson	Kottayam	India
CB	Corgi books	New York	USA
DC	DC books	Kottayam	India
DP	Dell publisher	New York	USA
HC	Happer Collins	London	UK
HY	Hyperion	New York	USA
JW	John Wiley & Sons	New York	USA
LV	Leon Vikas	Madras	India
MG	McGraw-hill	New York	USA
PB	Penguin books	New Delhi	India
PH	Prentice hall ptr	New jersey	USA
RH	Random House	London	UK
SS	Simon & Schuster	New York	USA
TM	Tata McGraw-hill	New Delhi	India
WB	Warner book	New York	USA

ORDER_SUMMARY TABLE

Order No	Member ID	Order Date	Amount	Order Status
11096733	C01	25-Feb-00	2038.3	Shipped
11364223	C03	7-Apr-00	1237.5	Shipped
12806544	C05	30-Apr-00	159.5	Shipped
13976543	C03	25-Jul-00	3575	Cancelled
26970379	C01	25-Jul-00	3897.3	Shipped
26971454	C02	22-Aug-00	7012.5	Shipped

36532264	C05	22-Aug-00	4371.4	Shipped
36552341	C01	2-Feb-01	1793	Processing
55946511	C03	7-Feb-01	2131.8	Processing
86464430	C01	21-Feb-01	982.3	Processing

BOOK TABLE

ID	TITLE	AUTHOR	PUBLISHER	YEAR	PRICE
A01	A painted house	Grisham	Random House	2001	195.55
A02	Abduction	Cook	Pan Books	2000	360
A03	Airport	Hailey	Corgi Books	1968	175.45
B01	Biplane	Bach	Dell Books	1966	283.35
B02	Bloodline	Sheldon	Warner Books	1977	100.15
B03	Blue gold	Cussler	Simon & Schuster	2000	285
C01	Catch 22	Heller	Random House	1994	250.1
D01	Doctors	Segal	Bantam Books	1988	150
D02	Dragon	Cussler	Harper Collins	1990	123.55
F01	Flood tide	Cussler	Simon & Schuster	1997	414.5
H01	Hawaii	Michener	Mandarin Books	1959	124.5
H02	Hotel	Hailey	Corgi Books	1965	175.75
I01	Icon	Forsyth	Corgi Books	1996	182.35
I02	Illusions	Bach	Dell Books	1997	330
I03	Inca Gold	Cussler	Harper Collins	1994	127
I04	Invasion	Cook	Pan Books	1997	177.9
O01	One	Bach	Dell Books	1988	289
P01	Prizes	Segal	Bantam Books	1995	256.1
S01	Serpent	Cussler	Simon & Schuster	1999	532.8
S02	Sheba	Higgins	Signet Books	1995	145.8
T01	The Class	Segal	Bantam Books	1985	125
T02	The runner	Reich	Headline Books	2000	199.95
T03	The Simple truth	Baldacci	Simon & Schuster	1997	211
T04	Thunder point	Higgins	Signet Books	1993	95.65
T05	Time line	Crichton	Century Books	1999	623.3
V01	Vector	Cook	Macmillan	1999	424.8

CATALOG TABLE

Book ID	Title	Author ID	Publisher ID	Category ID	Year	Price
P1	201 Principles of software development	AD	MG	CS	1995	898
G1	A Guide to SCM	AL	AH	CS	2000	3715
A1	Airport	AH	CB	FN	1968	175
B1	Beachcombing at Miramar	RD	WB	GR	1996	423
B2	Beyond Soap, Water and comb	EM	AP	GR	1998	705

B3	Biplane	RB	DP	GR	1966	282
B4	Businesss	FE	CN	BS	1998	522
C1	Catch 22	JH	RH	FN	1994	250
C2	Computors for Everyone	AL	LV	CS	2001	200
C3	Countdown 2000	AL	TM	CS	1998	275
D1	Death march	EY	PH	CS	1997	890
D2	Digital Economy	DT	MG	CS	1996	538
D3	Doctors	ES	BB	FN	1988	150
D4	Dragon	CC	HC	FN	1990	123
E1	E-Biz primer	ML	TM	CS	2001	325
E2	ERP Demystified	AL	TM	BS	2000	397
F1	First You Have to Row a Little Boat	RD	WB	GR	1993	601
H1	Hotel	AH	CB	FN	1965	175
I1	Illusions	RB	DP	GR	1997	330
I2	Inca Gold	CC	HC	FN	1994	124
I3	Internet for Everyone	ML	LV	CS	1997	150
I4	Introduction to Computers	ML	LV	CS	1999	200
J1	Jesus CEO	LJ	HY	GR	1995	350
J2	Jesus in blue Jeans	LJ	HY	GR	1997	521
K1	Kane and Abel	JA	HC	FN	1979	145
M1	Management Mole	JM	CB	BS	1988	150
M2	Managing Technical People	WH	AW	CS	1997	593
M3	Managing the software Process	WH	AW	CS	1990	845
S1	Silicon Samural	TF	BP	BS	1993	250
B5	The Book of Business Wisdom	PK	JW	BS	1997	1318
B6	The Book of Leadership Wisdom	PK	JW	BS	1998	1309
C4	The class	ES	BB	FN	1985	145
C5	The Complete Adventures of Feluda	SR	PB	FN	2000	395
E3	The End of Imagination	AR	DC	GR	1998	48
H2	The Hunt for Red October	TC	HC	FN	1984	145
L1	The Legends of Khasak	OV	PB	FN	1994	125
M4	The Mythical Man Month	FB	AW	CS	1995	502
P2	The path	LJ	HY	GR	1996	545
T1	The Testament	JG	CP	FN	1999	590
Z1	Zen Computing	PS	SS	CS	1999	888

AUTHOR TABLE

Author ID	Names	City	Country
AD	Alan M.Davis	Colorado	USA
AL	Alexis	Cochin	India
AH	Arthur Hailey	Bahamas	Bahamas
AR	Arundhat Roy	New Delhi	India
CC	Clive	Arizona	USA
DT	Don Tapscott	New york	USA

EM	ED marquand	Seattle	USA
EY	Edward Yourdon	California	USA
ES	Erich Seagal	Harvard	USA
FE	Florence Isaacs	New york	USA
FB	Frederick P. Brooks	North Carolina	USA
JA	Jeffrey Archer	Cambridge	UK
JG	John Grisham	Virginia	USA
JM	John Mole	London	UK
JH	Joseph Heller	New York	USA
LJ	Laurie Beth Jones	Texas	USA
ML	Mathews Leon	Cochin	India
OV	O.V. Vijayan	Palghat	India
PK	Peter Krass	Connecticut	USA
PS	Philip Toshio Sudo	Hawaii	USA
RB	Richard Bach	Washington	USA
RD	Richard Bode	Denver	USA
SR	Satyajit Ray	Calcutta	India
TC	Tom Clancy	Maryland	USA
TF	Tom Forester	Queensland	Australia
WH	Watts S. Humphrey	Florida	USA

Note: The above tables is used in the below queries.

SQL> SELECT TITLE FROM CATALOGS WHERE PRICE > 507.8;

TITLE

201Principles of S/W Devp
 Airport
 A Guide to scm
 Beach at Miramar
 Beyond Soap,Water
 Biplane
 Business Notes
 Catch 22
 Computers for Everyone
 Countdown 2000
 Death March
 Digital Economy
 Doctors
 Dragon
 E-Biz primer
 ERP Demystified
 First little Boat
 Hotel
 Illusions

Inca Gold
Internet for Everyone

TITLE

Intro to computers
Jesus CEO
Jesus in Blue jeans
Kane and Abel
Management Mole
Managing Tech People
Managing software process
Silicon Samurai
Book of Business Wisdom
Book of Business Wisdom
Book of Leadership Wisdom
The class
The Complete Ad of Feluda
The End of Imagination
The Hunt of Red October
The Legends of Khasak
The Mythical Man Month
The Path
The Testament
Zen Computing

41 rows selected.

```
2.SQL> SELECT ID,TITLE,AUTHOR,PUBLISHER,YEAR,PRICE  
        FROM BOOK;
```

ID	TITLE	AUTHOR	PUBLISHER	YEAR	PRICE
----	-----	-----	-----	-----	-----
A01	A Painted House	Grisham	Random House	2001	195.55
A02	Abduction	Cook	Pan books	2000	360
A03	Airport	Hailey	Corgi books	1968	175.45
B01	Biplane	Bach	Dells books	1966	283.45
B02	Blood line	Sheldon	Warner books	1977	100.15
B03	Blue Gold	Cussler	Simon and schuster books	2000	285
C01	Catch 22	Heller	Random House	1994	250.1
D01	Doctors	Segal	Bantam books	1998	150
D02	Dragon	Cussler	Harper collins	1990	123.55
F01	Flood tide	Cussler	Simon and schuster	1997	414.5
H01	Hawaii	Michener	Mandarin Books	1959	124.5
H02	Hotel	Hailey	Corgi books	1965	175.75
I01	Icon	Forsyth	Corgi books	1966	182.35

I02	Illusions	Bach	Dell books	1997	330
I03	Inca Gold	Cussler	Harper collins	1994	124
I04	Invasions	Cook	Pan books	1997	177.9
O01	One	Bach	Dell books	1988	289
P01	Prizes	Segal	Bantam books	1995	256.1
S01	Serpent	Cussler	Simon and schuster	1999	532.8
S02	Sheba	Higgins	Signet books	1995	125
T01	The class	Segal	Bantam books	1985	145.8
T02	The Runner	Reich	Headlines books	2000	199.95
T03	The Simple truth	Baldacci	Simon and schuster	1997	211
T04	Thunder Point	Haggins	Signet books	1993	95.65
T05	Timeline	Crichton	Century books	1999	623.3
V01	Vector	Cook	Macmillan	1999	424.8

26 rows selected.

```
3.SQL> SELECT PUBLISHER
        FROM BOOK;
```

PUBLISHER

Random House
Pan books
Corgi books
Dells books
Warner books
Simon and schuster books
Random House
Bantam books
Harper collins
Simon and schuster
Mandarin Books
Corgi books
Corgi books
Dell books
Harper collins
Pan books
Dell books
Bantam books
Simon and schuster
Signet books
Bantam books

PUBLISHER

Headlines books

Simon and schuster
Signet books
Century books
Macmillan

26 rows selected.

```
4.SQL> SELECT TITLE,AUTHOR,PUBLISHER,YEAR
2 FROM BOOK
3 WHERE YEAR IN(1999,2000);
```

TITLE	AUTHOR	PUBLISHER	YEAR
-----	-----	-----	----
Abduction	Cook	Pan books	2000
Blue Gold	Cussler	Simon and schuster books	2000
Serpent	Cussler	Simon and schuster	1999
The Runner	Reich	Headlines books	2000
Timeline	Crichton	Century books	1999
Vector	Cook	Macmillan	1999

6 rows selected.

```
5.SQL> SELECT TITLE,AUTHOR,PUBLISHER,YEAR
2 FROM BOOK
3 WHERE YEAR=1999
4 OR YEAR=2000;
```

TITLE	AUTHOR	PUBLISHER	YEAR
-----	-----	-----	----
Abduction	Cook	Pan books	2000
Blue Gold	Cussler	Simon and schuster books	2000
Serpent	Cussler	Simon and schuster	1999
The Runner	Reich	Headlines books	2000
Timeline	Crichton	Century books	1999
Vector	Cook	Macmillan	1999

6 rows selected.

```
6.SQL> SELECT TITLE,AUTHOR,PUBLISHER,PRICE
2 FROM BOOK
3 WHERE PRICE >=400
4 AND PRICE <=600;
```

TITLE	AUTHOR	PUBLISHER	PRICE
Flood tide	Cussler	Simon and schuster	414.5
Serpent	Cussler	Simon and schuster	532.8
Vector	Cook	Macmillan	424.8

```
7.SQL> SELECT TITLE,AUTHOR,PUBLISHER,PRICE
2 FROM BOOK
3 WHERE PRICE NOT BETWEEN 400 AND 600;
```

TITLE	AUTHOR	PUBLISHER	PRICE
A Painted House	Grisham	Random House	219.016
Abduction	Cook	Pan books	360
Airport	Hailey	Corgi books	175.45
Biplane	Bach	Dells books	283.45
Blood line	Sheldon	Warner books	100.15
Blue Gold	Cussler	Simon and schuster books	285
Catch 22	Heller	Random House	250.1
Doctors	Segal	Bantam books	150
Dragon	Cussler	Harper collins	123.55
Hawaii	Michener	Mandarin Books	124.5
Hotel	Hailey	Corgi books	175.75
Icon	Forsyth	Corgi books	182.35

TITLE	AUTHOR	PUBLISHER	PRICE
Illusions	Bach	Dell books	330
Inca Gold	Cussler	Harper collins	124
Invasions	Cook	Pan books	177.9
One	Bach	Dell books	289
Prizes	Segal	Bantam books	256.1
Sheba	Higgins	Signet books	125
The class	Segal	Bantam books	145.8
The Runner	Reich	Headlines books	199.95
The Simple truth	Baldacci	Simon and schuster	211
Thunder Point	Haggins	Signet books	95.65
Timeline	Crichton	Century books	623.3

23 rows selected.

```
8.SQL> SELECT TITLE,AUTHOR,PRICE/47 AS PRICE_D
2 FROM BOOK
3 WHERE PRICE > 400;
```

TITLE	AUTHOR	PRICE_D
-----	-----	-----
Flood tide	Cussler	8.8191489
Serpent	Cussler	11.33617
Timeline	Crichton	13.261702
Vector	Cook	9.0382979

```
9.SQL> SELECT *
2 FROM BOOK
3 WHERE PRICE IS NULL;
```

No rows selected

AGGREGATE FUNCTIONS:

```
10.SQL> SELECT PUBLISHER,
2 AVG(PRICE) AS AVGPR,
3 MAX(PRICE) AS MAXPR,
4 MIN(PRICE) AS MINPR
5 FROM BOOK
6 GROUP BY PUBLISHER
7 HAVING COUNT(*)>2;
```

PUBLISHER	AVGPR	MAXPR	MINPR
-----	-----	-----	-----
Bantam books	183.96667	256.1	145.8
Corgi books	177.85	182.35	175.45
Simon and schuster	386.1	532.8	211

```
11.SQL> SELECT TITLE,AUTHOR,PUBLISHER,PRICE
2 FROM BOOK
3 WHERE PRICE > 300
4 ORDER BY PRICE DESC;
```

TITLE	AUTHOR	PUBLISHER	PRICE
-----	-----	-----	-----
Timeline	Crichton	Century books	623.3
Serpent	Cussler	Simon and schuster	532.8
Vector	Cook	Macmillan	424.8
Flood tide	Cussler	Simon and schuster	414.5
Abduction	Cook	Pan books	360
Illusions	Bach	Dell books	330

6 rows selected.

```
12.SQL> SELECT TITLE,AUTHOR,PUBLISHER,PRICE/40 AS PRICE_D
2 FROM BOOK
3 WHERE PRICE > 300
4 ORDER BY 4;
```

TITLE	AUTHOR	PUBLISHER	PRICE_D
Illusions	Bach	Dell books	8.25
Abduction	Cook	Pan books	9
Flood tide	Cussler	Simon and schuster	10.3625
Vector	Cook	Macmillan	10.62
Serpent	Cussler	Simon and schuster	13.32
Timeline	Crichton	Century books	15.5825

6 rows selected.

```
13.SQL> SELECT TITLE
2 FROM CATALOGS
3 WHERE PRICE > (SELECT AVG(PRICE)
4 FROM CATALOGS);
```

```
TITLE
-----
201Principles of S/W Devp
A Guide to scm
Beyond Soap,Water
Death March
Digital Economy
First little Boat
Managing Tech People
Managing software process
Book of Business Wisdom

Book of Business Wisdom
Book of Leadership Wisdom
The Path
The Testament
Zen Computing
```

14 rows selected.

```
14.SQL> SELECT TITLE
```

-
- 2 FROM CATALOGS
3 WHERE PRICE > 507.8;

TITLE

201Principles of S/W Devp
Airport
A Guide to scm
Beach at Miramar
Beyond Soap,Water
Biplane
Business Notes
Catch 22
Computers for Everyone
Countdown 2000
Death March
Digital Economy
Doctors
Dragon
E-Biz primer
ERP Demystified
First little Boat
Hotel
Illusions
Inca Gold
Internet for Everyone
Intro to computers
Jesus CEO
Jesus in Blue jeans
Kane and Abel
Management Mole
Managing Tech People
Managing software process
Silicon Samurai
Book of Business Wisdom

TITLE

Book of Business Wisdom
Book of Leadership Wisdom
The class
The Complete Ad of Feluda
The End of Imagination
The Hunt of Red October
The Legends of Khasak

The Mythical Man Month
The Path
The Testament
Zen Computing

41 rows selected.

SUBQUERIES:

```
15.SQL> SELECT TITLE,PRICE
2 FROM CATALOGS
3 WHERE PRICE < (SELECT AVG (PRICE)
4 FROM CATALOGS);
```

TITLE	PRICE
-----	-----
Airport	617.6
Beach at Miramar	784.26
Biplane	689.51
Business Notes	850.79
Catch 22	668
Computers for Everyone	634.4
Countdown 2000	684.8
Doctors	600.8
Dragon	582.66
E-Biz primer	718.4
ERP Demystified	766.79
Hotel	617.6
Illusions	721.76
Inca Gold	583.33
Internet for Everyone	600.8
Intro to computers	634.4
Jesus CEO	735.2
Jesus in Blue jeans	850.11
Kane and Abel	597.44
Management Mole	600.8
Silicon Samurai	668

TITLE	PRICE
-----	-----
The class	597.44
The Complete Ad of Feluda	765.44
The End of Imagination	532.26
The Hunt of Red October	597.44
The Legends of Khasak	584
The Mythical Man Month	837.35

27 rows selected.

```
16.SQL> SELECT TITLE, YEAR, PRICE
2 FROM CATALOGS
3 WHERE PRICE > (SELECT AVG (PRICE)
4 FROM CATALOGS)
5 AND YEAR > (SELECT AVG(YEAR)
6 FROM CATALOGS);
```

TITLE	YEAR	PRICE
-----	-----	-----
201Principles of S/W Devp	1995	1103.46
A Guide to scm	2000	2996.48
Beyond Soap, Water	1998	973.76
Death March	1997	1098.08
Digital Economy	1996	861.54
First little Boat	1993	903.87
Managing Tech People	1997	898.5
Book of Business Wisdom	1997	1385.7
Book of Business Wisdom	1997	1385.7
Book of Leadership Wisdom	1998	1379.65
The Path	1996	866.24
The Testament	1999	896.48
Zen Computing	1999	1096.74

13 rows selected.

UNIT-V

PROCEDURE LANGUAGE-SQL

1. Write a short note on PL/SQL.

PL/SQL is an extension to non-procedural language SQL. It can be called inside a standard procedural language SQL. Thus the power of both SQL and that of standard programming language can be combined together. It is achieved using PL/SQL.

The advantages of this combination are:

- Modularity
- Portability
- Higher productivity
- Better performance
- Better control over program
- Having block structure.

2. Explain about PL/SQL block structure.

PL/SQL block are of two types:

- An anonymous block
- A sub-program

ANONYMOUS BLOCK:

It is a block without a name. It is declared in the application at a point from where it is to be executed. The structure of an anonymous block is:

```
DECLARE  
  <Declaration>  
BEGIN  
  <Executable statements>  
EXCEPTION  
  <Exception Handlers>  
END;
```

Corresponding to the word DECLARE, a Declaration Section is defined. This section contains all variables, constants, cursors and user-defined exceptions which will be references in the executable section. It is an optional section.

All the executable statements of SQL and PL/SQL are placed inside a block which starts with Reserved Word BEGIN and ends with Reserved Word END. It is a mandatory section.

PL/SQL allows grouping of several SQL and PL/SQL executable statements and naming this group as a single unit called sub-program.

The single named block may be declared as Procedures, Functions or Stored Package.

Corresponding to Exception, an Exception Handling Section can be defined by the User. It indicates what action to perform when errors and exceptions arise. It is also an optional section.

3. Explain about PL/SQL declaration section.

PL/SQL has a character set, reserved word punctuation, data types, syntax, rules and statement formation.

Character set:

The PL/SQL character set includes: The upper case and lower case letters A,B,...,Z,a,b,...,z. The decimal numbers 0 to 9, tabs, space and carriage returns. The symbols (), -, +, *, / < > ! : ; . ' \$ ^ & ? { }, [].

PL/SQL is not case sensitive like C language, and so lower case letters are equivalent to corresponding upper case letters.

Reserved words:

Reserved words have special meanings. They cannot be redefined. These are BEGIN, END and are used to bracket the executable section of a block or sub-program. Generally the reserved words should be written in upper case to enhance readability, but they can be written in lower case as well.

Lexical symbols:

The symbols such as identifier, names and literals are separated by one or more spaces.

Delimiters:

It is a simple or compound symbol which has a special meaning in PL/SQL. Delimiters can be used to indicate arithmetic operations such as addition, Substraction, etc.,

Identifiers:

These can be used to name PL/SQL program objects and units, which include constants, variables, exceptions, cursors, subprogram and packages.

Literals:

Character and Date literals must be enclosed in single quotes. Numeric literals are of two kinds, integers and real numbers.

PL/SQL datatypes:

Besides scalar types like int, varchar, real DATE, etc., there are composite Types- RECORD, TABLE, VARRAY and LOB Types- RECORD, TABLE, VARRAY.

Declaring variables and constants:

Declaration allocates storage space for a value, specifies its data type and names the storage location so that the value can be referenced. Variables are declared in the DECLARE section of PL/SQL block.

Its general syntax is:

```
<VAR-NAME><TYPE>[:=<VALUE>];
```

Example:

```
TitleId NUMBER(3);  
Tname CHAR(20);  
Price NUMBER(3);  
Discount CONSTANT NUMBER(2):=10;  
TextCat VARCHAR2(5);
```

4. Explain about the RESERVED WORD in PL/SQL.

DEFAULT values can also be used in PL/SQL.

For example,

Discount CONSTANT NUMBER(2):=10;can be written as

Discount NUMBER(2) DEFAULT 10;

Using reserved word %TYPE: To avoid type and size conflict between declared variable and a column of a table ,the attribute %TYPE can be used.

Var _TID Title.TitleId%TYPE;

Using reserved word %ROWTYPE: It can store entire row of a Table selected from it or fetched by a cursor.

Var_Row_Title Title%ROWTYPE;

To refer to a particular column say, Tname we write:

Var_Row_Title.Tname:='JAVA';

It must be noted that%ROWTYPE cannot be used to initialize a column.

5. Explain Operator Precedence in PL/SQL.

The arithmetic,logical and concatenation operators used in SQL . Besides them,the exponentiation operator ** is included in PL/SQL.

OPERATOR	OPERATON
**, NOT	Exponentiation and Negation
+, -	Identity,negation
/, *	Division, Multiplication
+, -,	Addition, subtraction, concatenation
=, !=, <, > <=, >= IS NULL, LIKE,BETWEEN,IN	Comparison
AND	Conjunction
OR	Inclusion

The SQL group functions. MAX, MIN, COUNT, AVG, SUM, STDDEV and VARIANCE are not valid in PL/SQL block since they deal with a group of rows in a table. They are available only within SQL statements.

For error reporting in PL/SQL execution, errors are trapped using SQLCODE and SQLERRM functions.

- SQLCODE returns the internal error code of ORACLE.
- SQLERRM returns the message or description corresponding to the code.

A User can capture the code and take appropriate action in case some exception occurs during execution of PL/SQL.

6. Explain about PL/SQL Executable statements.

PL/SQL is very rich in terms of iterative Loops. There are various Loops-Simple Loop, WHILE LOOP and FOR LOOP. Conditional statements such as IF-ENDIF, and break statements like EXIT are incorporated in PL/SQL.

LOOPS

Simple Loop or LOOP-END LOOP: Its syntax is:

```
LOOP <statements>
END LOOP;
LOOP
    Total:=Total + 1;
    EXIT WHEN Total > 10
END LOOP
```

When the flow of execution reaches the END LOOP statements, the control goes back to the LOOP. So it is an indefinite loop. An EXIT statement can break the flow of execution.

Conditional statements:

IF-THEN-ELSE is a conditional statement. Its syntax is;

```
IF <condition-1> THEN<statement 1>
[ELSEIF <condition-2> THEN<statement 2>]
ELSE
    <statement 3>
ENDIF
```

ELSE and ELSEIF, both are optional clauses. If first IF fails (i.e. condition-1) controls evaluate condition under ELSEIF (i.e. condition-2). If both condition-1 and condition-2 fail, control goes to statements 3. Under IF there can be any number of ELSEIFs.

```
IF Price > 1000 THEN
    Discount:=25;
ELSEIF Price > 750 AND Price < 1000 THEN
    Discount:=20;
ELSEIF Price > 500 AND Price < 750 THEN
    Discount:=15;
ELSE
    Discount:=10;
ENDIF;
```

7. Explain EXIT statement with suitable example.

EXIT statement breaks the flow of execution of an endless loop. It takes out the control of the statement placed below the END LOOP statement.

```
LOOP
    n:=n+1;
    If n>10 THEN
        EXIT;
    ENDIF;
END LOOP
```

EXIT WHEN statement:

This statement allows the LOOP to continue till the time WHEN condition is not true. The condition coming after the 'WHEN' clause evaluates to TRUE the control goes to the statement next to END LOOP.

The syntax of this statement is:

```
EXIT [loop-label] [WHEN condition];
```

EXAMPLE

```
LOOP
    n:=n+1;
    EXIT WHEN n=10;
END LOOP;
```

8. Explain Looping statements with suitable example.

FOR LOOP:

This loop iterates over a specified range of integer values. It is similar to the for loop of C language. Its syntax is:

```
FOR<var>IN, lower value>.. <upper value> LOOP
<statements>
END LOOP;
```

EXAMPLE

```
For n IN 1..10
LOOP
INSERT INTO TempTable VALUES(n);
END LOOP;
```

WHILE-LOOP:

The syntax of WHILE-LOOP statement is

```
WHILE<condition> LOOP
<statements>
END LOOP;
```

Condition after the WHILE statement is evaluated. If it comes out to be TRUE, then all statements within LOOP and END LOOP are executed. The execution of the LOOP continues as long as the condition remains true.

EXAMPLE

Suppose there are 10 records(tuples) in Author_Title table and TitleId for which CopiesSold is more than 500, their TitleIds are to be identified by inserting them into another column BookId.

```
WHILE n<10 LOOP
SELECT TitleId INTO BookId FROM Title WHERE CopiesSold >500;
```

```
n:=n+1;
END LOOP;
```

Executing PL/SQL Code

PL/SQL Code can be written in any text editor and it can be compiled and executed using the command

```
@<filename>
```

dbms_output.put_line(). This procedure produces the output on the screen. It can accept many variables. The variables are concatenated using double pipe (||) symbol.

EXAMPLE

```
dbms_output.put_line ( TitleId || Tname);
```

The above Example can be rewritten as

```
BEGIN
```

```
WHILE n<10 LOOP
```

```
SELECT TitleId INTO BookId FROM Title WHERE CopiesSold>500;
```

```
n:=n+1;
```

```
dbms_output.put_line(TitleId);
```

```
END LOOP;
```

```
END;
```

COMPOSITE DATA TYPES

9. Explain PL/SQL records with suitable example.

The related variables can be combined together to form a record. The type of record needs to be defined before it can be declared.

The syntax for creating a record is

```
TYPE <type-name> IS RECORD
(
    <field-name1>{field-type | variable%TYPE | table.column%TYPE|
    table%ROWTYPE},
    <field-name2>{field-type | variable%TYPE | table.column%TYPE}
    table%ROWTYPE}
    ....
);
```

Where

<type-name> is the record name

And <field-type> stands for data type including RECORD datatype

For example, consider the following declarations;

```
DECLARE
```

```
EmpId NUMBER(4);
```

```
E-firstname VARCHAR2(10);
```

```
E-middlename VARCHAR2(10);
```

```
E-lastname VARCHAR2(10);
```

```
: : :
```

```
: : :
```

The above variables can be defined in the form of a record data type.

```

DECLARE
  TYPE EMPLOYEE IS RECORD
  (
    EmpId      NUMBER(4);
    E-firstname VARCHAR2(10);
    E-middlename VARCHAR2(10);
    E-lastname  VARCHAR2(10);
    : : :
    : : :
  );
  EmpRec EMPLOYEE;

```

Let us take another example from the realm of Title table.

```

DECLARE
  TYPE TITLETYP IS RECORD
  (
    TId Title.TitleId % TYPE NOT NULL,
    TitleName Title.Tname % TYPE,
    Tprice Title.Price % TYPE,
    Tdiscount Title.Discount % TYPE,
    TTextCat Title.TextCat % TYPE
  );
  TRec TITLETYP;
  SELECT TitleId, Tname, Price,Discount, TextCat INTO TRec FROM Title
  WHERE TitleId='T1';

```

The individual field can be referenced using the DOT notation as used in 'C' Language. Its syntax is:
record-name.field-name

The following example shows how to refer to TitleName field of record TRec. It will contain TitleName as Oracle since we have inserted this value using the SELECT statement above. We print it using the command:

```

BEGIN
  dbms_output.put_line(TRec.TitleName);
END;

```

10.Explain Nested Record with suitable example.

If a record type contains another record type as its component ,then it is called a nested record. Its syntax is:

```

RECORD_NAME.COMPONENT_NAME.FIELD_NAME

```

EXAMPLE 1

Define a record to hold the publisher details and total earnings received after selling all the titles.

```

DECLARE

```

```

TYPE ROYALTYTYPE IS RECORD
(
    TRec      TITLERECTYPE,
    TRoyalty  NUMBER(s),
);
RoyaltyRec  ROYALTYTYPE;
To refer to field TitleId of TRec we use dot notation.
BEGIN
    RoyaltyRec.TRec.TitleId := 'T5';
    :
END

```

Coming back to RECORDTYPE TITLERECTYPE, define two variables: one TitleRec1 and another TitleRec2 of %ROWTYPE of Title as follows:

```

TitleRec1  TITLERECTYPE;
TitleRec2  Title% ROWTYPE;

```

But the following two assignment statement are not valid.

```

1.          TitleRec1:=TitleRec2;
2.          If TitleRec1 = TitleRec2 THEN
    :
    ENDIF;

```

EXAMPLE 2

Consider the following example which defines AUTHORTITLETYPE, uses TITLETYPE to know Price of each Title and then accordingly calculates 15% of price, multiplies it with the number of copies sold to find out the royalty for Authors who have written TitleId='T5'.

```

DECLARE
    TYPE AUTHORTITLETYPE is RECORD
    (
        Tno      Author_Title.TitleId% TYPE;
        Tsold    Author_Title.CopiesSold% TYPE;
        TA#      Author_Title.A#% TYPE,
        Royalty  NUMBER(4);
        ATRec   AUTHORTITLETYPE;
    BEGIN
    SELECT TitleId, A#, CopiesSold, CopiesSold*
    0.15*Price
    INTO ATRec.Tno,ATRec.TA#,
    ATRec.Tsold,ATRec.Royalty
    FROM Author_Title,Title
    WHERE TitleId='T5';
    dbms_output.put_line(ATRec.Tno || ATRec.TA# || ATRec.Tsold || ATRec.Royalty);
    END;

```

11.Explain PL/SQL TABLES with suitable example.

PL/SQL table is a dimensional array.Primary keys can be attached to access rows. The number of rows in PL/SQL table grows as new rows are added. But rows cannot be deleted. The PL/SQL tables can here have only one column and a primary key which cannot be named.Datatype of column is any scalar type whereas primary key is of BINARY_INTEGER. PL/SQL table is defined in two steps. To define a table type, the syntax is:

```
TYPE<type-name> IS TABLE OF
<column-type | variables%TYPE | table.column %TYPE> [NOT NULL]
INDEX BY BINARY_INTEGER;
```

Where

<type-name>is the name used to declare PL/SQL table is subsequent declaration.
<column-type>is a data type like CHAR,DATE,OR NUMBER or %TYPE

To define a table type

```
TYPE  TITLENAMETYPE IS TABLE OF Title.Tname%TYPE NOT NULL
      INDEX BY BINARY_INTEGER;
```

To define a variable of take type,i.e BookName of type TitleNamType we write:

```
      BOOKNAME      TITLENAMETYPE;
```

To refer to rows of PL/SQL table,we must specify a primary key value using an array.

Its syntax is:

```
      PL/SQL_table_name(primary_key_value)
```

Where primary_key_value is of type BINARY_INTEGER.

e.g: BOOKNAME(subscript)

EXAMPLE

Load the Tnames and their Prices into PL/SQL tables and then display the contents of the table.

```
DECLARE
TYPE TITLENAMETYPE IS TABLE OF Title.Tname%TYPE NOT NULL
      INDEX BY BINARY_INTEGER;
TYPE TITLEPRICETYPE IS TABLE OF Title.Price%TYPE INDEX BY
      BINARY_INTEGER;
      BOOKNAME      TITLENAMETYPE;
      BOOKPRICE     TITLEPRICETYPE;
      Subscript     BINARY_INTEGER:=1;
      Ind           NUMBER(3):=1;
BEGIN
      FOR TitleRec IN(SELECT Tname, Price FROM Title)
LOOP
      BOOKNAME(SUBSCRIPT)  := TitleRec.Tname,
```

```

        BOOKPRICE(SUBSCRIPT)    :=TitleRec.Price;
        Subscript := Subscript + 1;
    END LOOP;
    WHILE
        Ind<Subscript
    LOOP
        dbms_output.put_line(BOOKNAME(Ind));
        dbms_output.put_line(BOOKPRICE(Ind));
        Ind := Ind +1
    END LOOP;
END;
```

Write a PL/SQL program to insert all the details of TitleId='T3' to a new table New_Title which has the same structure as Title table.

```

DECLARE
    NewRec    Title%ROWTYPE;
BEGIN
    SELECT INTO NewRec FROM Title WHERE TitleId='T3';
    INSERT INTO New-Title
        VALUES(NewRec.TitleId,NewRec.Tname,NewRec.Price,NewRec.TextCat);
END;
```

12. Explain the Precautionary measures while using PL/SQL procedures.

- * DML in PL/SQL are allowed along with INTO clause. These DML statements are: SELECT,INSERT,DELETE and UPDATE statements which return one and only one row.If SELECT statement returns more than one row then internal system of PL/SQL automatically returns error.
- * INTO clause is used with SELECT statements to store values from the table into variables. It is placed between SELECT and FROM clause.
- * DDL are not allowed in PL/SQL.

CURSOR MANAGEMENT AND ADVANCED PL/SQL

13. Write a short note on Cursors.

Cursor allows the developers to access individual rows of data instead of working with the entire set of rows which are returned using a SELECT statement. By accessing individual row, the developers can perform any operation on it as they can do with any record in a programming language such as C,C++ or COBOL.

To execute SQL statement and store information, ORACLE uses private areas. The private area can be named and accessed via a PL/SQL construct known as Cursor.

In fact, any SQL statement can be used within an application program-not only DML statements but DDL statements also. Most DML statements are used but it is easy to find the usage of DDL also, e.g. CREATE TABLE can also be used to save intermediate results. All SQL statements cannot be used in PL/SQL program, DML statements of SQL-SELECT, INSERT, UPDATE and DELETE are only allowed in PL/SQL program.

Most of the SQL DML statements can be embedded straight into higher level languages. But SELECT statement requires special treatment since it may return multiple rows. Some languages like PL/1 were not able to handle more than one record at a time. Therefore, it was necessary to provide by a new type of object known as Explicit Cursor.

Explicit cursor is defined using CURSOR statement. Its syntax is:

```
CURSOR<cursor-name> IS <SELECT-statement>;
```

Where <select statement> includes almost every clause except INTO clause. NULL also cannot be defined as SELECTED item.

EXAMPLE

```
DECLARE
    CURSOR CR IS SELECT Tname, Price FROM Title
    WHERE CopiesSold>1000;
    ....
BEGIN
    ....
END
```

NOTE: CR the CURSOR name is not declared a PL/SQL variable. It is used to store the result of SELECT query.

14. Write a short note on Opening a cursor.

To fetch rows one by one from the cursor, we need to open it. The syntax for opening a cursor is:

```
OPEN <cursor-name>
```

The OPEN statement makes the cursor active. The SELECT statement associated with <cursor-name> is executed. A set of records is identified and cursor <cursor-name> (Say CR) is associated with that set. Cursor CR, also identifies a position in that set, i.e. the position just before the first record in the set.

EXAMPLE

```
OPEN CR
```

The cursor points to the first row. Once the cursor is opened, the current row can be loaded into the variables.

```
FETCH <cursor-name> INTO <variables>;
```

This advances cursor, CR, and positions it onto the next record in the associated set. It assigns fields from that record (to variables) which are defined in SELECT and INTO clauses for

<cursor-name>, CR. After the OPEN, CR points just before the first record, the first FETCH points on it and fetches fields from the first record.

EXAMPLE

```

DECLARE
TempPrice          Title.Price%TYPE;
TempTitleId        Title.TitleId%TYPE:=T5
Cursor CR is
SELECT Price FROM Title WHERE TitleId = TempTitleId;
BEGIN
.....
OPEN CR
LOOP
    FETCH CR INTO TempPrice;
    EXIT          WHEN CR%NOT FOUND
.....
END LOOP;
END;

```

15.What is the command for closing a cursor?

CLOSE CR command closes the cursor, i.e. the active set of rows, which were selected as a result of execution of SQL statements. SELECT are no longer connected with the set of records which were selected when it was opened. However, it is connected with the same SELECT statement. Again if it is opened it will again become connected with some other set

16.Explain Explicit Cursor Attributes with suitable example.

Explicit Cursor has four attributes as summarized in below Table. It is user defined cursor, as the name indicates defined by explicitly users. It is used only to handle multiple rows obtained from DML statement SELECT issued directly in PL/SQL block. So this cursor is used only for queries.

Explicit Cursor Attributes

ATTRIBUTE	MEANING
% ROWCOUNT	It returns the total number of rows fetched.
%ISOPEN	It evaluates to TRUE if cursor is opened otherwise FALSE.
%FOUND	It evaluates to TRUE if last fetch succeeded, i.e, if some rows are available in active set.
%NOTFOUND	It evaluates to TRUE if last fetch failed,i.e, no more rows are available in active set.

%ROWCOUNT

It returns the number of rows that it gets from the database when the cursor is opened. %ROWCOUNT is normally zero. With each fetch the % ROWCOUNT is incremented by one.

EXAMPLE

```

FETCH CR INTO      TPrice;
IF CR% ROWCOUNT > 10 THEN
.....
ENDIF

```

%ISOPEN

It evaluates to TRUE if cursor is opened.

EXAMPLE

```
IF CR% ISOPEN THEN
    ....
ENDIF
```

%FOUND

Once an explicit cursor is opened before the first FETCH statement, it evaluates to FALSE. But after the first FETCH if a row is returned successfully it evaluates to TRUE.

EXAMPLE

```
LOOP
    FETCH CR INTO TempPrice;
    IF CR % FOUND THEN
        ...
    ENDIF
```

%NOT FOUND

If in an active set the last row returned is unsuccessful ,i.e. no more rows are left in active set, the %NOTFOUND evaluates to TRUE.

EXAMPLE

```
LOOP
    FETCH CR INTO TempPrice;
    EXIT WHEN CR% NOTFOUND;
    .
END LOOP;
```

EXAMPLE

Using cursor, get details of those titles from table title where price is more than 100 and display them.

DECLARE

```
Title TitleId    Title.TitleId%TYPE;
Temp Tname      Title.Tname %TYPE;
Temp Price      Title.Price%TYPE;
CURSOR          CR IS
SELECT          TitleId, Tname, Price, FROM Title
WHERE PRICE > 100;
```

BEGIN

```
OPEN CR;
LOOP
    FETCH CR INTO Temp TitleId, TempTname,TempPrice;
    IF CR%FOUND THEN
```

```

    Dbms _output.put_line(Temp TitleId || Temp Tname ||
    Temp Price);
    ELSE
    EXIT;
    ENDIF;
    END LOOP,
    CLOSE CR;
    END;

```

Example

Suppose the Title Tran table has following format:

```
TrantId TranQty TranType Updatestatus
```

And Title is the master table whose fields are : TitleId and Qty only. Tran Type can either be 'RETURN' or 'ISSUE'. If it is RETURN, we add TranQty into Qty field of Title, if it is 'ISSUE' we subtract it from Qty.

```
DECLARE
```

```

    LetTitleId Title.TitleId% TYPE;
    LetQtyin hand Title.TranQty% TYPE;
    LclTranType Titletran.TranType% TYPE
    LclTranQty TitleTran .TranQty% TYPE;

```

```
CURSOR CrTitle is
```

```

    SELECT TrantId, TranQty, TranType FROM Title Tran
    WHERE UPPER (Updatestatus)='N';

```

```
BEGIN
```

```
OPEN CrTitle;
```

```
LOOP
```

```
    FETCH CrTitle INTO
```

```
    lclTitleId,lclQtyinhand , lclTranQty;
```

```
    EXIT when CrTitle%NOTFOUND
```

```
SELECT Qty into lclTranQty
```

```
FROM Title WHERE TitleId=lclTitleId;
```

```
IF Upper (lclTranType)='RETURN'THEN
```

```
    /*check lclTrntype is Return or Issue */
```

```
lclQtyinhand:=lclQtyinhand + lclTranQty;
```

```
ELSE
```

```
lclQtyinhand:=lclQtyinhand- lclTranQty;
```

```
ENDIF;
```

```
UPDATE Title SET Qty = lclQtyinhand /* update Qty in Title*/
```

```
WHERE TitleId = lclTitleId;
```

```
    UPDATE TitleTran SET Updatestatus = 'Y'/* Update status is made
```

```
As 'Y' in Title Tran after this record is processed*/
```

```
WHERE TrantId = lclTitleId;
```

```
END LOOP;
```

```
CLOSE CrTitle;
```

```
COMMIT WORK;
```

END;

In the case of cursor, we first capture the various fields of the database through SQL statement in to the cursor , then define corresponding fields inside the declare section and use them further. The step of defining various fields inside the declare section can be avoided if the cursor is straight fetched into the record. Consider the example prior to the discussed above.

EXAMPLE

```
DECLARE
  CURSOR CR is
  SELECT TitleId, Tname, Price, WHERE Price > 100;
  CrRec CR % ROWTYPE;
BEGIN
  OPEN CR;
  LOOP
    FETCH CR INTO CrRec;
    IF CR%FOUND THEN
  dbms_output.put_line(CrRec. TitleId || CrRec.Tname || CrRec.Price);
    ELSE
      EXIT;
    ENDIF;
  ENDLOOP;
  CLOSE CR;
END;
```

Two more statements can also include references to cursor: current forms of UPDATE and DELETE. If the cursor, CR is positioned on a particular record/row in the database then we can UPDATE or DELETE the “current of CR”, i.e the record on which CR is positioned.

To refer the ‘current row’ from explicit cursor, WHERE CURRENT OF clause of SQL command is used, along with the name of the cursor

Example on WHERE CURRENT OF clause

```
  FETCH CR INTO = 15 THEN
  DELETE FROM Title WHERE CURRENT OF CR;
  UPDATE FROM title WHERE CURRENT OF CR;
```

UPDATE and DELETE are subject to some restrictions.

1. The record to be updated or deleted must be a real record in the database, i.e, it cannot be a record arising due to JOIN.
2. The cursor statement must not include ORDER BY clause.
3. UPDATE clause if included must be of the form FOR UPDATE OF FIELD name [field-name]

Example on FOR UPDATE clause within cursor

```
DECLARE
  CURSOR CR is
  SELECT Titleid , Tname ,Price FROM Title WHERE Price >
  500
  FOR UPDATE of Price;
  CrRec GR%ROWTYPE;
BEGIN
  OPEN CR;
  LOOP
  FETCH GR INTO CrRec;
  IF CrRec.Discount < 20 THEN
    :
    :
  ENDIF;
  ENDIF;
  ENDLOOP;
  CLOSE CR;
  END;
```

17.Explain Implicit Cursor Attribute with suitable example.

Let us study the attributes of an implicit cursor from the below table.

Attributes	Meaning
SQL%FOUND	If DML statement is true then it is TRUE .
SQL%NOTFOUND	If DML statement fails then it is TRUE.
SQL% ISOPEN	It is FALSE always since ORACLE automatically closes an implicit cursor after executing its SQL statement .
SQL % ROWCOUNT	It returns the total number of rows affected by an INSERT, UPDATE , DELETE or single row select.

. With explicit cursor SELECT only can be used. Whereas with rest of the DML statements – INSERT , UPDATE and DELETE implicit cursor is automatically declared by PL/SQL. Also it is declared for DML statement SELECT provided it returns only one row.

1. PL/SQL implicitly gets a cursor for all other SQL statements which are not defined with an explicit cursor.
2. The Values of the cursor attributes refer to latest executed SQL statements .
3. The attributes are prefixed by the key word SQL.
4. Implicit cursor attributes are used in procedural statements . They are not used in SQL statements.

SQL% FOUND

SQL %FOUND evaluates to TRUE if an INSERT , UPDATE o DELETE affects one or more row as a SELECT INTO returns one or more than one row.

```

    IF SQL%FOUND THEN
        INSERT INTO Title VALUES
(& TitleId , &Tname , &Price , &Discount & Textcat)
    ENDIF

```

SQL%NOTFOUND

If an INSERT , UPDATE or DELETE affects no rows or a SELECT INTO clause returns no; rows then SQL%NOTFOUND evaluate to TRUE.

```

    UPDATE TitleId SET Discount =Discount + 5
    WHERE TitleId = 100
    IF SQL%NOTFOUND THEN
        INSERT INTO Interim Table VALUES (TitleId , Tname, Price, Discount Textcat)

```

SQL% ISOPEN

RDBMS Oracle closes the SQL cursor automatically after executing SQL statements. So SQL%ISOPEN always evaluates to FALSE.

SQL % ROWCOUNT

```

    INSERT INTO Title VALUES('T7', 'DB2',300,15,'RDBMS');
    IF SQL % ROWCOUNT > 1 THEN
        dbms_ output.put_line("Duplicate insertion")
    ENDIF

```

CURSOR FOR LOOP

- To open a cursor and fetch rows from the active set(cursor) into fields of the predefined record.
- To close the cursor when all the rows from active set are read or when EXIT command comes.

A cursor FOR LOOP can be used. The cursor FOR LOOP implicitly declares a loop index as a record of type % ROWTYPE. Its syntax is:

```

    FOR <INDEX> IN <CURSOR-NAME> LOOP
        <statements>;
    END LOOP;

```

EXAMPLE

```

    DECLARE
        IclTitleId          Title.TitleId %TYPE;
        IclQtyinhand        Title.TranQty %TYPE;
        IclTranType         TitleTran.TranType %TYPE;
        IclTranQty          TitleTran.TranQty %TYPE;
    CURSOR CrTitle IS
    SELECT TransId, TranQty,TranType FROM TitleTran
    WHERE UPPER (UpdateStatus) ='N';

```

```

BEGIN
  FOR T IN CrTitle /* Five statements are written in example 11.8a */
  LOOP
    /* Corresponding to these two statements */
    SELECT Qty INTO IclQtyinhand
    FROM Title WHERE TitleId = T.TrantId
    IF UPPER (T.TranType) = 'RETURN' THEN
      IclQtyinhand := IclQtyinhand + T.TranQty
    ELSE
      IclQtyinhand := IclQtyinhand – T.TranQty;
    ENDIF;
  UPDATE Title SET Qty = IclQtyinhand
  WHERE TitleId =T.TranId
  UPDATE TitleTran SET UpdateStatus = 'Y'
  WHERE TitleId = T.TrantId;
END LOOP;
COMMIT WORK
END;

```

18. Write a short note on Exception Handlers with suitable example.

The syntax of defining an exception handler is

When <exception-identifier> THEN <actions>,
 <action> may be one or more PL/SQL or SQL statements. The actions of one exception handler are delimited by either the end of the block (END) or by the start of another exception handler (WHEN). The main exceptions which are going to occur as a result of SELECT statement are:

NO_DATA_FOUND (no rows returned) and TOO_MANY_ROWS (more than one row returned).

These are predefined exceptions which are defined by runtime system. There is no need to raise. They are raised automatically

EXAMPLE

The following program displays the titlename for a given TitleId along with Price. If there is no title or if there is more than one title, then instead of giving error the program will give a proper message.

```

CREATE PROCEDURE prShowPrice (TID member)AS
DECLARE
  IclTname      Title.Tname % TYPE;
  IclPrice      Title.Price % TYPE;
BEGIN
  SELECT Tname,Price INTO IclTname,IclPrice
  FROM Title
  WHERE TitleId = TID;
  dbms_output.Put_line (IclTname || ' ' || IclPrice);

```

```

EXCEPTION
WHEN NO_DATA_FOUND THEN dbms_output.put_line('No Such Title');
WHEN TOO_MANY_ROWS THEN dbms_output.put_line('More than one Title');
END;
```

EXAMPLE

```

DECLARE
Error-message          VARCHAR2(100);
Error-code             NUMBER;
BEGIN
.....
.....
EXCEPTION
WHEN OTHERS THEN
Error-message:= SUBSTR (SQLERRM,1,100);
Err-code:= SQLCODE;
dbms_output.put_line(error-message || ' ' || err-code);
END;
```

Some of the predefined exceptions are as follows:

Expection Name	Oracle Error
CURSOR-ALREADY-OPEN	ORA -06511
ZERO-DIVIDE	ORA – 01476
LOGIN-DENIED	ORA -01017
NO-DATA-FOUND	ORA -01403
TOO-MANY-ROWS	ORA- 01422

19. Write a short note on ‘WHEN OTHERS ‘ Exception Handler with suitable example.

In the EXCEPTION section will trap the two exceptions ; NO _DATA_ FOUND and TOO_MANY_ROWS and rest it will ignore. Instead of defining a separate handler for every exception type, we can use ‘WHEN OTHERS’ exception handler is used it should be the last exception in the exception block.

in addition to the above two exception handler , we define one more exception handler ‘WHEN OTHERS’EXCEPTION.

```

WHEN NO_DATA_FOUND THEN
WHEN TOO_MANY_ROWS THEN
WHEN OTHERS THEN dbms_output.put_line ('Error not caputerd above').
```

If we want to capture the exact error, then PL/SQL provides two functions for it.

SQL CODE : It returns the error number associated with the exception which has occurred.

SQL ERRM : It returns character data. It returns full error message related to the exception.

User defined exceptions:

As the name indicates, these are user declared and defined exceptions. These are invoked using RAISE statement. PL/SQL permits users to be defined exceptions. These are different from predefined exceptions. These are to be defined and raised by users only.

Declaring an Exceptions:

Exceptions are declared in declarative part of PL/SQL block and sub programmes. The Syntax is : Identifier EXCEPTION;

The declaration of exceptions and variables is similar, but nothing can be assigned to exception as it is done for variables.

Example

```
DECLARE
.....
/* User defined exception */
Price Negative Exception ;
Lclprice NUMBER(3);
BEGIN
.....
IF lclPrice < 0 THEN
RAISE Price Negative;
ENDIF;
.....
.....
EXCEPTION
WHEN Price Negaive THEN
/* Execute exception handling code which follows*/
.....
END;
```

Exceptions are raised in two ways: Either by the system internally and automatically, for example , when too many rows are selected by SELECT statement (it should be only one in PL/SQL , unlike SQL where you can select many rows), the system generates the exception TOO_MANY_ROWS.

Or as a user you can generate exceptions using the RAISE statement whose syntax is : RAISE exceptions –identifier;

While defining exceptions, remember:

- * Exception cannot be declared twice in the same block . Yes, in two different blocks, the same exceptions can be defined .
- * Exceptions declared inside a block is local within that block, but it is global for all sub – blocks nested inside it.

Example

```
DECLARE
  UserException EXCEPTION;
....
BEGIN
  .....
  BEGIN
    .....
    RAISE UserException;
    .....
  .....
  EXCEPTION
    WHEN UserException THEN
    .....
  END;
.....
END;
```

When an exception is raised in inner BEGIN-END block, control goes to the EXCEPTION block, executes all statements inside this block and then goes out of inner BEGIN-END block.

20. Write a short note on Sub Programs in PL/SQL.

PL/SQL has two types of subprograms: Procedure and Functions. Each sub program is given a name. Sub programs can take arguments. PL/SQL sub program can be stored in the database as stored programs. Such subprograms can be invoked as and when required.

The frameworks of a PL/SQL subprogram consists of :

- A declarative part.
- An executable part, and
- An optional execution handling part.

The declarative part includes declaration of:

- Types.
- Constants
- Variables
- Cursors
- Exceptions.

21. Describe about Procedures with suitable example.

It is a sub program, which does a specific task, and can be called from any PL/SQL Program and also from the SQL prompt. It has two parts: the specification and body .

The procedure specification starts with the keyword PROCEDURE followed by procedure name and an optional list of arguments inside the parenthesis. The procedure body consists of a declarative part, an executable part and an optional exception handling part. The procedure body starts with the keyword IS or AS. It ends with the keyword END followed by procedure name encoded in parenthesis.

The syntax for defining a procedure is:

```
CREATE [OR REPLACE] PROCEDURE <Procedure _Name>
  [parameter list] is
  <local declaration>;
  (executable statements)
  [exception](exception handlers)
END[Procedure_Name];
```

The syntax to execute a procedure is:

```
sql > EXEC <Procedure _Name>(parameters);
```

while declaring the parameter list, containing variables after the <Procedure _Name>, width of datatype is not declared.

EXAMPLE

Procedure prProcl(Fname CHAR(20))is

```
BEGIN
  (set of statements);
END;
```

Here CHAR(20) is wrong ,it should be declared as CHAR. Suppose it is to be found what royalty is to be given for TitleId = T1.The formula to compute Royalty = CopiesSold * Price *0.15.

EXAMPLE

```
CREATE OR REPLACE PROCEDURE prRoyaltyDue (TitCode NUMBER) IS
  IclBookSold NUMBER;
  IclRoyalty NUMBER;
  IclPrice NUMBER;
```

```
BEGIN
SELECT CopiesSold,Price INTO IclBookSold, IclPrice FROM Title WHERE
TitCode = TitleId;
IclRoyalty = IclBookSold * IclPrice * 0.15
dbms_output.put_line ('Royalty due' IclRoyalty);
EXCEPTION
  WHEN NO _DATA_FOUND then
  dbms_output.put_line('No such TitleId exist');
END;
```

The above procedure can be executed from SQL prompt using commands as shown below:

```
SQL > EXEC prRoyaltyDue ('T1')
```

IN Parameter

This Parameter When used to pass values to the subprogram indicates that it is a constant. It cannot be assigned a value. In other words , the value assigned to IN parameter does not change; it remains same , even if a new value is assigned to it inside the called sub progr. It is default parameter mode. It is known as call by value.

Data type specifier , unlike varisble declaration , must be unconstrained as far as its size specification is concerened while giving it as a parameter inside the parenthesis.

Example 1

```
CREATE PROCEDURE prAdd Number (VarA NUMBER(2),VarB NUMBER(2))IS
lclVarC NUMBER(3);
BEGIN
lclVarC:= VarA+VarB;
dbms-output.put-line (lclVarC);
END;
```

Here declaration of VarA and VarB is Illegal . The correct syntax is

```
CREATE PROCEDURE pr AddNumber (VarA NUMBER ,Var B NUMBER) IS
```

```
lclVarC NUMBER(3);
BEGIN
lclVarC:= VarA+VarB;
dbms _otuput.put _line (lclVarC);
END;
```

Example 2

/* The TitleId and the increase in amount of price is passed as parameter */

```
CREATE OR REPLACE PROCEDURE pr Price Increase (Tid NUMBER, PAMt NUMBER )IS
  lclPrice      NUMBER;
  TitleAbsent   EXCEPTION;
BEGIN
SELECT Price INTO FROM Title
Where TitleId = Tid;
IF Title IS NULL THEN
RAISSE TitleAbsent;
ELSE
UPDATE Title set Price FROM Title
Where Title Id = Tid;
IF Title Id IS NULL THEN
RAISE TitleAbsent;
ELSE
UPDATE Title set Price = Price + PAMt
WHERE TitleId = Tid;
ENDIF;
EXCEPTION
WHEN TitleAbsent THEN
```

```
Dbms_output.put_line (TId || 'has TitleId as NULL');  
END prPrice Increase;
```

1. A procedure can be called from any PL/SQL procedure command is prPriceIncreases(TId, PriceHike)

OUT PARAMETER

OUT parameter mode is used when some value is to be returned to the calling program. The next three examples illustrate the use of OUT parameter.

Suppose the user wants to know the Price of a TitleId. The procedure is as follows:

EXAMPLE 3

```
CREATE OR REPLACE PROCEDURE prGetPrice (TitleCode IN CHAR, Cost OUT NUMBER)  
BEGIN  
    SELECT Price from Title WHERE TitleId = TitleCode;  
    Cost := Price;  
END;
```

The procedure prGetPrice can be called from inside other procedure.

```
DECLARE  
    TitleCode NUMBER;  
    Cost NUMBER;  
BEGIN  
    prGetPrice ('T2', Cost);  
    dbms_output.put_line('cost is' || cost);  
END;
```

Output of the above program is:

Cost is 299

EXAMPLE 4

```
CREATE PROCEDURE prAddNum ( A NUMBER, B NUMBER, C OUT NUMBER)  
IS  
BEGIN  
    C: = A+B  
END;  
EXEC prAddNum (5,7,C)  
dbms_output.put_line(C);  
The output for this is 12.
```

Example 5

```
CREATE PROCEDURE prGetTextCat  
(VarTitleId Title, TitleId% TYPE  
VarTexCat OUT TitleCat% TYPE)  
IS  
BEGIN
```

```

SELECT Textcat INTO VarTextCat FROM Title
WHERE TitleId = VarTitleId;
END;

```

IN OUT PARAMETER:

IN OUT Parameter , as the name indicates , can be used to send input from the caller program and output can be sent from the called program to the caller program .

Suppose the publisher has decided to give minimum royalty to each A#s as Rs.2000/- if the actual royalty which is computed as $Royalty = CopiesSold * 0.15 * Price$ is more then 2000/- then actual royalty will be paid to A#s otherwise Rs. 2000/- will be paid . Write a procedure for it.

```

CREATE OR REPLACE PROCEDURE prGetRoyalty (Aid IN NUMBER , Royalty IN OUT
NUMBER, Titlecode IN NUMBER )IS

```

```

    lclPrice      NUMBER(3);
    lclCopiesSold NUMBER(4);
BEGIN
    SELECT Price, CopiesSold INTO lclPrice , lclCopiesSold FROM Title, Author_Title ,
Author where Titlecode = Title.TitleId AND (Title. TitleId = Author_ Ttitle.TitleId AND
Author_Title.A#=Aid);
IF(lclPrice* lclCopiesSold *0.15 > 2000) THEN
Royalty = lclPrice* CopiesSold * 0.15;
ELSE
Royalty:=2000;
ENDIF
END;

```

22.Describe about Functions with suitable example.

Functions and procedures have a similar structure. The only difference is that the functions have a RETURN CLAUSE while procedures donot.

```

CREATE [OR REPLACE] FUNCTION < FUCTION_NAME>
(<arg1> [mode] <datatype>,.....)
RETURN datatype IS
[local declaration]
BEGIN
PL/SQL executable statement
EXCEPTION
Exception handler
END(FUNCTION _NAME);

```

Write a function which sets discount as 10% for TitleId's whose price is less then or equal to Rs.100/-,15% for those whose price is between Rs.100 and Rs.200 otherwise 20%.

```

CREATE OR REPLACE FUNCTIONS fnDiscoDecision(TitleCode NUMBER)
RETURN NUMBER IS
Disc Title.Discount % Type;
IclPrice      Title.Price % TYPE;
IclTitleId    Title.TitleId %TYPE;
BEGIN
SELECT      TitleId, Price INTO IclTitleId,IclPrice FROM Title
      WHERE TitleId = TitleCode;
IF Price    <=100 THEN
Disc := 10;
ELSEIF price >100 AND Price <=200 THEN
  Disc:= 15;
  Disc:=20;
ELSE
ENDIF;
RETURN(Disc);
END fnDisco Decision

```

How to call function fnDiscoDecision?It is shown below

```

DECLARE
      Disc NUMBER(2);
BEGIN
      Disc:= fnDiscoDecision('T2');
      Dbms_output.put_line(Disc);
END;

```

Like a procedure a function has two components: the specification and the body. The functions specification begins with the keyword **FUNCTION** and ends with the **RETURN** clause, which specifies the datatype of the result value. The function body is identical Procedure body.

RETURN

A **RETURN** statement indicates completion of a sub program and returns control to the caller. The execution starts from the statement meant to the subprogram cell.

The **RETURN** statement used in functions must contain an expression which is computed when the **RETURN** statement is executed. The computed value is assigned to the function identifier. So a function must contain at least one **RETURN** any expression. It simply returns control to the caller.

We take examples of computing the area of a rectangle using both Functions and Procedure in Examples below examples.

Example 1

Return statement used in Functions must contain an expression whereas it should not contain any expression in Procedure.

```

CREATE FUNCTION fn ComputerArea of Rectangle (Length NUMBER, Breadth
NUMBER)
    RETURN NUMBER IS
    Area NUMBER (3);
    BEGIN
    Area = Length * Breadth;
    RETURN (Area);
    END fn computer Area of Rectangle;

```

Example 2

```

CREATE PROCEDURE prComputer Area of Rectangle (Length NUMBER , Breadth NUMBER
, Area OUT NUMBER )IS
BEGIN
Area :=length * Breadth;
RETURN
END prComputer Area of Rectangle

```

Precaution While Using PL/SQL Functions

PL/SQL functions can be called only in PL/SQL block and not in SQL as we shall see below;

Example 3

```

CREATE FUNCTION fnSubsidiaryPrice (VarTitleId Title.Title % Type)
    RETURN
    NUMBER IS Netprice NUMBER(3);
    lclPrice NUMBER(3);
    lclDiscount NUMBER(2);
    BEGIN
    SELECT Price , Discount INTO lclPrice ,lclDiscount FROM Title WHERE
    TitleId = Vartitle Id;
    NetPrice := lclPrice – lclPrice * Discount/100;
    RETURN (NetPrice);
    END fnSubsidiaryPrice;

```

NOTE:The above function defined by user can be called in any PL/SQL block as shown below.

```

DECLARE
    PayPrice NUMBER(4);
    BEGIN
    PayPrice := fnSubsidiary Price('T2');
    END;

```

But the above function cannot be called in SQL statement as shown below:

```

BEGIN
    ....
    UPDATE Title Set Price = fnSubsidiaryPrice('T2');
END;
```

The above call to fnSubsidiary Price('T2') is not valid in SQL statement.

23.Explain Stored Packages with suitable example.

A Packages is a databases object that groups logically related PL/SQL objects. It consists of related procedure,functions,cursors and variables as a single logical unit in the database.

Package has two parts: **Specification and body.**

Specification is an interface to applications and contains declaration of variables,procedure,functions,etc. whereas the body contains of different procedure and functions.

The syntax of creating a package is:

```

CREATE [OR REPLACE] PACKAGE <PKG-NAME> AS
/* This is package specification part of the package*/
    /* It contains declarations of global variable and constants which become available to all
    blocks of the package */
    /* It also contains declarations of procedures and functions */
    END [PKG-NAME]
CREATE [OR REPLACE] PACKAGE BODY<PKG-NAME> AS
    /* Specification of each procedure and function along with their declarations*/
    /*Specification of local declarations,their names, datatype and size */
[BEGIN
    ▪ Executable statements
    ▪ Exception;]
END [PKG-NAME];
```

EXAMPLE 1

/* The name of package is pkTitlePack. It has procedure prTitleProc and functions fnSubsidiaryPrice */

/* For a given TitleId, we find out its TitleName and Price in Procedure prTitleProc whereas we find out its net price in Function fnSubsidiary Price */

```

CREATE OR REPLACE PACKAGE pkTitlePack AS
PROCEDURE prTitleProc (TitleId IN TitleId % TYPE);
FUNCTION fnSubsidiaryPrice (TID NUMBER,Cost NUMBER) RETURN NUMBER;
END pkTitlePack;
```

```

CREATE OR REPLACE PACKAGE BODY pk Title pack
AS
```

```

PROCEDURE prTitleProc(TitleId IN Title.TitleId % TYPE)
    lclTname    Title.Tname%TYPE;
    lclPrice    Title.Price%TYPE;
BEGIN
SELECT Tname, Price INTO lclTname ,lclPrice FROM Title WHERE TitleId= TID;
Dbms_output . put_line(TID | | lclTname | | lclPrice);
EXCEPTION
WHEN NO_DATA_FOUND THEN
Dbms_output.put_line (TID | | 'NO DATA' );
END prTitleProc;
/* Here we accept TitleId , find discount corresponding to it , compute net price and return
it to caller*/

CREATE FUNCTION fn SubsidiaryPrice (VarTitleId Title.TitleId%Type)RETURN
NUMBER IS Netprice NUMBER (3) ;
    lclPrice NUMBER(3);
    lclDiscount NUMBER(2);
BEGIN
SELECT Prie , Discount INTO lclPrice , lclPrice , lclDiscount FROM Title WHERE
TitleId = VarTitleId;
NetPrice :=lclPrice – lclPrice * Discount/100;
RETURN (NetPrice);
END fnSubsidiaryPrice;
END pkTitlePack;

```

Dropping Procedure , Function and Package:

To drop a procedure , say prTitleProc , the command is :

DROP PROCEDURE prTitleProc;

Similarly to drop a function , the command is:

DROP PACKAGE pkTitlePack;

You can drop a procedure provided you have a DROP ANY PROCEDURE privilege.

24. Describe about Triggers.

T-SQL

In T-SQL , a trigger is executed only once per action query irrespective of the number of rows affected by the action query. Suppose,15 rows are affected in Title table due to update operation, the update trigger on the Title table fires only once. So trigger must be designed keeping in mind the multiple rows,one row or zero row being affected.

When a row is inserted (or deleted) into a table, the insert (delete) trigger creates an inserted (deleted) table whose column structure is the same as that of original table to which trigger is linked. If we insert (delete) a row in title table, its trigger will create similar –structured table inserted(deleted), and for every row inserted(deleted) in Title table, a row is contained in the inserted (deleted)table. Table 11.4(a) and (b) in a slightly different form for comparing how triggers are written in T-SQL and PL/SQL.

Author_Title Table

A#	TitleId	CopiesSold
90	T1	700
90	T2	900
100	T2	1000
100	T3	500
110	T1	100
110	T2	500
110	T3	800
110	T4	200
110	T5	100

TitleTable

TitleId	Tname	Price
T1	Oracle	399
T2	C	299
T3	C++	199
T4	Sybase	100
T5	TCP/IP	299

EXAMPLE 1

Suppose the deleted Title data is to be archived to TitleArchive table (Table 11.5) for future queries, a delete trigger has to be set.

TitleArchive

TA#	TName	TPrice

```
CREATE TRIGGER trTitleDeleted ON Title
FOR DELETE
AS
INSERT TitleArchive
SELECT TitleId, Tname,Price FROM deleted
GO
```

PL/SQL

In PL/SQL the option is given to execute the statements of Triggers **BEFORE OR AFTER INSERT, DELETE OR UPDATE**. Besides, **BEFORE** and **AFTER** option, PL/SQL gives one more option. **INSTEAD OF**. Let's see how the triggers is written using PL/SQL

```
CREATE OR REPLACE TRIGGER trTITLEDeleted ON Title
AFTER DELETE ON Title
FOR EACH ROW
BEGIN
INSERT TA#, Tname , Tprice VALUES
(OLD ,TitleId , OLD , Tname , OLD.Price)
END;
```

In PL/SQL, let's see how we use **INSTEAD OF** clause in Triggers. Suppose we create a view V1 using Tables and Author-Title. Whenever a delete is fired on V1 for a TitleId, corresponding records are deleted from Title and Author_Title.

```
CREATE TRIGGER T1
SELECT Title.TitleId , A#, Tname , CopeisSold FROM Title , Author_Title
WHERE Title.TitleId = Author_Title.TitleId
CREATE TRIGGER T1
INSTEAD OF DELETE ON V1
FOR EACH ROW
BEGIN
DELETE FROM Title
WHERE Title = OLD.TitleId;
DELETE FROM Author _ Title
WHERE TitleId = OLD.TitleId;
END;
```

OBSERVATIONS

In PL/SQL we don't pick up the old column values of the table on which trigger is set from, deleted or inserted table as it happens in T-SQL, inserted these values are picked up using the keyword, OLD.

The INSTEAD OF clause is used to delete rows from tables rather than deleting them from views.

25. Explain Object oriented Technology in PL/SQL

In PL/SQL supports object orientation. Suppose, we have the Author Table

Author Table

A#(Primary key)	NUMBER(3),
Aname	VARCHAR(20),
Astatus	NUMBER(2),
Street	VARCHAR2(20)
City	VARCHAR2(20)
State	VARCHAR2(20)
PinCode	NUMBER(6);

In object – oriented model , the above information can be implemented as

```
CREATE TYPE ADD_TYPE AS OBJECT
(STREET VARCHAR2(20),
CITY VARCHAR2(20),
STATE VARCHAR2(20),
PINCODE NUMBER(6));
CREATE TYPE AUTHOR OF AUTHOR _TYPE ;
INSERT INTO AUTHOR VALUES
(AUTHOR TYPE (130, 'METHA', 20,
ADD_TYPE ('C2D/56B', 'JANAKPURI' 'NEW DELHI', 'DELHI', 110058));
```

The above feature is available in ORACLE . The ORACLE RDBMS is called ORDBMS. SQL and PL/SQL both can be used to manipulate Relational and object Data.

26. Explain Nested table in PL/SQL with suitable example.

A group of similar item is known as **collections**. Nested Table is a way of grouping similar items for data models which require integrity . It suitable for master- transaction and one to many relationships.

A Nested Table is a Table which stores in it , data which cannot be accessed directly . Suppose there is table of the form given table.

A#	Aname	TitleId	Tname
100	Metha	01	MS SQL Server
		02	Sybase
110	Nanda	03	Informix
		04	Oracle

We have a table with column TitleId and Tname (i.e books written by an Author) inside the table the columns are : A# and Aname . The table with columns TitleId and Bookswritten can be made as a column of table containing A# and Aname . In ORACLE, nested tables are supported . Nested Tables can be included in a Table definitions as one of the columns . They can be manipulated using SQL.

REFERED BOOKS:

1. Rajesh Narang, “*Database Management Systems*”, Second Edition, PHI Learning pvt., 2012.
2. Alexis Leon and Mathews Leon, “*Essentials of Database Management System*”, Vijay Nicole Imprints pvt., 2006.