DATABASE MANAGEMENT SYSTEMS

DATABASE SYSTEM ARCHITECTURE

Multiple Choice Type Questions

1. Which defining a numeric number field that can hold 3 digits before the decimal point and 3 digits after the decimal points, the width would be given as [WBUT 2011]

a) 6 b) 7 c) 8 d) 3

Answer: (a)

2. The data dictionary tells the DBMS [WBUT 2012]
   a) what files are in the database
   b) what attributes are possessed by the data
   c) what these files contain
   d) all of these

Answer: (d)

3. The employee salary should not be greater than Rs 20,000. This is [WBUT 2014, 2017]

a) integrity constraint b) referential constraint
c) over-defined constraint d) feasible constraint

Answer: (a)

4. The information about data in a database is called [WBUT 2017]
   a) Metadata b) Teradata
   c) Hyperdata d) None of these

Answer: (a)

5. Tree structures are used to store data in [WBUT 2018]
   a) network model b) relational model
   c) hierarchical model d) file based system

Answer: (b)

Short Answer Type Questions

1. Describe the three-schema architecture. [WBUT 2006]

OR,

Explain in brief 3-schema architecture of DBMS. [WBUT 2013]

OR,

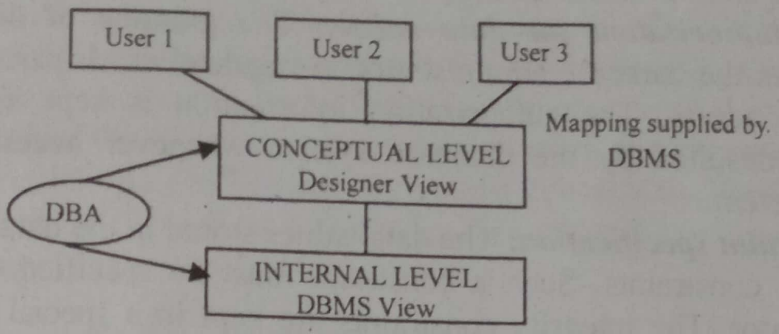
Describe three layer architecture of DBMS. [WBUT 2013]

OR,

Describe Three-Schema architecture of DBMS. [WBUT 2015]

Answer:

Three Schema Architecture of DBMS



The three levels of the architecture

- **The internal level is** the one closest to physical storage – i.e., it is the one concerned with the way the data is physically stored.
- **The conceptual level** is a "level of indirection" between the other two. (Gives abstraction on to the upper level of the lower level)
- **The external level** is the one closest to the users i.e., it is the one concerned with the way the data is viewed by individual users.

If the external level is concerned with *individual* user views, then the conceptual level is concerned with a *community* user view. In other words, there will be many distinct external views, each consisting of a more or less abstract representation of some portion of the total database, and there will be precisely one conceptual view, consisting of a similarly abstract representation of the database in its entirely. There will be precisely one internal view, representing the total database as physically stored.

2. What are the main functions of a database manager? List five major functions of Data Base Administrator. **[WBUT 2006]**

OR,

Explain the roles of a database administrator (DBA). **[WBUT 2009]**

OR,

Discuss the role of DBA. **[WBUT 2017]**

OR,

Explain the role of database administrator? **[WBUT 2018]**

Answer:

The functions of the DBA include the following:

(1) Schema definition: The DBA creates the original database schema by writing set of definitions that is translated by the DDL compiler to a set of tables that is stored permanently in the data dictionary.

(2) Storage structure and access-method definition: The DBA creates appropriate storage structures and access methods by writing a set of definitions, which is translated by the data-storage and data-definition-language compiler.

(3) Schema and physical-organization modification: Programmers accomplish the relatively rare modifications either to the database schema or to the description of the physical storage organization by writing a set of definitions which is used by either the

DDL compiler or the data-storage and data-definition-language compiler to generate modifications to the appropriate internal system tables (for example, the data dictionary).

(4) Granting of authorization for data access: The granting of different types of authorization allows the database administrator to regulate which parts of the database various users eon access. The authorization information is kept in special system structure that is consulted by the database system whenever **access** to the data is attempted in the system.

(5) Integrity-constraint specification: The data values stored in the database must satisfy certain consistency constraints. Such a constraint must be specified explicitly by the database administrator. The integrity constraints are kept in a special system structure that is consulted by the database system whenever an update takes place in the system.

(6) Designing security and integrity constraint specification: The data values stored in the database must satisfy certain consistency constraints. Such a constraint must be specified explicitly by the database administrator. The integrity constraints are kept in a special system structure that is consulted by the database system whenever an update takes place in the system.

(7) Routine Maintenance: Periodically backing up the data either onto some movable devices and/or to remote servers to prevent and protect data from loss due to various reasons. Also ensures free disk space for regular operations. Monitoring jobs running on the database and ensuring that performance is not degraded by the method loosely called performance tuning.

3. Write the difference between procedural and non-procedural DML. [WBUT 2006]

OR,

What is the difference between procedural and non-procedural DML?

[WBUT 2013, 2017]

Answer:

DML language enable user to access or manipulate data as organized by appropriate data model.

- **Procedural DML or Low level:** DML requires a user to specify what data are needed and how to get those data.
- **Non-Procedural DML or High level:** DML requires a user to specify what data are needed without specifying how to get those data.

4. What is the difference between a database and a table? [WBUT 2009, 2011]

Answer:

Database is a Pool of data. Tables are the containers of holding the data. As per relational database management system, tables are organized following the rules of total participation and/or partial participation. Implementation is done with the help of primary Key and foreign key relationship.

5. Explain DDL, DML and DCL. **[WBUT 2012]**

Answer:

Data Definition Language (DDL)

A Data Definition Language (DDL) is used to describe the details of the data. The conceptual schema is specified by a set of definitions expressed by this special language. The DBMS will have a compiler whose function is to process DDL statements in order to identify description of the schema constructs and to store the schema descriptions as a set of tables in a special file called *data dictionary*. Thus, data dictionary is a file that contains data about data which is *called metadata*. This file is consulted before actual data are read or modified in the data base system.

Storage Definition Language: The SDL is used to specify the internal schema. The mapping between two languages can be specified by any one of the languages. In some DBMS, there is only one language which has both DDL and SDL capabilities.

To construct, modify and administer the database the following commands are used.

- CREATE Table
- ALTER TABLE
- DROP TABLE

Data Manipulation Language (DML)

Once the database schemas are compiled and the database is filled with data, the data manipulation needs to covers the following functions:

1. The retrieval of information stored in the database.
2. The insertion of new information into the database.
3. The deletion of information from the database.
4. The modification of information stored in the database.

The language that enables the users to access or manipulate data is called the ***data-manipulation language*** (DML). DML are of two types.

Procedural DML: which require users to specify what data are needed and how to get those data.

Nonprocedural DML: which require a user to specify what data are needed without specifying how to get it.

The following statements are used for data manipulation.

- **SELECT** is used to retrieve data from the table.
- **UPDATE** is used to modify the data in the table.
- **DELETE** is used to delete one or more records from a relation.
- **INSERT** is used to insert a new tuple into a specified relation.

Data Control Language (DCL)

DCL is used to control the data population. It enables users to specify the beginning and ending of transactions. It deals with the authorization, access rights, recovery and integrity issues. The following statements are used to control the data in the data base.

Recovery and Concurrency:

- COMMIT
- ROLLBACK

Security:

- GRANT
- REVOKE

Integrity Constraints:

Integrity constraints are enforced by the system. For example, one can specify that an attribute of a relation will not accept null values. Detail syntaxes and examples are discussed later.

6. Indicate the advantage of DBMS over conventional file system. **[WBUT 2013]**

Answer:

- Reduced data redundancy and Reduced updating errors and increased consistency
- Greater data integrity and independence from applications programs
- Improved data access to users through use of host and query languages
- Improved data security

7. Could a modern university be supported if a file processing system was used to manage enterprise information? List four major difficulties that are likely to arise if no database system was available to the university. Discuss the difficulties.

Answer: **[WBUT 2014]**

A modern university have to manage its large enterprise information because it can have many departments, each with its own information system and data files. Three potential problems exist in a file processing environment. The first problem is data redundancy, which means that data common to two or more information systems is stored in several places. Data redundancy requires more storage space and maintaining and updating data in several locations is expensive. Second, data integrity problems can occur if updates are not applied in every file. Changing the data in only one of the systems will cause inconsistent data and result in incorrect information in the second system. The third problem is the rigid data structure of a typical file processing environment. Businesses must take decisions based on company-wide data and managers often require information from multiple business units and departments. In a file processing environment, that means retrieving information from independent, file-based systems, which is slow and inefficient. Fourth is database system needs less storage. Theoretically, all occurrences of data items need be stored only once, thereby eliminating the storage of redundant data. System developers and database designers often use data normalization to minimize data redundancy.

8. What is Data dictionary? **[WBUT 2016]**

Answer:

Data Dictionary: 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. Most database management systems keep the *data dictionary* hidden from users to prevent them from accidentally destroying its contents. In other words it is known as meta data or data about the data.
Relational algebra is a procedural *query language*, which takes instances of relations as input and yields instances of relations as output. It uses operators to perform queries. An operator can be either unary or binary. The five fundamental unary operations are select, project, Union, Set different and rename.

1. $\sigma_{subject = "database"}(\text{Books})$
2. $\prod_{\text{subject, author}} (\text{Books})$
3. $\prod_{\text{author}} (\text{Books}) \cup \prod_{\text{author}} (\text{Articles})$
4. $\prod_{\text{author}} (\text{Books}) - \prod_{\text{author}} (\text{Articles})$
5. $\rho_{x}(E)$

Long Answer Type Questions

1. Write a short note on Advantages of DBMS. **[WBUT 2016]**

Answer:

Advantages

- Reduced data redundancy
- Reduced updating errors and increased consistency
- Greater data integrity and independence from applications programs
- Improved data access to users through use of host and query languages
- Improved data security
- Reduced data entry, storage, and retrieval costs
- Facilitated development of new applications program

ENTITY-RELATIONSHIP MODEL

Multiple Choice Type Questions

[WBUT 2006, 2008]

1. Cardinality Ratio means
   a) Number attributes associated with an entity
   b) Number of Relation of an entity-relationship diagram
   c) A ratio between number of relation and number of entity of an entity-relationship diagram
   d) The number of entities to which another entity can be associated via a relationship set

Answer: (d)

2. Overall logical structure of a database can be expressed graphically by [WBUT 2009, 2016]
   a) ER diagram b) Records c) Relations d) Hierarchy

Answer: (a)

3. The information about data in a database is called [WBUT 2009, 2011, 2016]
   a) meta data b) tera data c) hyper data d) none of these

Answer: (a)

4. A table can have only one [WBUT 2010, 2012]
   a) Primary key b) Candidate key c) Super key d) all of these

Answer: (a)

5. What is a RDBMS terminology for a set of legal values that an attribute can have? [WBUT 2010]
   a) Tuple b) Relation c) Attribute d) Domain

Answer: (d)

6. What is the smallest unit of data in a relational model? [WBUT 2010, 2012]
   a) Data type b) Field c) Data value d) None of these

Answer: (c)

7. The word 'loss' in lossless refers to [WBUT 2010]
   a) loss of information b) loss of attributes
   c) loss of relations d) none of these

Answer: (c)

8. What separates the physical aspects of data storage from the logical aspects of data representation? [WBUT 2010]
   a) Data b) Schema c) Constraints d) Relationship

Answer: (b)

9. What schema defines how and where the data are organized in a physical data storage? [WBUT 2010]
a) External b) Internal c) Conceptual d) None of these
Answer: (c)

10. In ER model  symbol is used for [WBUT 2012]
a) attribute b) entity c) relation d) none of these
Answer: (a)

11. What is the cardinality of a table with 1000 rows & 10 columns? [WBUT 2012]
a) 10 b) 100 c) 1000 d) none of these
Answer: (c)

12. In the relational modes, cardinality is termed as [WBUT 2013, 2019]
a) number of tuples b) number of attributes
c) number of tables d) number of constraints
Answer: (a)

13. The different levels of data abstraction are [WBUT 2015]
a) physical level b) logical level c) view level d) all of these
Answer: (d)

14. Which key cannot be null? [WBUT 2016]
a) Unique key b) Primary key c) Super key d) Foreign key
Answer: (b)

15. In the E-R diagram the term 'Cardinality' is synonymous to [WBUT 2017]
a) Attribute b) Degree c) Entities d) Cartesian
Answer: (b)

16. It is an abstraction through which relationships are treated as higher level entities? [WBUT 2018]
a) generalization b) specialization
c) aggregation d) inheritance
Answer: (c)

Short Answer Type Questions

1. What is the difference between logical data independence and physical data independence? [WBUT 2006]

OR,

Define physical data independence and logical data independence. [WBUT 2015]

Answer:

Logical data independence: The ability to change the logical (conceptual) schema without changing the External schema (User View) is called logical data independence. For example, the addition or removal of new entities, attributes, or relationships to the

conceptual schema should be possible without having to change existing external schemas or having to rewrite existing application programs. The external scheme may stay unchanged for most changes of the logical scheme. This is especially desirable as the application software does not need to be modified or newly translated.

Physical data independence: Physical data independence deals with hiding the details of the storage structure from user applications. The application should not be involved with these issues, since there is no difference in the operation carried out against the data. The ability to change the physical schema without changing the logical schema is called physical data independence. For example, a change to the internal schema, such as using different file organization or storage structures, storage devices, or indexing strategy, should be possible without having to change the conceptual or external schemas.

2. Explain the terms Candidate key, Primary key, Foreign key and Super key.
[WBUT 2006, 2017]

OR,

Explain with example super key, candidate key and primary key.
[WBUT 2013, 2019]

OR,

Explain with examples the term super key, candidate key, primary key and alternate key.
[WBUT 2015]

Answer:

Candidate key

In the relational model of databases, a **candidate key** of a relation is a minimal superkey for that relation; that is, a set of attributes such that

1. the relation does not have two distinct tuples with the same values for these attributes (which means that the set of attributes is a superkey)
2. there is no proper subset of these attributes for which (1) holds (which means that the set is minimal).

Since a relation contains no duplicate tuples, the set of all its attributes is a superkey if NULL values are not used. It follows that every relation will have at least one candidate key.

The candidate keys of a relation tell us all the possible ways we can identify its tuples. As such they are an important concept for the design database schema.

Superkey: A **superkey** is defined in the relational model of database organization as a set of attributes of a relation variable (relvar) for which it holds that in all relations assigned to that variable there are no two distinct tuples (rows) that have the same values for the attributes in this set. Equivalently a superkey can also be defined as a set of attributes of a relvar upon which all attributes of the relvar arefunctionally dependent.

Note that if attribute set K is a superkey of relvar R, then at all times it is the case that the projection of R over K has the same cardinality as R itself.

Informally, a superkey is a set of columns within a table whose values can be used to uniquely identify a row. A candidate key is a minimal set of columns necessary to identify a row, this is also called a minimal superkey. For example, given an employee table, consisting of the columns employeeID, name, job, and departmentID, we could use the employeeID in combination with any or all other columns of this table to uniquely identify a row in the table. Examples of superkeys in this table would be {employeeID, Name}, {employeeID, Name, job}, and {employeeID, Name, job, departmentID}.

Foreign key: In the context of relational databases, a **foreign key** is a referential constraint between two tables. The foreign key identifies a column or a set of columns in one (referencing) table that refers to a set of columns in another (referenced) table. The columns in the referencing table must be the primary key or other candidate key in the referenced table. The values in one row of the referencing columns must occur in a single row in the referenced table. Thus, a row in the referencing table cannot contain values that don't exist in the referenced table (except potentially NULL). This way references can be made to link information together and it is an essential part of database normalization. Multiple rows in the referencing table may refer to the same row in the referenced table. Most of the time, it reflects the one (master table, or referenced table) to many (child table, or referencing table) relationship.

The referencing and referenced table may be the same table, i.e. the foreign key refers back to the same table. Such a foreign key is known in SQL: 2003 as a **self-referencing** or **recursive** foreign key.

A table may have multiple foreign keys, and each foreign key can have a different referenced table. Each foreign key is enforced independently by the database system. Therefore, cascading relationships between tables can be established using foreign keys. Improper foreign key/primary key relationships or not enforcing those relationships are often the source of many database and data modeling problems.

Primary Key: The **primary key** of a relational table uniquely identifies each record in the table. It can either be a normal attribute that is guaranteed to be unique (such as Social Security Number in a table with no more than one record per person) or it can be generated by the DBMS (such as a globally unique identifier, or GUID, in Microsoft SQL Server). Primary keys may consist of a single attribute or multiple attributes in combination.

Alternate key: In the candidate key the key which is not primary key is the alternate key.

Example:

CUSTOMER {custno, Name, Address}

ORDERS {Order No, Order Date, custno, part-no, project-no}

Here, the relation CUSTOMER having the key attribute custno, the **ORDERS** relation holds the custno as a non key attribute.

Thus custno in ORDERS is a foreign key w.r.t CUSTOMER

3. Explain the difference between weak entity set and strong entity set. [WBUT 2006, 2010]

Answer:

Strong entity sets	Weak entity sets
1) An entity set, which has a primary key, is termed as strong entity set.	1) An entity set is called a weak entity set if its existence depends on other entities i.e., strong entities.
2) A strong entity set have sufficient attributes to form a primary key.	2) A weak entity set does not have sufficient attributes to form a primary key.
3) A strong entity set is indicated in E-R diagram by a singly outlined rectangle.	3) A weak entity set is indicated in E-R diagrams by a doubly outlined rectangle.
4) The primary key of strong entity set is formed by the primary key of the strong entity set.	4) The primary key of weak entity set is formed by the primary key of the strong entity set plus its discriminator.

4. a) What is metadata? [WBUT 2007, 2010, 2017]

b) What do you mean by data dictionary? Why is the data dictionary required In RDBMS? [WBUT 2007]

OR,

What is Data dictionary? [WBUT 2015, 2017, 2019]

Answer:

a) Metadata is defined as data providing information about one or more other pieces of data, such as:

- Means of creation of the data
- Purpose of the data
- Time and date of creation
- Creator or author of data
- Placement on a computer network where the data was created
- Standards used

For example, a digital image may include metadata that describes how large the picture is, the color depth, the image resolution, when the image was created, and other data. A text document's metadata may contain information about how long the document is, who the author is, when the document was written, and a short summary of the document.

Metadata is data. As such, metadata can be stored and managed in a database, often called a registry or repository. However, it is impossible to identify metadata just by looking at it because a user would not know when data is metadata or just data.

b) Data Dictionary

A **data dictionary** or metadata repository is a "centralized repository of information about data such as meaning, relationships to other data, origin, usage, and format.

A data dictionary is organized into five sections:

- Data elements
- Data Flows
- Data Stores
- Processes
- External Entities.

The format of the data dictionary contains:

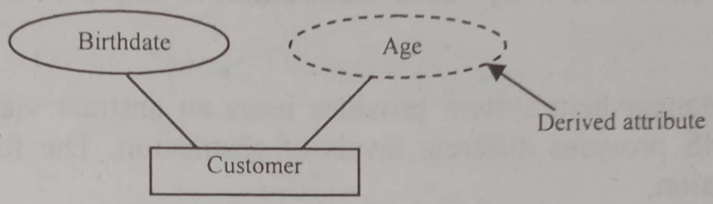
- ***Data Type:*** Data Element/Data Flow/Data Store
- ***Data Name:*** Name of the Data Elements, Data Flow/Data Store.
- ***Data Aliases:*** Alternate names used for the convenience of multiple users.
- ***Data Description:*** A short description of data.
- ***Data characteristics:*** Frequency of the use. Data length, Range of data values etc.
- ***Data Composition:*** Various data elements contained in a data store of data flow.

5. Give an example of derived attribute. **[WBUT 2009, 2010]**

Answer:

Derived Attribute

The attribute values which are not a part of the entity but can be computed in association with some other stored attributes.



For example the age of a person which can be computed by subtracting the dob from the current date.

6. Explain candidate key with an example. **[WBUT 2011]**

Answer:

Refer to Question No. 2 of Short Answer Type Questions.

7. Define super key, candidate key and primary key. **[WBUT 2012]**

Answer:

Refer to Question No. 2 of Short Answer Type Questions.

8. What is cardinality ratio? **[WBUT 2013, 2017]**

Answer:

Cardinality: Cardinality is the specification of the number of occurrences of one object that can be related to the number of occurrences of another object.

For example, one object can relate to only one other object (1:1 relationship); one object can relate to many objects (1:N relationship); Some number of occurrences of an object can relate to some other number of occurrences of another object (M:N relationship). Cardinality defines "the maximum number of objects that can participate in a relationship". However, it does not provide an indication of whether or not a particular data object must participate in the relationship. To specify this information, the data model adds modality to the object/ relationship pair.

9. What is disjointless constraint? **[WBUT 2013]**

Answer:

Disjoint Constraints: One or more sub collections of a given super collection may be declared to be disjoint. It implies that no object may occur in more than one of these sub

collections. E.g. Employees and Customers may be declared to be disjoint sub collections of Persons. Such that they can't be both an employee and a customer.

10. What do you mean by degree, cardinality of relationship? **[WBUT 2015]**

Answer:

The **degree of relationship** (also known as cardinality) is the number of occurrences in one entity which are associated (or linked) to the number of occurrences in another.

There are three degrees of relationship, known as:

1. one-to-one (1:1)
2. one-to-many (1:M)
3. many-to-many (M:N)

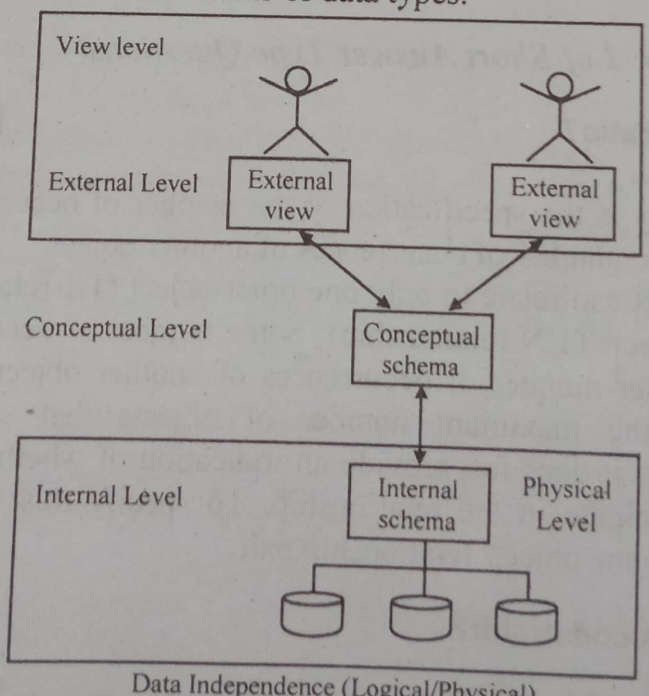
11. What do you mean by data abstraction? Explain three levels of data abstraction? **[WBUT 2015]**

Answer:

The database Management system provides users an abstract view of data. For users simplicity DBMS provides different levels of abstraction. The following are different levels of abstraction.

1. Physical Level
2. Logical Level
3. View Level

Physical Level is the lowest level of abstraction and it defines the storage structure. Logical Level is the next higher level of abstraction which describes what data are stored in database, relation between data, types of data etc. View Level is the highest level of abstraction. It provides different view to different users. At the view level users see a set of application programs that hide details of data types.



Data Independence (Logical/Physical)

12. What is Weak entity set? Explain with suitable example. **[WBUT 2016]**

Answer:

An entity set is called a weak entity set if its existence depends on other entities (called strong entities). Thus a weak entity set does not have sufficient attributes to form a primary key.

As shown in fig. the entity set Departures which has the only attribute date is a weak entity set. This is because of the fact that although each departure entity is distinct, different flights may have departure on the same day. As a result different flights may share the same date value. A weak entity set is indicates in E-R diagrams by a doubly outlined rectangle.

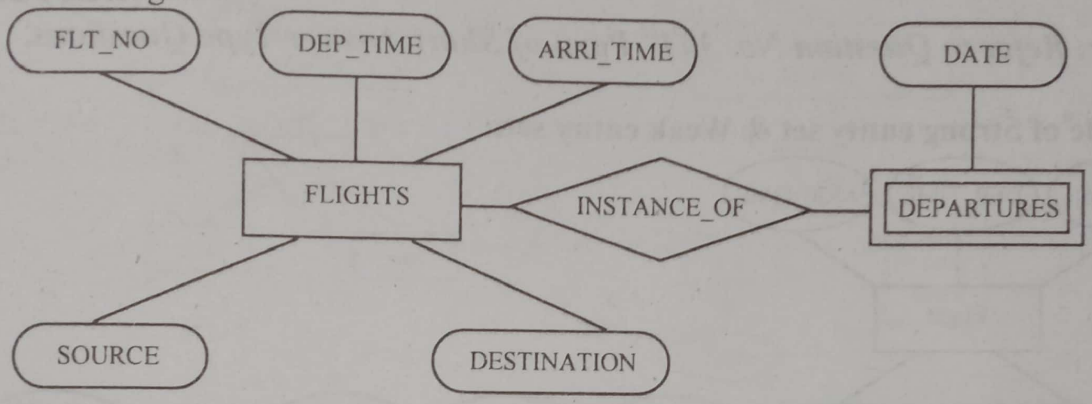


Fig: E-R diagram for a weak entity set which is dependent on a strong entity set

13. Describe different types of attribute. **[WBUT 2017]**

Answer:

Types of Attribute are:

Single valued Attributes: An attribute, that has a single value for a particular entity. For example, age of a employee entity.

Multi valued Attributes: An attributes that may have multiple values for the same entity. For example colors of a car entity.

Compound /Composite Attribute: Attribute can be subdivided into two or more other Attribute. For Example, Name can be divided into First name, Middle name and Last name.

Stored Attribute: An attribute, which cannot be derived from other attribute, is known as stored attribute. For example, BirthDate of employee.

Derived Attribute: Attributes derived from other stored attribute. For example age from Date of Birth and Today's date.

Key Attribute: represents primary key. (main characteristics of an entity). It is an attribute, that has distinct value for each entity/element in an entity set. For example, Roll number in a Student Entity Type.

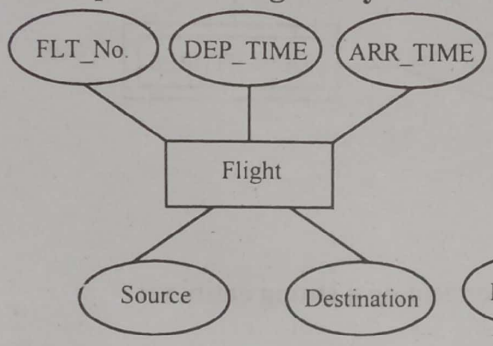
14. "Primary keys are candidate keys but reverse is not true." – explain. **[WBUT 2018]**

Answer:
Super Key: An attribute or set of attributes that uniquely identifies a tuple within a relation
Candidate key: A super key such that no proper subset is a super key within the relation
Primary key: The candidate key that is selected to identify tuples uniquely within the relation, the candidate keys which are not selected as PKs are called "Alternate keys". So the reverse is Alternate keys.

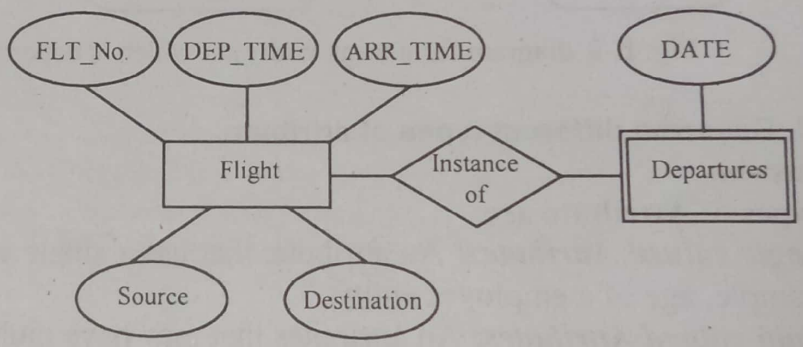
15. Explain with example the difference between strong and weak entity sets. **[WBUT 2019]**

Answer: ***Refer to Question No. 3 (1st Part) of Short Answer Type Questions.***

Example of Strong entity set & Weak entity set:



Flight is a strong entity.

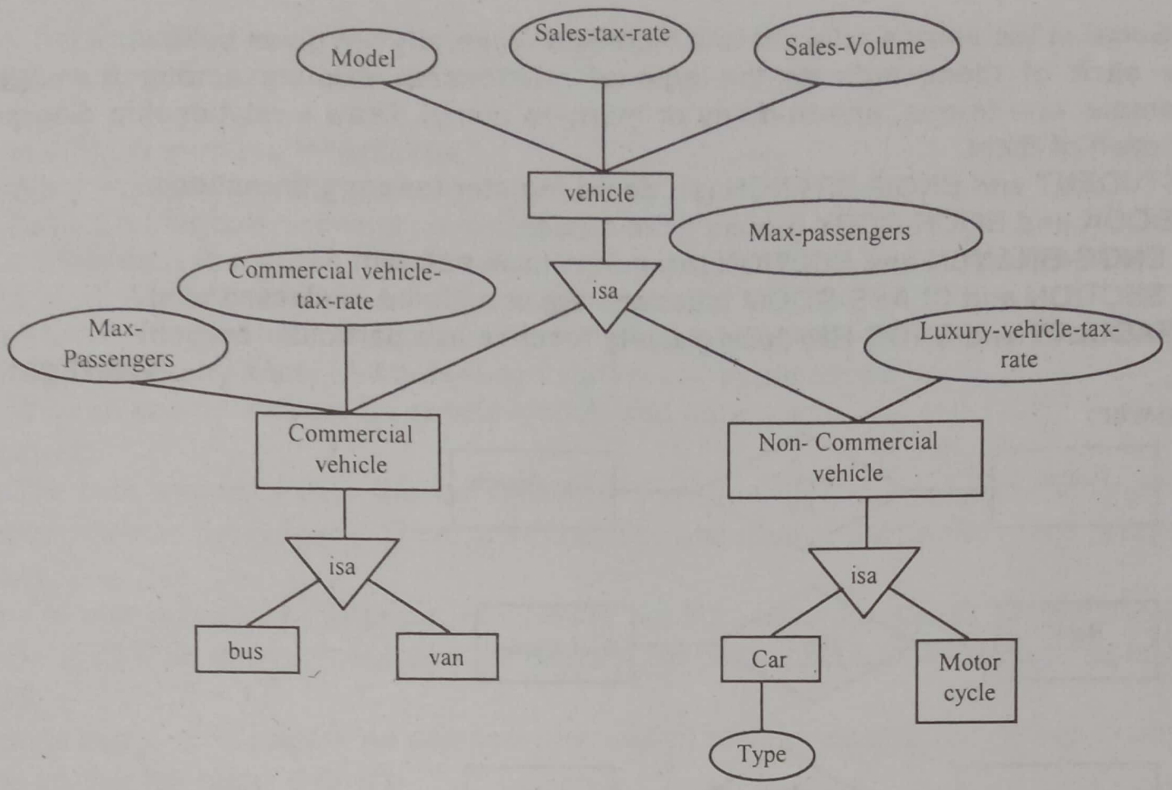


Departures is a weak entity.

Long Answer Type Questions

1. Design a Generalization-Specialization hierarchy for a motor-vehicle sales company. The company sells motor-cycles, passenger cars, vans, buses. Justify your placement of attributes at each level of the hierarchy. **[WBUT 2010, 2011]**

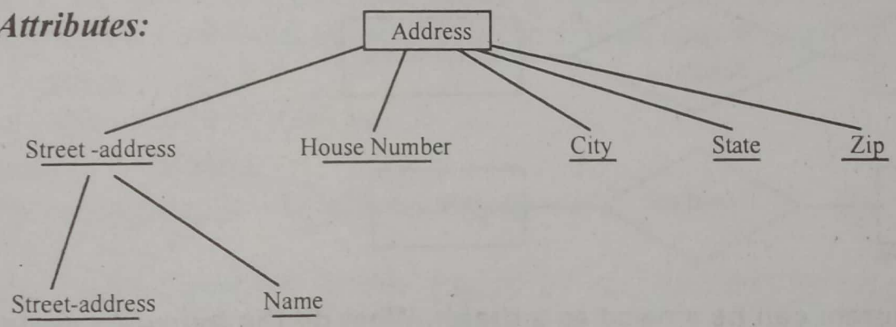
Answer:
The generalization–specialization hierarchy for the motor-vehicle company is given in the figure. *model*, *sales-tax-rate* and *sales-volume* are attributes necessary for all types of vehicles. Commercial vehicles attract commercial vehicle tax, and each kind of commercial vehicle has a passenger carrying capacity specified for it. Some kinds of non-commercial vehicles attract luxury vehicle tax. Cars alone can be of several types, such as sports-car, sedan, wagon etc. hence the attribute type.



2. a) What do you mean by composite attribute and derived attribute? Give example. **[WBUT 2011]**

Answer:

Composite Attributes:

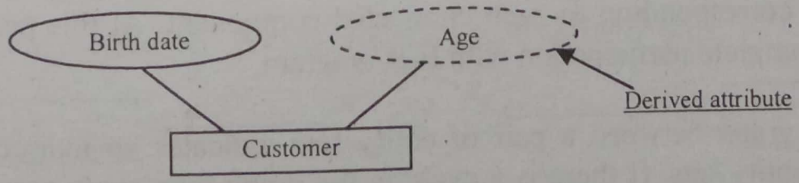


Composite Attribute

Can be divided into smaller subparts, which contains independent meaning. The address attribute of the employee relation can be divided into House Number, Street Address, City, State and zip.

Derived Attribute:

The attribute values which are not a part of the entity but can be computed in association with some other stored attributes. It is not stored but can be computed as required.



For example the age of a person which can be computed by subtracting the dob from the current date.

b) Some of the entries relevant to a technical university are given bellow: For each of them, indicate the type of relationship existing among them (for example, one-to-one, one-to-many or many to many). Draw a relationship diagram for each of them.

i) STUDENT and ENGG-BRANCH (students register for engg branches)
ii) BOOK and BOOK-COPY (books have copies)
iii) ENGG-BRANCH and SECTION (branches have section)
iv) SECTION and CLASS-ROOM (sections are scheduled in classrooms)
v) FACULTY and ENGG-BRANCH (faculty teaches in a particular branch)

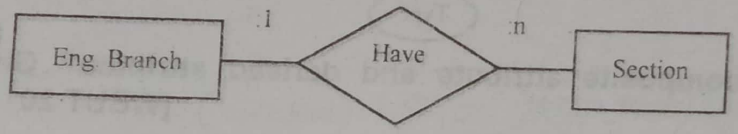
[WBUT 2011]

Answer:

i)

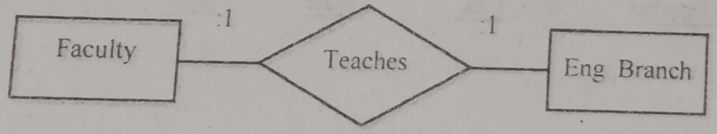
ii)

iii)



iv)

v)



c) An E-R diagram can be viewed as a graph. What do the following means in terms of the structure of an enterprise schema? **[WBUT 2011]**

i) The graph is disconnected
ii) The graph is cyclic.

Answer:

i) If pair of entity sets is connected by a path in an E-R diagram, the entity sets are related. A disconnected graph implies that there are pairs of entity sets that are unrelated to each other. If we split the graph into connected components, we have, in effect, a separate database corresponding to each connected component. At this point analogy is with partial and complete participation with E-R diagram.

ii) A path in the graph between a pair of entity sets indicates an indirect relationship between the two entity sets. If there is a cycle in the graph then every pair of entity sets on the cycle are related to each other at least in two distinct ways. If the E-R diagram is

acyclic then there is a unique path between every pair of entity sets and a unique relationship between every pair of entity sets and is a total participation only.

3. a) What is multiple inheritance? **[WBUT 2012]**
b) What is attribute inheritance?
c) Draw ER diagram showing cardinality:
i) A bill is sent to a customer. A customer may receive may receive many bills.
ii) A clerk works in a bank. A bank has many clerks.
iii) Students appear for seats in college. Each student can get almost one seat. A college has many seats. A student can sent many applications.
d) With an example describe specialization and generalization.

Answer:

a) The best way to answer this question is to look at each relationship that is possible between tables (or entities). There are three. The one to one, one to many and many to many.

One to one is easy. For example, one person can only have one social security number (SIN in US). So one table contains the name of the person and another table contains the SSN.

One to many - Here again we compare two values, each in separate tables. For example one teacher has many students.

Many to many - There's a simple way to look at this. Let's take our example from above. One teacher has many students. But isn't it also the case that one student has many teachers? So if we have two tables, one showing all the information about teachers and the other showing all the information about students. If we connect the table thus –

Teacher - 1 ◊ many – Student

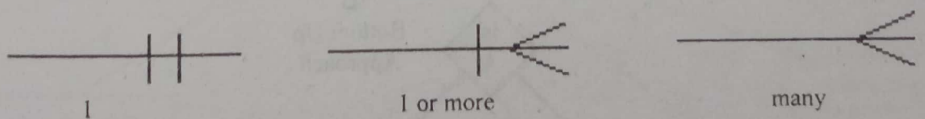
Then look at it from the other direction

Teacher - many ◊ 1 – Student

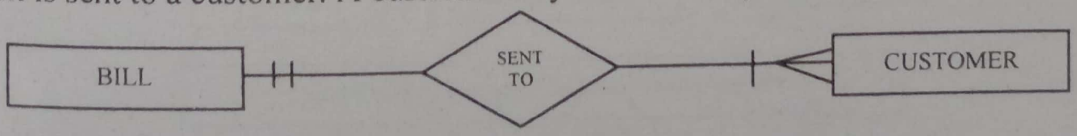
But really the relationship is – Teacher - many ◊◊ many - Student

b) The higher and lower level entities created by specialization and generalization is **attribute inheritance**. The lower level entities are inherited from the higher level entities. For example, the entities project-member and the entities project-manager are inherited form the entity person.

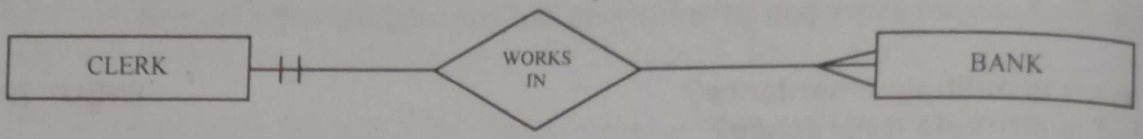
c)



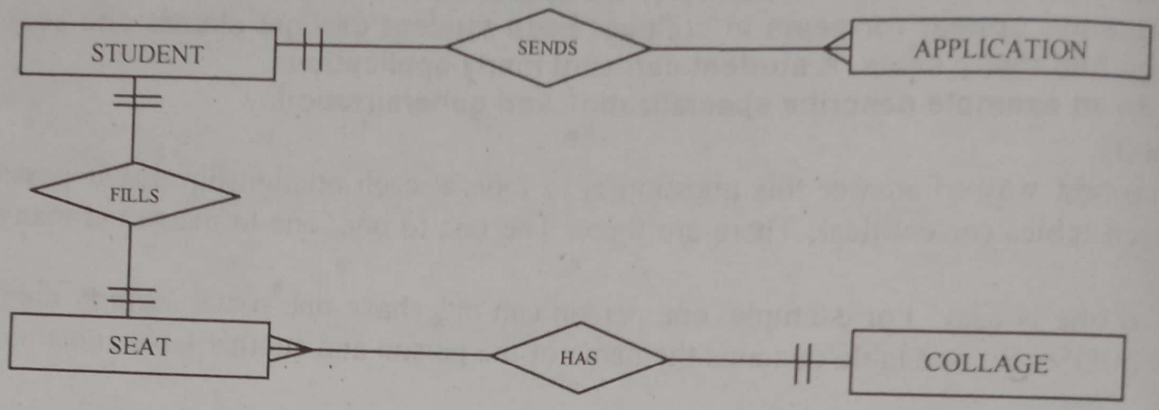
i) A bill is sent to a customer. A customer may receive many bills.



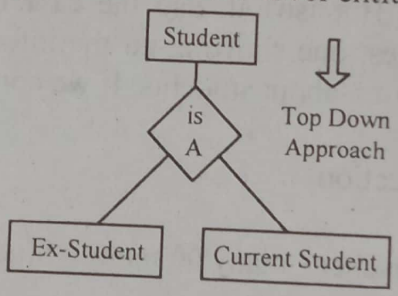
ii) A clerk works in a blank. The bank has many clerks.



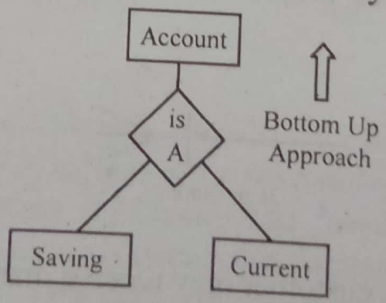
iii) Students appear for seats in collage. Each student can get almost one seat. A college has many seats. A student can sent many applications.



d) Specialization is opposite to Generalization. It is a top-down approach in which one higher level entity can be broken down into two lower level entity. In specialization, some higher level entities may not have lower-level entity sets at all.



Generalization is a bottom-up approach in which two lower level entities combine to form a higher level entity. In generalization, the higher level entity can also combine with other lower level entity to make further higher level entity.



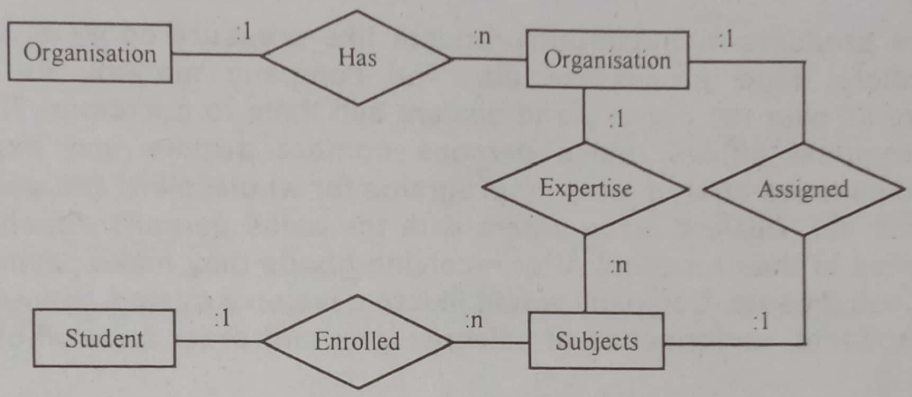
4. a) Draw an ER diagram for the system given as follows:
An organization has number of faculties who are expert in one or more subjects. For each subject, number of such experts is there, system will store faculty and subject information and must support query on finding expertise on subjects. Students get enrolled to have training on one or more subjects. System will keep student information also. One faculty is allotted to teach one or more subjects and for one subject only one faculty is assigned. System will keep the information regarding such assignment. **[WBUT 2014, 2019]**

b) Describe how the entity animal (in a ZOO) can be developed into a specialization hierarchy. **[WBUT 2014]**

c) What is meant by a recursive relationship type? Give one example of recursive relationship type. **[WBUT 2014]**

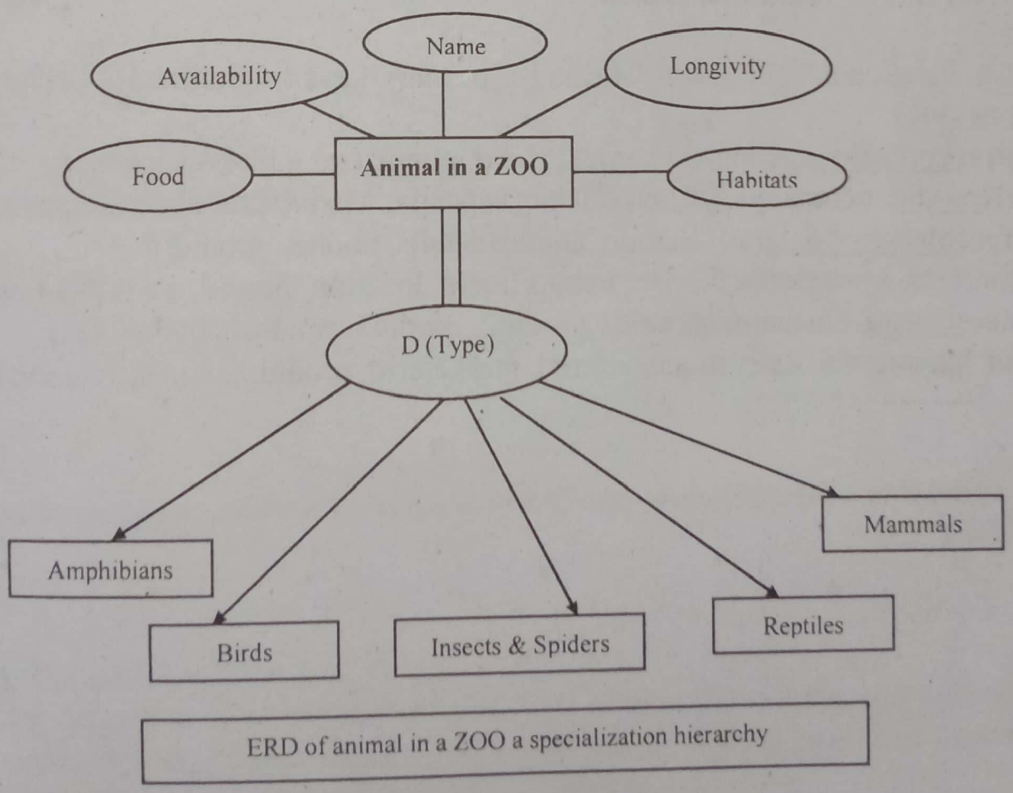
Answer:

a)



E-R diagram of the Organization-Faculty-Subject-Student

b)



ERD of animal in a ZOO a specialization hierarchy

c) A recursive relationship type exists if an entity can be related to itself or in other words, the same entity type participates more than once in a relationship type in different roles. An example of a recursive relationship type is courses that require one or more other courses as prerequisites. The course entity is related to another course entity. In this case, the recursive relationship "course is a prerequisite to course" also happens to be a M:N relationship. This is because a course can have more than one prerequisite, and a course can be a prerequisite to many other courses.

Another example of a recursive relationship type is in a supervision relationship type between an employee and a supervisor. Both entities are members of the same Employee entity type. In this example, the employee entity type participates twice in the supervision relationships, once in the role of a supervisor, and once in the role of supervisee.

5. Supreme products manufacture products like pressure cookers, cook wares, water purifiers, food processors etc. The company markets its product to wholesalers all over the country and dealers sell them to customer. The company has five regional offices. Sales persons contact dealers and explain about products, incentives offered, panting programs for wholesalers and demonstration for customer etc. Dealers place orders with the sales persons attached with the regional office of their location. After receiving goods they make payments, which may be in installments. Company would like to develop a system to monitor sale of different products, performance of salespersons and order from wholesalers. Do the following:

i) Identify entities, attributes and relationship.

ii) Draw an E-R diagram.

iii) Convert this to relational tables. **[WBUT 2015]**

Answer:

i) and iii) Relational Tables with attributes, primary keys (underlined), foreign keys are marked as (FK)

Company{companyid, companyname, Prodid, phone,fax, address,License}

Product{prodid, companyid(fk),prodname, capacity, Manufacturedate, inspectiondate}

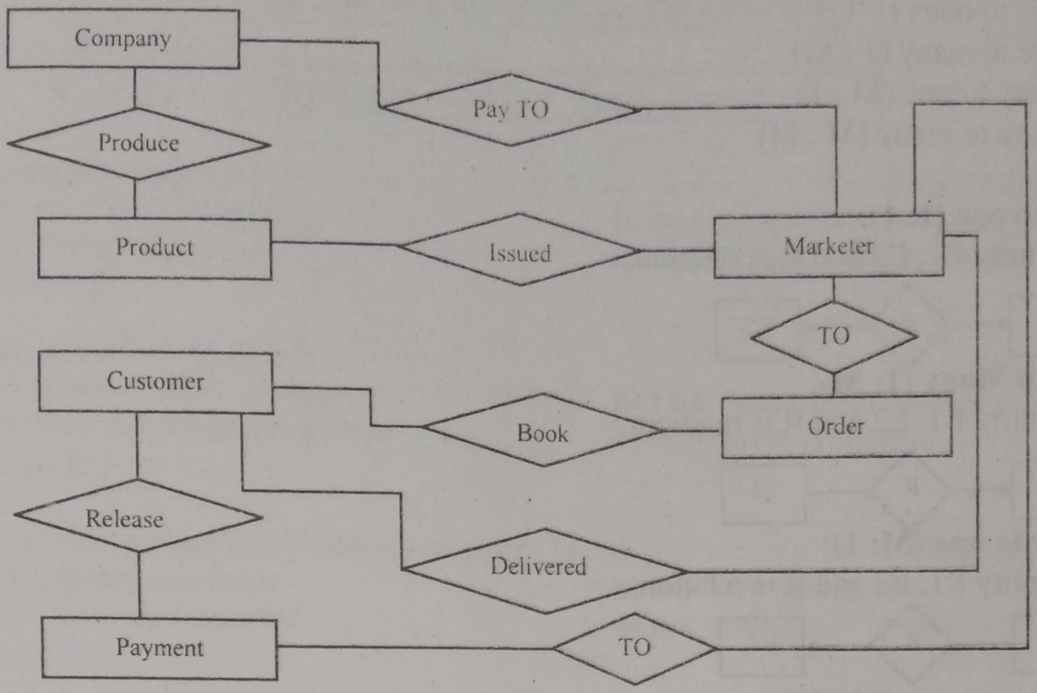
Marketer{Marketerid, type, location, contactdetails, phone, prodid(fk)}

Order{orderid, Marketerid(fk), Regionalofficeid, location, prodid, custid(Fk), orderdate}

Customer{Custid, custname, prodid1,prodid2, prodid3, prodid4, phone, fax}

Payment {paymentid, date, mode,ordered, marketerid, prodid, amount, remarks}

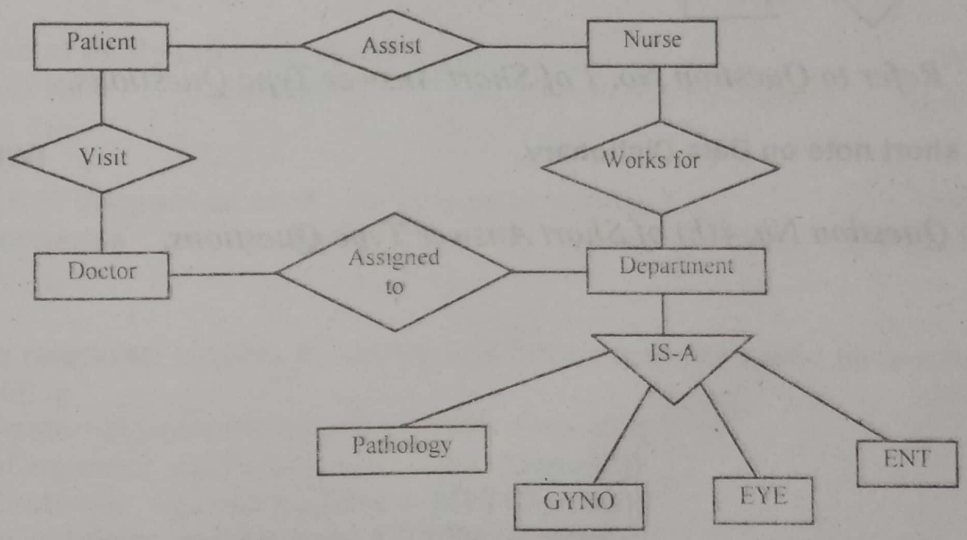
ii) E-R diagram



6. Draw the ER diagram of a hospital management system and explain.

[WBUT 2016]

Answer:



7. What is mapping constraint? Describe three-layer architecture of DBMS.

[WBUT 2019]

Answer:

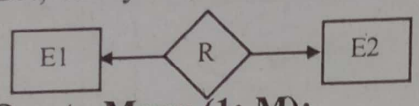
1st Part:

Mapping Constraint:

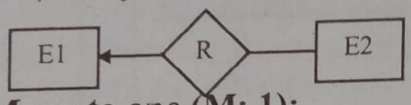
A mapping constraint is a data constraint that expresses the number of entitles to which another entity can be related via a relationship set. It is most useful in describing the relationship sets that involve more than two entity sets.

There are four possible mapping cardinalities. These are as follow:
1. One to one (1 : 1)
2. One to many (1 : M)
3. Many to one (M : 1)
4. Many to many (M : M)

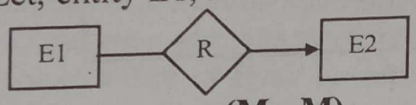
One to one (1: 1):
Let, entity E1, E2 and R is relation.



One to Many (1: M):
Let, entity E1, E2 and R is relation.



Many to one (M: 1):
Let, entity E1, E2 and R is relation.



Many to many (M : M)
Let, entity E1, E2 and R is relation.

2nd Part: ***Refer to Question No. 1 of Short Answer Type Questions.***

8. Write short note on Data Dictionary. **[WBUT 2013]**

Answer:
Refer to Question No. 4(b) of Short Answer Type Questions.

RELATIONAL MODEL

Multiple Choice Type Questions

1. Relational algebra is a [WBUT 2007, 2019]
   a) procedural language b) non-procedural language
   c) query language d) normalization technique

Answer: (a)

2. Relational calculus is a [WBUT 2013]
   a) procedural language b) non-procedural language
   c) data definition language d) high level language

Answer: (b)

3. Cartesian product in relational algebra is [WBUT 2013, 2019]
   a) a unary operator b) a binary operator
   c) a ternary operator d) not defined

Answer: (a)

4. In a relational model, relations are termed as [WBUT 2013]
   a) tuples b) attributes c) tables d) rows

Answer: (c)

5. In case of entity integrity, the primary key may be [WBUT 2013]
   a) not Null b) Null c) both Null & not Null d) any value

Answer: (a)

6. In an E-R diagram an entity set is represented by a [WBUT 2013]
   a) rectangle b) ellipse c) diamond box d) circle

Answer: (a)

7. Using relational algebra the query that finds customers, who have a balance of over 1000 is [WBUT 2013]
   a) Customer_name(σ balance > 1000 (Deposit))
   b) σCustomer_name(balance > 1000 (Deposit))
   c) Customer_name(σ balance > 1000 (Borrow))
   d) σCustomer_name(balance > 1000 (Borrow))

Answer: (a)

8. Which of the following operations is used if we are interested in only certain columns of a table? [WBUT 2013, 2019]
   a) PROJECTION b) SELECTION c) UNION d) JOIN

Answer: (a)

9. The strategy for processing a query is improved by [WBUT 2015, 2019]
a) query evaluation
b) decomposition
c) query optimization
d) none of these
Answer: (c)

10. Relational calculus is a [WBUT 2016]
a) Query language
b) procedural language
c) Non-procedural language
d) None of these
Answer: (a)

11. A row from a table is selected by [WBUT 2016]
a) selection operator
b) projection operator
c) union operator
d) none of these
Answer: (a)

12. According to the levels of abstractions the schema at the intermediate level is called [WBUT 2018]
a) logical schema
b) physical schema
c) subschema
d) super schema
Answer: (a)

13. The operation which is not considered as basic operation in relational algebra [WBUT 2018]
a) join b) selections c) union d) cross product
Answer: (a)

Short Answer Type Questions

1. Why we need query optimization? [WBUT 2008]
Answer:
Having long-running queries not only consumes system resources that makes the server and application run slowly, but also may lead to table locking and data corruption issues. So, query optimization (QO) becomes an important task. As there are many equivalent transformations of same high-level query, aim of QO is to choose one that minimizes resource usage. In general it also reduces total execution time of query. Since the problem is computationally intractable with large number of relations, so strategy adopted is reduced to finding near optimum solution.

2. Discuss the entity integrity and referential integrity constraints. Why is each considered important? Explain with suitable example. [WBUT 2008, 2010, 2012]
Answer:
Entity Integrity: In a relational database, entity integrity is a property that ensures that no records are duplicated and that no attributes that make up the primary key are NULL. It is one of the properties necessary to ensure the consistency of the database.
Entity Integrity ensures that there are no duplicate records within the table and that the field that identifies each record within the table is unique and never null. The existence of

the Primary Key is the core of the entity integrity. If one defines a primary key for each entity, they follow the entity integrity rule. Entity integrity specifies that the Primary Keys on every instance of an entity must be kept, must be unique and must have values other than NULL. Entity Integrity is the mechanism the system provides to maintain primary keys. The primary key serves as a unique identifier for rows in the table.
The system enforces Entity Integrity by not allowing operations (INSERT, UPDATE) to produce an invalid primary key. Any operation that creates a duplicate primary key or one containing nulls is rejected.

Referential Integrity: It is a database management safeguard that ensures every foreign key matches a primary key. For example, customer numbers in a customer file are the primary keys, and customer numbers in the order file are the foreign keys. If a customer record is deleted, the order records must also be deleted; otherwise they are left without a primary reference. If the DBMS does not test for this, it must be programmed into the applications.
Referential integrity in a relational database is consistency between coupled tables. Referential integrity is usually enforced by the combination of a primary key(candidate key) and a foreign key. For referential integrity to hold, any field in a table that is declared a foreign key can contain only values from a parent table's primary key or a candidate key. For instance, deleting a record that contains a value referred to by a foreign key in another table would break referential integrity. The RDBMS enforces referential integrity, normally either by deleting the foreign key rows as well to maintain integrity, or by returning an error and not performing the delete. Which method is used would be defined by the definition of the referential integrity constraint.

Example: An employee database stores the department in which each employee works. The field "DepartmentNumber" in the Employee table is declared a foreign key, and it refers to the field "Index" in the Department table which is declared a primary key. Referential integrity would be broken by deleting a department from the Department table if employees listed in the Employee table are listed as working for that department, unless those employees are moved to a different department at the same time.

3. Why are entity integrity and referential integrity important in a database?
[WBUT 2009, 2011]

Answer:
Refer to Question No. 2(1st Part) of Short Answer Type Questions.

4. consider the following "sailors" and "reserves" relations: [WBUT 2012, 2014]
R (sid, bid, day)
S (sid, sname, rating, age)
Formulate the following queries using relational algebra"
a) Find names of sailors who have reserved boat # XXX.
b) Find names and ages of sailors who have reserved a boat

Answer:

a) $\pi_{sname}(\sigma_{S.sid = R.sid}(S \times R))$.

b) $\pi_{sname, age}(\sigma_{S.sid = R.sid}(S \times R))$.

5. a) What is the disadvantage of Cartesian product and how to recover from it? **[WBUT 2014]**

Answer:

The Cartesian product, is referred to as a cross-join, returns all the rows in all the tables listed in the query. Each row in the first table is paired with all the rows in the second table. This happens when there is no relationship defined between the two tables. Say AUTHOR and STORE tables have ten rows each. If we use a Cartesian join in these two tables, we get 100 rows.

To get rid of this problem, most of the time, we provide a filter on the join. If we actually want a cartesian join, then we should use the ANSI cross join to tell others reading the script that we actually wanted a Cartesian join.

b) Explain full outer join, left outer join, right outer join with examples. **[WBUT 2014]**

Answer:

Before we proceed lets consider the table Company and Department

Company

id	name	age	address	salary	join_date
1	Paul	32	California	20000	2001-07-13
3	Teddy	23	Norway	20000	
4	Mark	25	Rich-Mond	65000	2007-12-13
5	David	27	Texas	85000	2007-12-13
2	Allen	25	Texas		2007-12-13
8	Paul	24	Houston	20000	2005-07-13
9	James	44	Norway	5000	2005-07-13
10	James	45	Texas	5000	2005-07-13

Department

id	dept	emp_id
1	IT Billing	1
2	Engineering	2
3	Finance	7

The LEFT OUTER JOIN

The OUTER JOIN is an extension of the INNER JOIN. SQL standard defines three types of OUTER JOINs: LEFT, RIGHT, and FULL and PostgreSQL supports all of these.

In case of LEFT OUTER JOIN, an inner join is performed first. Then, for each row in table T1 that does not satisfy the join condition with any row in table T2, a joined row is

added with null values in columns of T2. Thus, the joined table always has at least one row for each row in T1.

Following is the syntax of LEFT OUTER JOIN:

```
SELECT ... FROM table1 LEFT OUTER JOIN table2 ON
conditional_expression ...
```

Based on the above tables, we can write a inner join as follows:

```
testdb=# SELECT EMP_ID, NAME, DEPT FROM COMPANY LEFT OUTER JOIN
DEPARTMENT
        ON COMPANY.ID = DEPARTMENT.EMP_ID;
```

Above query will produce the following result:

```
 emp_id | name   | dept
--------+--------+------------
      1 | Paul   | IT Billing
      2 | Allen  | Engineering
        | James  |
        | David  |
        | Paul   |
        | Mark   |
        | Teddy  |
        | James  |
```

The RIGHT OUTER JOIN

First, an inner join is performed. Then, for each row in table T2 that does not satisfy the join condition with any row in table T1, a joined row is added with null values in columns of T1. This is the converse of a left join; the result table will always have a row for each row in T2.

Following is the syntax of LEFT OUTER JOIN:

```
SELECT ... FROM table1 RIGHT OUTER JOIN table2 ON
conditional_expression ...
```

Based on the above tables, we can write a inner join as follows:

```
testdb=# SELECT EMP_ID, NAME, DEPT FROM COMPANY RIGHT OUTER JOIN
DEPARTMENT
        ON COMPANY.ID = DEPARTMENT.EMP_ID;
```

Above query will produce the following result:

```
 emp_id | name   | dept
--------+--------+---------
      1 | Paul   | IT Billing
      2 | Allen  | Engineering
      7 |        | Finance
```

The FULL OUTER JOIN

First, an inner join is performed. Then, for each row in table T1 that does not satisfy the join condition with any row in table T2, a joined row is added with null values in columns of T2. Also, for each row of T2 that does not satisfy the join condition with any row in T1, a joined row with null values in the columns of T1 is added.
Following is the syntax of FULL OUTER JOIN:

```
SELECT ... FROM table1 FULL OUTER JOIN table2 ON
conditional_expression ...
```

Based on the above tables, we can write a inner join as follows:

```
testdb=# SELECT EMP_ID, NAME, DEPT FROM COMPANY FULL OUTER JOIN
DEPARTMENT
        ON COMPANY.ID = DEPARTMENT.EMP_ID;
```

Above query will produce the following result:

```
 emp_id | name   | dept
--------+--------+---------------
      1 | Paul   | IT Billing
      2 | Allen  | Engineering
      7 |        | Finance
        | James  |
        | David  |
        | Paul   |
        | Mark   |
        | Teddy  |
        | James  |
```

6. What do you mean by unary and binary operations in Relational algebra? Give example. **[WBUT 2015, 2016]**

Answer:

Unary operator (one relation as operand)
Returns subset of the tuples from a relation that satisfies a selection condition: σ<*selection condition*>R
RENAME operator
Rename relation ρSR
Binary Relational Operations: JOIN and DIVISION
$R \bowtie$<*join condition*>S
where **join condition** is a Boolean expression involving attributes from both operand relations
$R \div S$
Attributes of S must be a subset of the attributes of R
attr($R \div S$) = attr(R) – attr(S)
t tuple in ($R \div S$) iff ($t \times S$) is a subset of R

For example:

P:

A	B
a1	b1
a1	b2
a2	b1
a3	b1
a4	b2
a5	b1
a5	b2

Q1:

B
b1
b2

Q2:

B
b1

Q3:

B
b1
b2
b3

Q4:

B

3 $P \div Q3 = R$:

A

4. $P \div Q4 = R$:

A
a1
a2
a3
a4
a5

7. Consider the following database with primary keys underlined
Project (P – No, P – Name, P – incharge)
Employee (E – no, E – Name)
Assigned–To (E – no, P – no)
Write relational algebra expression for the following:
(a) List detail of employee working on all projects.
(b) List E – no, E – name of employee who do not work on project with P – no = DB2003. **[WBUT 2018]**

Answer:

a) $K = \pi_{E-no,\ E-Name,\ P-no}(Employee \underset{E-no}{\overset{\theta}{\bowtie}} Assigned\text{–}To)$

$$\pi_{E-no,\ E-Name,\ P-no,\ P-Name}(Project \underset{P-no}{\overset{\theta}{\bowtie}} K)$$

b) $K = \pi_{E-no,\ E-Name,\ P-no}(Employee \underset{E-no}{\overset{\theta}{\bowtie}} Assigned\text{–}To)$

$$\pi_{E-no,\ E-Name,\ P-no,\ P-Named}\ \sigma_{P-no<>'DB2003'}(Project \underset{P-no}{\overset{\theta}{\bowtie}} K)$$

8. What is referential integrity? **[WBUT 2019]**

Answer:

Refer to Question No. 2 (1st Part) of Short Answer Type Questions.

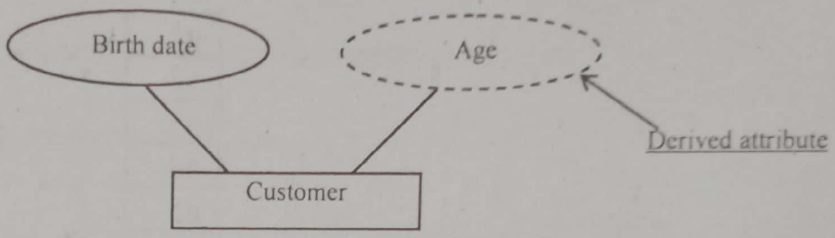
9. Define the following: [MODEL QUESTION]

i) Derived attribute
ii) Domain
iii) 1 to Many relationship
iv) Projection
v) Union compatibility

Answer:

i) *Derived Attribute:*

The attribute values which are not a part of the entity but can be computed in association with some other stored attributes. It is not stored but can be computed as required.



For example the age of a person which can be computed by subtracting the dob from the current date.

ii) *Domain:*

Each simple attribute of an entity type is associated with a value set, which specifies the set of values that may be assigned to that attribute for each individual entity.
e.g., If the range of ages allowed for employees is between 16 to 70. We can say that the domain of age attribute in employee table is (16–70).

iii) *One-to-many:*

1: M, exists when one occurrence of the first entity can be related to many occurrences of the second entity, but each occurrence of the second entity can be associated with only one occurrence of the first entity.

iv) *Projection Operator:* The projection operation projects all or an attribute or a set of selective attributes from a relation. It is a vertical subset of the given relation. The operation is symbolized by π.
The syntax of the projection query is:

$$\pi_{\text{<List of attribute names/ offset position>}} (\text{Relation Name})$$

For example, let us execute the following queries using the relation **Employee**
Find the employees whose EMP# is greater than 100 and less than 104 and Working under the manager Patel and show all the attributes

$$\Pi_{\text{emp\#, Name, Dept, Manager}}(\sigma_{\text{Emp>100 ^ Emp<104^Manager='Patel'}} (\text{EMPLOYEE}))$$

v) The operations union, difference, intersection are union compatible. Any two relations A and B are said to be union compatible if A and B are of the same degree say 'n 'and the domain of the corresponding n attributes are identical.
i.e. if $A = A\{a_1, a_2, \ldots\ldots, a_n\}$ and

$B=B\{b_1,b_2,\ldots\ldots,b_n\}$ then $Dom(A_i)=Dom(B_i)$ for $\{i=1,2,3,\ldots..,n\}$, where $Dom(A_i)$ represents the domain of the attribute A_i.

1. Union-compatible relation:

   $R=A \cup B$ s.t $R=\{t \mid t\in A \vee t \in B\}$

2. Difference:

   $R=A-B$ s.t. $R=\{t \mid t\in A \wedge t \notin B\}$

3. Intersection:

   $R=A\cap B$ s.t $R=\{t \mid t\in A \wedge t \in B\}$

4. Cartesian Product:

   $R=A \times B$ s.t $R=\{t1 \| t2 \mid t1\in A \wedge t2 \in B\}$

Difference and Intersection are childsets of union, thus on operations upon a table must be **union compatible**.

Relational Algebra is Complete:

The set of relational algebra operators $\{\sigma,\pi,\cup,-,\times\}$ is a complete set. Any other relational algebra operations can be expressed as a sequence of operations from this set. Let us see the equivalent expression of intersection is,

$R \cap S = R \cup S - ((R - S) \cup (S - R))$

10. Explain how a relational calculus expression can be unsafe. Illustrate your answer with an example. **[MODEL QUESTION]**

Answer:

1. A tuple relational calculus expression may generate an infinite expression. e.g. $\{t|\neg(t\in \textbf{Student})\}$

2. There are an infinite number of tuples that are not in ***student.*** Most of these tuples contain values that don't appear in the database.

Safe Tuple Expressions: The relational calculus domain of a formula **P**, denoted dom **(P)**, is the set of all values referenced in **P**.

These include values mentioned in **P** as well as values that appear in a tuple of a relation mentioned in **P**. So, the domain of **P** is the set of all values explicitly appearing in **P** or that appear in relations mentioned in **P**.

$dom(t \in borrow \wedge t[amount] < 1200)$ is the set of all values appearing in *borrow*.

$dom(t|\neg(t \in borrow))$ is the set of all values appearing in *borrow*.

We may say an expression $\{t|P(t)\}$ is **safe** if all values that appear in the **result** are values from dom**(P)**. A **safe** expression yields a finite number of tuples as its result. Otherwise, it is called **unsafe**.

Long Answer Type Questions

1. What is query optimization? Write down the steps to process a high level query. [WBUT 2009]

Answer:

1st Part: *Refer to Question No. 1 of Short Answer Type Questions.*

2nd Part:

Overview of Query Processing:

The main steps in processing a high-level query are illustrated in figure 1.

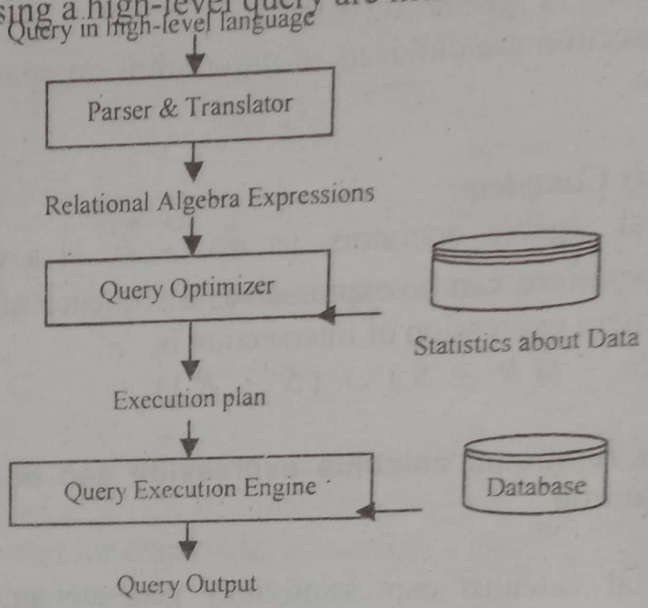


Fig: Steps in query processing process

The functions of Query Parser is parsing and translating a given high-level language query into its immediate form such as relational algebra expressions. The parser need to check for the syntax of the query and also check for the semantic of the query (it means verifying the relation names, the attribute names in the query are the names of relations and attributes in the database). A parse-tree of the query is constructed and then translated into relational algebra expression.

A relational algebra expression of a query specifies only partially how to evaluate a query, there are several ways to evaluate an relational algebra expression. For example, consider the query:

SELECT Salary FROM EMPLOYEE WHERE Salary > = 50,000 ;

The possible relational algebra expressions for this query are:

- $\Pi_{Salary}(\sigma_{Salary>=50000}(EMPLOYEE))$
- $\sigma_{Salary>=50000}(\Pi_{Salary}EMPLOYEE)$

Further, each relational algebra operation can be executed using various algorithms. For example, to implement the preceding selection, we can do a linear search in the EMPLOYEE file to retrieve the tuples with Salary >= 50000. However, if an index available on the Salary attribute, we can use the index to locate the tuples. Different algorithms might have different cost.

Thus, in order to specify fully how to evaluate a query, the system is responsible for constructing a query execution plan which made up of the relational algebra expression and the detailed algorithms to evaluate each operation in that expression. Moreover, the selected plan should minimize the cost of query evaluation. The process of choosing a suitable query execution plan is known as query optimization this process is perform by Query Optimizer.

One aspect of optimization occurs at relational algebra level. The system attempt to find an expression that is equivalent to the given expression but that is more efficient to execute. The other aspect involves the selection of a detail strategy for processing the query, this relates to choosing the processing algorithm, choosing the indices to use and so on.

Once the query plan is chosen, the Query Execution Engine lastly take the plan, executes that plan and return the answer of the query.

2. Consider the relational database as given below and write down expressions in relational algebra for the following queries.
Material_Master (item_id, item name, reorder level)
Material_Dts (item_id, Supplier_id, Pharchase_date, Qty, Utcost)
i) Select the quantities of each purchased material alphabetically.
ii) Select the names of materials which have the highest total quantity.
iii) Replace the material name 'power supply' with 'UPS'. **[WBUT 2010]**

Answer:

i) $\tau_{\text{item\_name}}(\pi_{\text{item\_name, Qty}}\sigma_{\text{item\_id=item\_id}}(\textbf{Material\_Master} \bowtie \textbf{Material\_Dts})$.

ii) Select m.item_name, Max(sum(d.qty)) from **Material-Master m, Material-Dts d where m.** item-id=d. item-id group by d. item-id ;

$\text{item-id}\ \chi_{\text{sum(Qty)}}\text{item\_name}\ (\sigma_{\text{item\_id=item\_id}}(\textbf{Material\_Master} \bowtie \textbf{Material\_Dts}))$

iii) $\pi_{\textbf{item\_id, reorder level}, \text{item\_name} \leftarrow \text{'UPS'}}\ (\sigma_{\text{item\_name='power supply'}}(\text{material\_master}))$

3. Write short notes on the following:
a) Theta (θ) join **[WBUT 2009]**
b) Database models **[WBUT 2016]**
c) Inner join and Outer join **[WBUT 2016]**
d) Tuple Relational Calculus **[MODEL QUESTION]**
e) Degree of relationship **[MODEL QUESTION]**

Answer:

a) Theta (θ) join:

The theta join operation is an extension to the natural-join operation that allows us to combine selection and a Cartesian product into a single operation. Consider relations *r(R)* and *s(S)*, and let _ be a predicate on attributes in the schema *R* È *S*. The theta join operation $r \bowtie \theta s$ is defined as follows:

$$r \bowtie \theta s = \sigma\theta(r \times s)$$

b) Database models:

Depending on data abstract levels and actual applications, different database models have their advantages and disadvantages. This is the reason why there exist a lot of database models, conceptual ones and logical ones. It is not appropriate to state that one database model is always better than the others. Conceptual data models are generally used for engineering information modeling at a high level of abstraction. However, engineering information systems are constructed based on logical database models. So at the level of data manipulation, that is, a low level of abstraction, the logical database model is used for engineering information modeling. Here, logical database models are often created through mapping conceptual data models into logical database models. This conversion is called *conceptual design of databases*. The relationships among conceptual data models, logical database models and engineering information systems are shown in figure. In this figure, *Logical DB Model (A)* and *Logical DB Model (B)* are different database systems. That means that they may have different logical database models, say relation database and object-oriented database or they may be different database products, say *Oracle*™ and *DB2*, although they have the same logical database model. It can be seen from the figure that a developed conceptual data model can be mapped into different logical database models. Besides, it can also be seen that a logical database model can be mapped into a conceptual data model. This conversion is called *database reverse engineering*. It is clear that it is possible that different logical database models can be converted one another through database reverse engineering.

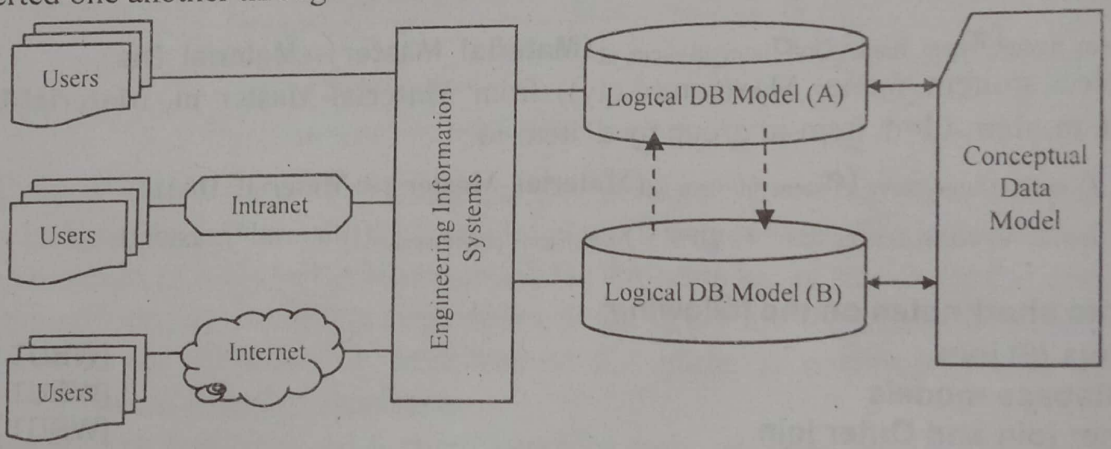
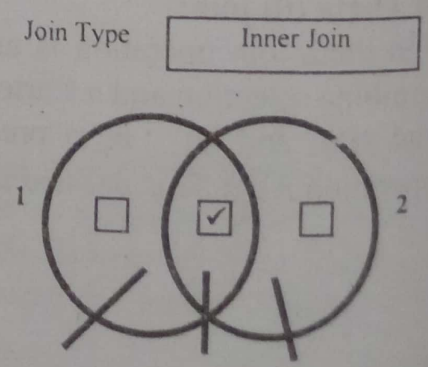


Fig: Relationships among conceptual data model, logical database model and engineering information systems

c) Inner join:

An inner join of 1 and 2 gives the result of 1 intersect 2, i.e. the inner part of a Venn diagram intersection.



STUDENTS		
student_ID	student_Name	advisor-Id
1	student-1	1
10	student-10	3
2	student-2	8
4	student-4	2
5	student-5	3
7	student-7	3
9	student-9	1

advisors	
advisor_ID	advisor_Name
1	advisor-1
3	advisor-3
5	advisor-5

Corresponding SQL statement:

```
SELECT S.student_ID, S.student_Name, S.advisor-Id, A.advisor_Name
FROM STUDENTS S INNER JOIN advisors A ON S.advisor-
Id=A.advisor_ID;
Output is
```

Query-Output			
student_ID	student_Name	advisor-Id	advisor_Name
1	student-1	1	advisor-1
9	student-9	1	advisor-1
5	student-5	3	advisor-3
7	student-7	3	advisor-3
10	student-10	3	advisor-3

Outer join: ***Refer to Question No. 5(b) of Short Answer Type Questions.***

d) Tuple relational calculus:

The tuple relational calculus is a non procedural language (declarative language). It describes the desired information without giving a specific procedure for obtaining that information. A query in the tuple relational calculus is expressed as {t | P(t)} i.e. it is the set of all tuples t such that predicate P is true for t. P is a well formed formula. A well-formed formula consists of a set of atoms, logical connectives and quantifiers connect those items.

The tuple-oriented calculus uses a tuple variables i.e., variable whose only permitted values are tuples of that relation.

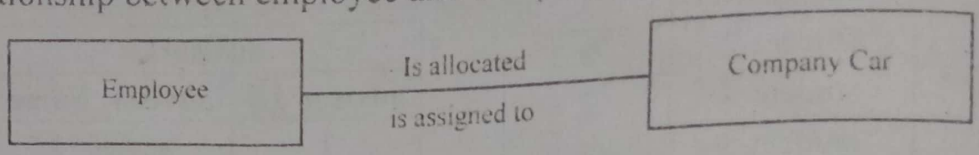
e) Degree of relationship:

The **degree of relationship** (also known as cardinality) is the number of occurrences in one entity which are associated (or linked) to the number of occurrences in another.

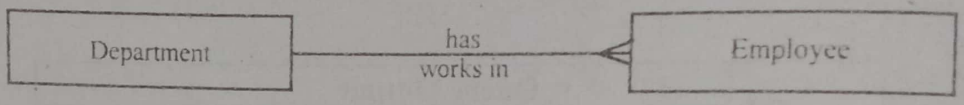
There are three degrees of relationship, known as:

one-to-one (1:1)
one-to-many (1:M)
many-to-many (M:N)

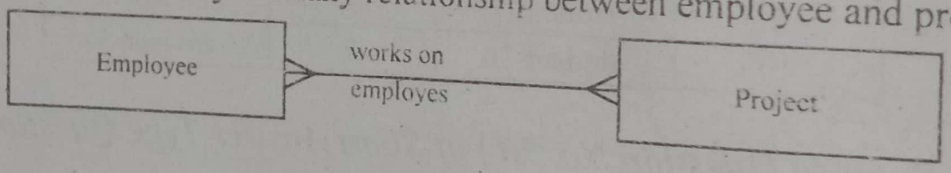
- **One-to-one (1:1) Relationship:** This is where one occurrence of an entity relates to only one occurrence in another entity. For example, an employee is allocated a company car, which can only be driven by that employee. Therefore, there is a one-to-one relationship between employee and company car.



- **One-to-many (1:M) Relationship:** This is where one occurrence in an entity relates to many occurrences in another entity. For example, taking the employee and department entities shown on the previous page, an employee works in one department, but a department has many employees. Therefore, there is a one-to-many relationship between department and employee.



- **Many-to-many (M:N) Relationship:** This is where many occurrences in an entity relate to many occurrences in another entity. For example, an employee may work on several projects at the same time and a project has a team of many employees. Therefore, there is a many-to-many relationship between employee and project.



SQL

Multiple Choice Type Questions

1. View is a [WBUT 2007, 2015]
   a) temporary table b) virtual table c) SQL statement d) query

Answer: (b)

2. Four DML commands are [WBUT 2009, 2012]
   a) CREATE, UPDATE, DELETE, SELECT
   b) INSERT, UPDATE, DROP, SELECT
   c) INSERT, MODIFY, DELETE, SELECT
   d) INSERT, UPDATE, DELETE, SELECT

Answer: (c)

3. SELECT operation in SQL is a [WBUT 2010, 2012, 2017]
   a) data query language b) data definition language
   c) data manipulation language d) data control language

Answer: (c)

4. A trigger is [WBUT 2011]
   a) a statement that enables to start any DBMS
   b) a statement that is executed by the user when debugging an application program
   c) a condition the system tests for the validity of the database user
   d) a statement that is executed automatically by the system of a modification

Answer: (d)

5. One difference between TRUNCATE and DELETE command is [WBUT 2011]
   a) TRUNCATE deletes the table but DELETE only deletes records
   b) DELETE operation can be rolled back, bud TRUNCATE operation cannot be rolled back
   c) TRUNCATE can be rolled back but DELETE cannot be rolled back
   d) TRUNCATE is a DML command but DELETE is a DDL command

Answer: (b)

6. Consider the primary key foreign key relationship between Employee and Department table via Dept ID column. If you try to delete a department in Department table which is referred in Employee table, Oracle by default does not allow this. This is known as ON DELETE ..... [WBUT 2011]
   a) SET CASCADE b) SET DEFAULT
   c) RESTRICT d) SET NULL

Answer: (a)

7.

EMPNO	ENAME	SAL
A822	RAMASWAMY	3500
A812	NARAYAN	5000
A973	UMESH	2850
A500	BALAJI	5750

Use these data for the following questions.
Select SAL from EMP E1 where 3 > (select count (*) from Emp E2 where EI.SAL > E2.SAL) will retrieve [WBUT 2011]

a) 3500, 5000, 2500
b) 5000, 2850
c) 2850, 5750
d) 5000, 5750

Answer: (c)

8. What operator performs pattern matching is SQL? [WBUT 2012, 2017]

a) Except b) Intersect c) Like d) All of these

Answer: (d)

9. DML is provided for [WBUT 2013, 2019]

a) description of logical structure of database
b) addition of new structures in the database system
c) manipulation & processing of database
d) definition of physical structure of database system

Answer: (c)

10. Which of the following is a comparison operator in SQL? [WBUT 2013]

a) = b) LIKE c) BETWEEN d) all of these

Answer: (d)

11. Consider the following SQL statements: [WBUT 2014]
A. Select * from student where year = '2nd' or year = '3rd'
B. Select * from student where year in ('2nd', '3rd')

a) A is correct while B is not
b) B is correct while A is not
c) Both will generate same result set
d) A and B will generate different result sets

Answer: (c)

12. Consider the following query: [WBUT 2014]
(((P WHERE COLOUR='RED')'RED')[P#]JOIN SP)[S#]JOIN S)
[SNAME]Which one of the following set of operations does the above query involve?

a) 1 selection, 2 joins, 3 projections
b) 1 projection, 2 joins, 3 selections
c) 2 selections, 2 joins, 3 projections
d) 1 selection, 2 joins, 3 intersections

Answer: (b)

13. What is the name of a trigger that triggers that triggers itself? [WBUT 2014, 2017]

a) Triggering trigger
b) Cascading trigger
c) Mutating trigger
d) None of the above

Answer: (b)

14. DML stands for [WBUT 2015]

a) Data Manipulation Language
b) Data Media Language
c) both (a) & (b)
d) none of these

Answer: (a)

15. Which data type can store unstructured data? [WBUT 2016]

a) Raw
b) Char
c) Numeric
d) Varchar

Answer: (a)

16. The language that requires a user to specify the data to be retrieved without specifying how to get it is [WBUT 2018]

a) procedural DML
b) non procedural DML
c) procedural DDL
d) non-procedural DDL

Answer: (b)

17. The rule that a value of a foreign key must appear as a value of some specific table is called a [WBUT 2018]

a) referential constraint
b) index
c) integrity constraint
d) functional dependency

Answer: (a)

18. The clause in SQL that specifies that the query result should be sorted in ascending or descending order based on the values of one or more column is [WBUT 2018]

a) view
b) order by
c) group by
d) having

Answer: (b)

Short Answer Type Questions

1. Explain the terms View. [WBUT 2006, 2008, 2012]

Answer:

A view is a relation (virtual rather than base) and can be used in query expressions, that is, queries can be written using the view as a relation. In other words, 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). The base relations on which a view is based are sometimes called the existing relations. The definition of a view in a create view statement is stored in the system catalog. The syntax to create a view is:

```
CREATE [OR REPLACE] VIEW <view_name> [(<aliases>)] AS
<query> WITH {READ ONLY|CHECK OPTION [CONSTRAINT
<constraint_name>]}
```

[WBUT 2011]

2. What are the recovery implications of:
a) forcing buffers to the database at COMMIT?
b) fever physically writing buffers to the database prior to COMMIT?

Answer:

a) Provides durability without REDO logging. Never force the buffer from physical writing prior to commit.

b) A transaction ends when any of the following actions occurs:

1. User issue COMMIT/ROLLBACK command *without* [SAVEPOINT] option. Commit, makes the change permanent. It is visible to other users only after commit is executed.
2. When user issues a DDL command like; CREATE, DROP, RENAME, or ALTER. The database issues an implicit COMMIT statement before and after every DDL statement. If the current transaction contains DML statements, then it first commits the transaction and then runs and commits the DDL statement as a new, single-statement transaction.
3. Applications should always explicitly commit or undo transactions before program termination. A client process terminates abnormally, causing the transaction to be implicitly rolled back using metadata stored in the transaction table and the undo segment.

Thus after a transaction commits, users can view the changes. A commit is a fast operation, regardless of the transaction size and the speed of commit does not change with the size of the modified data in the transaction. The lengthiest part of the commit is the physical disk I/O performed by redo buffer e.g. LGWR in oracle. However, the amount of time spent by LGWR is reduced because it has been incrementally writing the contents of the redo log buffer in the background. However, for lower transaction commit latency, application developers can specify that redo be written asynchronously so that transactions need not wait for the redo to be on disk and can return from the COMMIT call immediately and from that perspective it is a fever.

3. What are the differences between Embedded SQL and Dynamic SQL?

[WBUT 2012]

Answer:

Embedded SQL	Dynamic SQL
i) In Embedded (static) mode the bind is done prior to the execution and is stored in a PLAN.	i) In dynamic mode the BIND occurs at execution time.
ii) It is hardcoded in the program itself and the changeable value	ii) It has the capability of changing the columns tables during the execution of program.
iii) It can contain any host variables.	iii) It does not contain any host variables.
iv) The embedded means to coding the SQL query in the general purpose programming language. and programmer must access to a database.	iv) The dynamic SQL means component of SQL allows program to construct and submit SQL queries at run time.
v) Embedded SQL statement must be completely present at the compile time.	v) Dynamic SQL statement may not be completely present at the compile time.

4. Specify the query in SQL to declare a "Cursor" to-find names & cities of residence of customers who have both an account and a loan at a particular bank branch in the same city as that customer. **[WBUT 2012]**

Answer:

select *customer-name, customer-city*
from *account,loan*
where account.*customer-name* = *loan.customer-name*

5. What is a trigger? How many types of trigger are there? **[WBUT 2012]**

Answer:

1st Part: ***Refer to Question No. 1(a) of Long Answer Type Questions.***

2nd Part:

In SQL Server we can create four types of triggers Data Definition Language (DDL) triggers, Data Manipulation Language (DML) triggers, CLR triggers and Logon triggers.

1. DDL Triggers

In SQL Server we can create triggers on DDL statements (like CREATE, ALTER, and DROP) and certain system defined stored procedures that perform DDL-like operations.

Example: If you are going to execute the CREATE LOGIN statement or the sp_addlogin stored procedure to create login user, then both these can execute/fire a DDL trigger that you can create on CREATE_LOGIN event of SQL Server.

We can use only FOR/AFTER clause in DDL triggers not INSTEAD OF clause means we can make only After Trigger on DDL statements.

DDL trigger can be used to observe and control actions performed on the server, and to audit these operations. DDL triggers can be used to manage administrator tasks such as auditing and regulating database operations.

2. DML Triggers

In SQL Server we can create triggers on DML statements (like INSERT, UPDATE, and DELETE) and stored procedures that perform DML-like operations. DML Triggers are of two types

a) **After Trigger (using FOR/AFTER CLAUSE)**
   This type of trigger fires after SQL Server finish the execution of the action successfully that fired it.
   Example: If you insert record/row in a table then the trigger related/associated with the insert event on this table will fire only after the row passes all the constraints, like as primary key constraint, and some rules. If the record/row insertion fails, SQL Server will not fire the After Trigger.

b) **Instead of Trigger (using INSTEAD OF CLAUSE)**
   This type of trigger fires before SQL Server starts the execution of the action that fired it. This is differ from the AFTER trigger, which fires after the action that caused it to fire. We can have an INSTEAD OF insert/update/delete trigger on a table that successfully executed but does not include the actual insert/ update/ delete to the table.

Example: If you insert record/row in a table then the trigger related/associated with the insert event on this table will fire before the row passes all the constraints, such as primary key constraint and some rules. If the record/row insertion fails, SQL Server will fire the Instead of Trigger.

3. CLR Triggers

CLR triggers are special type of triggers that based on the CLR (Common Language Runtime) in .net framework. CLR integration of triggers has been introduced with SQL Server 2008 and allows for triggers to be coded in one of .NET languages like C#, Visual Basic and F#.

We coded the objects (like trigger) in the CLR that have heavy computations or need references to objects outside the SQL Server. We can write code for both DDL and DML triggers, using a supported CLR language like C#, Visual basic and F#. I will discuss CLR trigger later.

4. Logon Triggers

Logon triggers are special type of trigger that fire when LOGON event of SQL Server is raised. This event is raised when a user session is being established with SQL Server that is made after the authentication phase finishes, but before the user session is actually established. Hence, all messages that we define in the trigger such as error messages, will be redirected to the SQL Server error log. Logon triggers do not fire if authentication fails. We can use these triggers to audit and control server sessions, such as to track login activity or limit the number of sessions for a specific login.

6. Discuss the different level of views. **[WBUT 2016]**

Answer:

To reduce the redundant data to the minimum possible, an object is created called View. The view is created mainly because of the following reasons:

1. When data security is required
2. When data redundancy is to be kept to the minimum while maintaining data security

Syntax:

Create View <viewname> as
Select <col 1> [,....] from <tablename>
Where <col Mane> = <expression list>
Group by<group expression> Having <predicate>;

e.g.,

Create View View-student As Select name, elective from student;

Creates a View by the name view-student based on the relation student;

After creation of the view, it can be queried like a base table

Syntax:

Select <col1>, <col2 >,..... from < Viewname>

e.g.,

Select name, elective from View-student;

Long Answer Type Questions

1. a) What is a trigger? **[WBUT 2006, 2007, 2008, 2010, 2012]**

Instead of allowing negative a/c balances, a bank deals with overdrafts by setting the a/c balance to zero, and creating a loan in the amount of the overdraft (An overdraft is an event when a customer's withdrawal amount exceeds the current a/c balance of the customer). The bank gives this loan a loan number identical to the a/c no. of the overdrawn a/c.

Write the trigger in SQL for the above event. **[WBUT 2006, 2007, 2008]**

Answer:

1st part:

SQL trigger is an SQL statements or a set of SQL statements which is stored to be activated or fired when an event associating with a database table occurs. The event can be any event including INSERT, UPDATE and DELETE.

Sometimes a trigger is referred as a special kind of stored procedure in term of procedural code inside its body. The difference between a trigger and a stored procedure is that a trigger is activated or called when an event happens in a database table, a stored procedure must be called explicitly. For example you can have some business logic to do before or after inserting a new record in a database table.

Triggers or stored procedures? It depends on the situation but it is practical that if you have no way to get the work done with stored procedure, think about triggers.

2nd Part:

```
Create trigger overdraft_loan on account
for update
as
if inserted.balance <0
begin
insert into borrower
        (Select customer ,account_no
from depositor, inserted  where
inserted.account_no=depositor.account.no)
insert into loan values
         (inserted .account_no,inserted.branch_name, - inserted.
Balance)
  Update account set balance = 0
           From account, inserted
Where account.account_no=inserted.account_no
End
```

b) Consider the tables:

Employee (Emp-code, Emp-Name, Designation, DOJ, BASIC-SAL, Dept-code and Dept (Dept-code, Dept-Name)

Suppose we want that the clerk in your office should not have access to the details of the salaries, but he needs such information as Emp-Name, Designation and Dept-Name. Create a view for this clerk. **[WBUT 2006, 2008]**

Answer:

```
Create view E-view as
select Emp-name, designation, Dept-name
      from  Employee, Dept
      where    Employee. Dept-code = Dept. Dept-code;
```

2. List two reasons why 'null' values might be introduced into the database? [WBUT 2010]

Answer:

A null value in a relational database is used when the value in a column is unknown or missing. A null is neither an empty string (for character or date time data types) nor a zero value (for numeric data types).

NULL is often used in places where a value is optional or unknown. It can be a convenient way of omitting data without having to resort to strange or arbitrary conventions, such as storing negative values in an integer field to represent omitted data. While your system requirements may change over time, the connotation of NULL is always NULL.

3. a) Consider the following relations and write queries in SQL:
Flights (flno, from, to, distance, departs, arrives, price)
Aircraft (aid, aname, cruising_range)
Certified (eid, aid)
Employees (eid, ename, salary)
i) Identify the flights that can be piloted by every pilot whose salary is more than $1,00,000.
ii) Find the eids of employees who make the second highest salary.
iii) Print the names and salary of every non-pilot whose salary is more than the average salary for pilots.
iv) For all aircraft with cruising_range over 1000 miles, find the name of the aircraft and the average salary of all pilots certified for this aircraft.
v) Find the names of pilots who can operate planes with a range greater than 3000 miles but are not certified on any Boeing aircraft. [WBUT 2010, 2012]

Answer:

i)

```
SELECT E.ename
FROM Aircraft A, Certi_ed C, Employees E, Flights F
WHERE A.aid = C.aid AND E.eid = C.eid AND
distance<cruisingrange AND salary >100,000
```

ii)

```
SELECT E.eid
FROM Employees E
WHERE E.salary = ( SelectMAX (E2.salary)
FROM Employees E2 )
```

iii)

```
SELECT
E.ename, E.salary FROM
Employes E WHERE
E.eid NOT IN
(SELECT DISTINCT
C.eid from Certified C
AND
E.salary> (select avg(E1.salary)
From employees E1
Where E1.eid in
(select distinct C1.eid from certified C1)
```

iv)

```
SELECT
E.ename
FROM
Employes E, Certified C, Aircraft A WHERE
C.aid = A.aid
AND
E.eid = C.eid GROUP BY
E.eid, E.ename
HAVING EVERY (A.cruisingrange>1000)
AND
COUNT(*) > 1
```

v)

```
SELECT E.ename
FROM Certi_ed C, Employees E, Aircraft A
WHERE A.aid = C.aid AND E.eid = C.eid AND A.cruisingrange >3000
AND E.eid NOT IN ( SELECTC2.eid
FROM Certi_ed C2, Aircraft A2
WHERE C2.aid = A2.aid AND A2.aname = `Boeing')
```

b) The saleman_master table records the salesman_no, name, rate_of_commission, qtd_sales. The commission_amount and date_of_payment along with the salesman_no is calculated and recorded in commission_payable table.
Write a PL/SQL block of code such that depending upon the user entered salesman_no, the commission_amount is calculated and inserted into the commission_payable table. If the user enters a saleman_no that is not in the saleman_master table, then the PL/SQL block must display appropriate error message back to the user. [WBUT 2010, 2012]

Answer:

```
DECLARE
 Sman_No Salesman_Master.Salesman_No%TYPE;
 Sales_Amt Salesman_Master.Ytd_Sales%TYPE;
 Comm_Rate Salesman_Master.Commission%TYPE;
```

```
BEGIN
 /* Retrieving records from the Salesman Master table
 and Assigning them to Memory variables for the
 Salesman No entered by the User. */

SELECT Salesman_No, Commission, Ytd_Sales
 INTO Sman_No, Comm_Rate, Sales_Amt
 FROM Salesman_Master
 WHERE Salesman_No = '&Salesman_No';
 /* Calculating the commission amount payable
and inserting into the Commission payable table. */

 INSERT INTO Commission_Payable
 VALUES (Sman_No, sysdate, Sales_Amt * Comm_Rate / 100);
 EXCEPTION
 /* Using the Oracle engine's named Exception
 handler to handle the error condition that may occur
 if the user enters a Salesman No that is not present
 in the Salesman Master Table */
WHEN no_data_found THEN
DBMS_OUTPUT.PUT_LINE('Salesman No ' || Sman_No
|| ' is not present in the Salesman Master table ');
 END;
*/
```

c) Consider the following relational schema:
An employee can work in more than one department; the pct_time field of the works relation shows the percentage of time that a given employee works in a given department:

Emp (eid, ename, age, salary)
Works (eid, did, pct_time)
Dept (did, budget, managerid)

Write a Trigger to ensure each of the following requirements, considered independently.
i) Employees must take a minimum salary $ 1000.
ii) Every manager must also be an employee.
iii) The total percentage of all appointments for an employee must be under 100%.
iv) A manager must always have a higher salary than any employee that he or she manages.

Answer: **[WBUT 2010]**

i)
```
ALTER TABLE Emp ADD CONSTRAINT MinSal
    CHECK (salary >= 1000)
```

ii)
```
ALTER TABLE Dept ADD CONSTRAINT ManIsEmp
    FOREIGN KEY (managerid) REFERENCES Emp (eid)
```

iii)
```
CREATE ASSERTION MangerMakesMore
 CHECK (NOT EXISTS
        (SELECT *
         FROM EmpAS e, Works AS w, DeptAS d
         WHERE e.eid = w.eid AND
               w.did = d.did AND
               e.salary > (SELECT salary
                           FROM EmpAS manager
                          WHERE d.managerid = manager.eid)))
```

4. Hotel (Hotel No, Hotel Name, Address) **[WBUT 2011]**
Room (Room No, Hotel No, Type, Price_pn)
Booking (Hotel No, Guest No, Date From, Date To, Room No)
Guest (Guest No, Guest Name, Guest Address)
Where the underlined column names are primary keys. Write down expressions in relational algebra for the following queries:
i) List all the hotels which are situated in Kolkata
Answer:

$\Pi_{Name}\ \sigma_{Address='Kolkata'}(Hotel)$

ii) List all single rooms with a charge below Rs. 1000 per night **[WBUT 2011]**
Answer:

$\Pi_{RoomNo}\ \sigma_{Type='Single'}\ ^{\wedge}\ _{Price\_Pn<1000}(Room)$

iii) List the name of all guests who are going to stay at ITC hotel from 25th December to 1st January. **[WBUT 2011]**
Answer:

$\Pi_{GuestName}\sigma_{HotelName='ITC'}\ ^{\wedge}\ _{(Datefrom='25th}$December$'\ ^{\wedge}\ _{Dateto='1st\ \ January'}$ ((Guest **EQ.Join** Booking) **EQ.Join** Hotel) [[Guest.GuestNo Θ Booking.GuestNo] Θ Hotelno] (Room)

iv) List the price per night and type of all rooms at Grand Hotel. **[WBUT 2011]**
Answer:

$\Pi_{RoomNo,\ Price\_Pn,\ Type}\ \sigma_{Hotel\ Name='Grand\ Hotel'}$ (Hotel **EQ.Join** Room)
Hotel.Hotel No Θ Room.Hotel No

5. a) Consider the following tables: **[WBUT 2014]**
REATAURANTS (rid, rname, rcity, phone, seat-capacity)
DISHES (did, dname, dtype)
CUSTOMER (cid, ename, ccity)
SERVES (rid, did)
HAS_ALLERGY (cid, did)
i) find the names of the restaurants that serve "Burger" (dish name)
ii) list the names of dishes that customer "Roy" can eat without allergy problem.
iii) list the names of dishes that appear in all restaurants in Kolkata.
iv) Find the names of customers who are in the same city as restaurant "Maachan" and who can eat at least one dish at "Maachan" without allergy problem.

Answer:

i)
```
SELECT resturant.rid, resturant.rname, dishes.dname
FROM resturant INNER JOIN (dishes INNER JOIN serves ON
dishes.did = serves.[did]) ON resturant.rid = serves.[rid]
WHERE dishes.dname="burger";
```

ii)
```
SELECT customer.cid, customer.cname,
serves.[hasalergy(cid,did)]
FROM customer INNER JOIN serves ON customer.cid =
serves.[hasalergy(cid)]
WHERE customer.cname="Roy";
```

iii)
```
SELECT resturant.rid, resturant.rname, resturant.rcity,
    dishes.dname FROM resturant INNER JOIN (dishes INNER
    JOIN serves ON dishes.did = serves.[rid,did]) ON
resturant.rid
    = serves.[rid,did] WHERE resturant.rcity="kolkata";
```

iv)
```
SELECT customer.cid, customer.cname, customer.ccity,
    resturant.rname, serves.[hasalergy(cid,did)], dishes.dname
FROM dishes INNER JOIN ((customer INNER JOIN resturant ON
customer.ccity = resturant.rcity) INNER JOIN serves ON
(customer.cid = serves.[rid,did]) AND (resturant.rid =
serves.[rid,did])) ON dishes.did = serves.[rid,did]
WHERE (((resturant.rname)="Maachan") AND
((serves.[hasalergy(cid,did)])="noll"));
```

b) What are the major problems with performing update operations through view. briefly discuss each. **[WBUT 2014]**

Answer:

VIEW is a virtual table, defined by a query, that does not exist until it is invoked by name in an SQL statement.

A VIEW is a table, so it is named just like any other table.

No constraints on a VIEW can't be put.

The name of the VIEW must be unique within the entire database schema, like a base table name.

The VIEW definition cannot reference itself, since it does not exist yet. Nor can the definition reference only other VIEWs; the nesting of VIEWs must eventually resolve to underlying base tables.

When the columns of a base tables change, the definition of the "star" will also change. However, when the VIEW has too many or too few columns the VIEW will run and give you unexpected results.

Updatable and Read-Only VIEWs

Unlike base tables, VIEWs are either updatable or read-only, but not both. INSERT, UPDATE, and DELETE operations are allowed on updatable VIEWs and base tables,

subject to other constraints. INSERT, UPDATE, and DELETE are not allowed on read-only VIEWs, but can be changed their base tables.

An updatable VIEW is one that can have each of its rows associated with exactly one row in an underlying base table. When the VIEW is changed, the changes pass through the VIEW to that underlying base table unambiguously. Updatable VIEWs in Standard SQL are defined only for queries that meet these criteria.

1. They are built on only one table
2. No GROUP BY clause
3. No HAVING clause
4. No aggregate functions
5. No calculated columns
6. No UNION, INTERSECT or EXCEPT
7. No SELECT DISTINCT clause
8. Any columns excluded from the VIEW must be NULL-able or have a DEFAULT clause in the base table, so that a whole row can be constructed for insertion.

By implication, the VIEW must also contain a key of the table. In short, we are absolutely sure that each row in the VIEW maps back to one and only one row in the base table. The major advantage of this limited definition is that it is based on syntax and not semantics. For example, these VIEWs are logically identical.

c) Specify the following constraints in SQL using accurate applicable command:
i) All values of attribute mark in relation enrolment must be within 0 and 100.
ii) If the value of attribute grade is 'A' then the mark must be more than or equal to 80. If the value of attribute grade is 'B' then the mark must be less than 80 but greater than or equal to 70. **[WBUT 2014]**

Answer:

i)
```
SELECT mark
FROM enrolment
where mark between 0 and 100
```

ii)
```
To display the grade for each student:
Select student_id, (Case when score < 80 then 'B'
                         else 'A'
                          end) as grade from marks
```

6. Consider the following employee database, primary keys are underlined.
Employee(employee-name, street, city)
Works(employee-name, company-name, salary)
Company(company-name, city)
Manages(employee-name, manager-name)

Write SQL's for the queries given below:
(i) Find the names of all employees who work for XYZ.
(ii) Find all employees in the database who live in the same cities aas the companies for which they work.

(iii) Find all employees in the database who live in the same cities and on the same streets as do their managers.
(iv) Find all employees who earn more than the average salary of all employees of their company.
(v) Find the company that has the smallest payroll. **[WBUT 2015]**

Answer:

i)
```
SELECT employee-name
        FROM employee e, works w
         WHERE e. employee-name=w. employee-name and
         company-name='XYZ';
```

ii)
```
SELECT employee-name
        FROM employee, works
WHERE employee.employee-name = works.employee-name AND
works.companyname= company.company-name AND
employee.city = company.city;
```

iii)
```
SELECT e1.employee-name
     FROM employee e1, employee e2, manages
    WHERE e1.employee-name = manages.employee-name AND
    e2. employee-name = manages.manager-name AND
    e1.street = e2.street AND e1.city = e2.city;
```

iv)
```
SELECT employee-name
    FROM works w1,
   (SELECT AVG(salary) AS avg-salary, company-name
     FROM works GROUP BY company-name)w2
   WHERE w1.company-name = w2.company-name AND
   w1.salary>w2.avg-salary;
```

v)
```
Select company-name
  From company
  Where min(salary)= SELECT sum(salary)
  FROM works
  GROUP BY company-name;
```

7. Consider the employee database: **[WBUT 2016, 2017]**
Employee (emp_name, street, emp_id)
Works (emp_name, company_name, salary)
Company (company_name, city)
Manages (emp_name, manager_name)
Write the appropriate SQL statement on the basis of the above table:

a) **Find the names and cities of residence of all employees who work for the UBI.**
b) **Find the names, street addresses and cities of residence of all employees who work for the UBI and earn more than Rs. 50,000.**
c) **Find all employees in the database who do not work for UBI.**
d) **Find the 2nd highest salary for employees in UBI.**
e) **Find the company that has the most employees.**

Answer:

a)
```
select e.employee-name, c.city
from employee e, works w, company c
where e.employee-name = w.employee-name and e.city = c.city
and w.company-name = c.company-name and w. company_name='UBI';
```

b)
```
select e.employee-name, e.street, c.city,
from employee e, works w, company c
where e.employee-name = w.employee-name and e.city = c.city
and w.company-name = c.company-name and w. company_name='UBI' and
w.salary>50,000;
```

c)
```
select employee-name
from works where company-name <> 'UBI'
```

d)
```
select max(salary)from(select salary from works where salary
not in(select max(salary)from works)) where works.company-name <>
'UBI'
```

e)
```
select company-name
    from works
    group by company-name
    having count(*) >= all (select count(*)
    from works group by company-name);
```

8. Answer the following queries in SQL using given database schema: CH4
EMP (E no, E name, E add, B date, super-no)
DEPT (D no, D name, mgrno)
PROJECT (P no, P name, D no, P location)
Work-on (E no, D no, Hours)
a) List E no, E name, E add, mgrno of all employee who work in "Research" department.
b) For all project in "Kolkata" list p-no, controlling department name, manager name, address and birth date.
c) List all project number, project name, manager name which belongs to "Product" department.
d) List all employee name who works more than 40 hours in "Research" department. **[WBUT 2018]**

Answer:

a)
```
SELECT EMP.[E no], EMP.[E name], EMP.[E add], dept.mgrno
   FROM EMP INNER JOIN dept ON EMP.[E no] = dept.mgrno
   WHERE (((dept.[D name])="research"));
```

b)
```
SELECT PROJECT.[P location], dept.[D name], EMP.[E name],
EMP.[E add], EMP.[B date]
FROM EMP INNER JOIN (dept INNER JOIN PROJECT ON dept.[D no] =
PROJECT.[D no]) ON EMP.[E no] = dept.mgrno
```

```
WHERE (((PROJECT.[P location])="kolkata"));
```

c)
```
SELECT PROJECT.[P no], PROJECT.[P name], EMP.[E no]
FROM (PROJECT INNER JOIN dept ON PROJECT.[D no] = dept.[D no])
INNER JOIN EMP ON dept.mgrno = EMP.[E no]
WHERE (((dept.[D name])="product"));
```

d)
```
SELECT EMP.[E name], Workon.Hours
FROM dept INNER JOIN (EMP INNER JOIN Workon ON EMP.[E no] =
Workon.                    [E no]) ON dept.[D no] = Workon.[D no]
WHERE (((Workon.Hours)>40) AND ((dept.[D name])="Research"));
```

FUNCTIONAL DEPENDENCIES AND NORMALIZATION

Multiple Choice Type Questions

1. Lack of normalization can lead to which one of the following problems? [WBUT 2006, 2008]

a) lost updates
b) insertion problems
c) deadlock
d) none of these

Answer: (b)

2. 2NF is always in [WBUT 2010, 2012]

a) 1NF
b) BCNF
c) MVD
d) none of these

Answer: (a)

3. When all the attributes in a relation describe and depend upon the primary key, the relation is said to be in [WBUT 2010]

a) 1NF
b) 2NF
c) 3NF
d) 4NF

Answer: (c)

4. Consider the schema R=(S, T, U, V) and the dependencies $S \to T, T \to U, U \to V$ and $V \to S$. Let $R = (R1 \text{ and } R2)$ be a decomposition such that $R1 \cap R2 = \theta$. The decomposition is: [WBUT 2011]

a) Not in 2NF
b) In 2NF but not in 3NF
c) In 3NF but not in 2NF
d) In both 2NF and 3NF

Answer: (c)

5. $R = \{I, J, K, L\}, F = \{I \to K, IL \to J, JK \to L, L \to K\}$. The candidate keys are [WBUT 2011]

a) J and K
b) JK
c) only I
d) JK and JL

Answer: (d)

6. BCNF is a type of [WBUT 2012]

a) Indexing
b) DFD
c) Normalization
d) None of these

Answer: (c)

7. Second Normal From [WBUT 2014]

a) Eliminates transitive dependency between non-key attributes
b) Eliminates partial dependency between non-key attributes and key attributes
c) Creates separate tables for the set of values that apply to multiply records
d) Creates a separate table for each set of related data and identify a primary key for each such set

Answer: (b)

8. When a row is deleted which one of the following techniques should be used to maintain integrity? [WBUT 2014]

a) The row is deleted and nothing else is done
b) The row is deleted and the references to the deleted primary key, if any, are replaced by NULL
c) The delete operation is not allowed if the row's primary key is a target of a foreign key
d) The row is deleted as well as the rows from other tables that have foreign keys that have the deleted primary key as their target

Answer: (c)

9. Given the following relation instance [WBUT 2014]

X	Y	Z
1	4	2
1	5	3
1	6	3
3	2	2

Which of the following functional dependencies are satisfied by the instance?

a) $XY \rightarrow Z$ and $Z \rightarrow Y$
b) $YZ \rightarrow X$ and $Y \rightarrow Z$
c) $YZ \rightarrow X$ and $X \rightarrow Z$
d) $XZ \rightarrow Y$ and $Y \rightarrow X$

Answer: (c)

10. If a table R consists only of its primary key (which may consists of a number of attributes) and has no other attributes, the table would always be in [WBUT 2014]

a) 2NF but may not be in 3NF
b) 3NF but may not be in BCNF
c) 4NF
d) None of these

Answer: (c)

11. Closure of F is [WBUT 2015]

a) F b) F+ c) F– d) F++

Answer: (b)

12. Relations produced from an ER-model will always be in [WBUT 2015]

a) 1NF b) 2NF c) 3NF d) 4NF

Answer: (a)

13. BCNF stands for [WBUT 2015]

a) Boyle Codd Normalization
b) Boyce Cold Normal Form
c) Boyce Codd Normal Form
d) none of these

Answer: (c)

14. Which of the following is true? [WBUT 2015]

a) a super key is always a candidate key
b) every 3NF schema is also in BCNF
c) generalization is a bottom-up design approach
d) none of these

Answer: (c)

15. 2 NF is based on [WBUT 2016]
a) full dependency b) transitive dependency
c) functional dependency d) partial dependency
Answer: (c)

16. A normal form in which every non-prime attribute is fully dependent on prime attribute is [WBUT 2016]
a) 1 NF b) 2 NF c) 3 NF d) BCNF
Answer: (b)

17. What is the highest normal form for the relational schema Bank? [WBUT 2017]
a) First b) Second c) Third d) Boyce code
Answer: (d)

18. Transitive dependency is removed in [WBUT 2019]
a) 1 NF b) 2NF c) 3NF d) 4NF
Answer: (c)

Short Answer Type Questions

1. Find out closure of attribute set (AG) i.e. $(AG)^+$ in the relational schema R and set of functional dependencies F as given blow: [WBUT 2007, 2012]
R = (A, B, C, G, .H, I)
F= {A—>B, A—> C, CG—>H ,CG—>I, B—>H}
Answer:
Let AG= {AB} the functional dependencies A—>B Given in F compute the closure AG^+ under F.
Initialise $AG^+_{<current>}=\{AB\}$
From the relation $A \rightarrow C$ in F s.t $C \subseteq AG^+_{<current>}$
$AG^+_{<new>}=\{ABC\}$
From the relation $B \rightarrow H$ in F s.t $H \subseteq AG^+_{<current>}$
$AG^+_{<new>}=\{ABCH\}$
Thus AG^+ is $\{ABCH\}$

2. Define BCNF. How does it differ from 3NF? Why is it considered a stronger from 3 NF? [WBUT 2007, 2009, 2010, 2011, 2012, 2019]
OR,
Explain with example "BCNF is stricter than 3NF". [WBUT 2015]
OR,
How does BCNF differ from 3rd normal form? [WBUT 2016]
Answer:
Boyce-Codd normal form (or **BCNF** or **3.5NF**) is a normal form used in database normalization. It is a slightly stronger version of the third normal form (3NF). A table is in Boyce-Codd normal form if and only if for every one of its nontrivial dependencies $X \rightarrow Y$, X is a super key—that is, X is either a candidate key or a superset thereof.

In the definition of BCNF each FD X →Y, X must always be a superkey. For 3NF, however, X must be a superkey only if Y is not part of any other relation key (i.e. if Y is non-prime).

Only in rare cases does a 3NF table not meet the requirements of BCNF. A 3NF table which does not have multiple overlapping candidate keys is guaranteed to be in BCNF. Depending on what its functional dependencies are, a 3NF table with two or more overlapping candidate keys may or may not be in BCNF.

BCNF is stronger than 3nf.

A relation R is in 3NF if and only if every dependency A->B satisfied by R meets at least ONE of the following criteria:

1. A->B is trivial (i.e. B is a subset of A)
2. A is a superkey
3. B is a subset of a candidate key

BCNF doesn't permit the third of these options. Therefore BCNF is said to be stronger than 3NF because 3NF permits some dependencies which BCNF does not.

3. Compute the closure of the following set F of functional dependencies for relation schema: **[WBUT 2008, 2011]**

R= (A, B, C, D, E)
A→BC, CD→E, B→D, E→A
List the candidate keys for R.

Answer:

From the given FD **A →BC**

Let us assume that F= {ABC} and we compute the closure F^+

From the relation **B→D** s.t D ⊆ $F^+_{<current>}$

Initialise $F^+_{<current>}$ ={ABCD }

From the relation CD→E s.t E ⊆ $F^+_{<current>}$

$F^+_{<new>}$={ABCDE}

Thus F^+ is {ABCDE } .

Dependencies Are:

A→B, A→C, A→D, A→E: A is a candidate key

CD→E is a composite key and a candidate key

4. Draw a functional dependency diagram (FD diagram) that is in 3NF but not in BCNF. Decompose that FD diagram into BCNF. **[WBUT 2009, 2011]**

Answer:

Example (3NF but not BCNF):

SUPPLIER_PART (supplier_no, supplier_name, part_no, quantity)

Functional Dependencies:

We assume that supplier_name's are always unique to each supplier. Thus we have two candidate keys:

(supplier_no, part_no) and (supplier_name, part_no)

Thus we have the following dependencies:

(supplier_no, part_no) → quantity

(supplier_no, part_no) → supplier_name
(supplier_name, part_no) → quantity
(supplier_name, part_no) → supplier_no
supplier_name → supplier_no
supplier_no → supplier_name
Although supplier_name → supplier_no (and vice versa), supplier_no is not a non-key column and is part of the primary key. Hence this relation technically satisfies the definition(s) of 3NF.

Decomposition (into BCNF):
SUPPLIER_ID (supplier_no, supplier_name)
SUPPLIER_PARTS (supplier_no, part_no, quantity)

5. What is lossless decomposition? **[WBUT 2011, 2013]**

OR,

Give an example of table decomposition which is not lossless. **[WBUT 2014]**

Answer:

Lossless decomposition:

Any relation when decomposed in to two or more relations the primary objective is to ensure that it should be Loss less. In order to do that the following functional dependencies are checked. Let us assume that the parent relation R be decomposed into R1 and R2. F be a set of functional dependencies in R. F^+ is the closure of R holding any of the following functional dependencies:

R1 ∩ R2 →R1
R1 ∩ R2 →R2

The relational expressions above ensures that the attributes participates in the natural join (R1 ∩ R2) is a candidate key for one of the two relations and the chance of generating unwanted tuples is also ruled out. For example;

ContractNo	HotelNo	Location
C1024	H25	Delhi
C1024	H25	Delhi
C1025	H04	Mumbai
C1025	H04	Mumbai

Let R={contractno, hotelno, location}
Let R1={ **Cntractno**, Hotelno }, R2= {**HotelNo**, location}, where Cntractno and Hotelno are the primary keys of the decomposed relations R1 and R2 respectively.
The closure F^+ is {R} above.
Let us apply the dependency rule above
R1 ∩ R2= {contractno, hotelno} ∩ {HotelNo, location}→R2→ Hotelno
Is the primary key of R2. Finally the loss less decomposed relations are:

ContractNoHotelNoC1024
H25C1025H04R1

HotelNoLocationH25Delhi
H04MumbaiR2

6. Consider relation R (A, B, C) and a set of function dependencies F = {A→BC, B→C, A→B, AB→C} Compute the canonical cover for F. [WBUT 2011]

Answer:

A canonical cover F_c for F is a set of dependencies such that F Logically implies all dependencies in F_c and F_c logically implies all dependencies in F, moreover

– No functional dependency in F_c contains an extraneous attribute.

– Each left side of a functional dependency in F_c is unique.

R = (A, B, C)

F = A→ BC, B →C, A→ B, AB → C

Let us combine A→ BC and A → B into A → BC

A is extraneous in AB →C because B→ C logically implies AB → C.

C is extraneous in A→BC, since A→BC is logically implied by A→B and B→C.

The canonical cover is:

A→B and B→C

7. Compare between 3NF and BCNF with example. [WBUT 2012]

Answer:

3NF	BCNF
i) In case of 3NF all the determinant cannot be candidate key.	i) In case of BCNF all the determinant must be candidate key.
ii) 3NF is less stricter than BCNF.	ii) BCNF is more stricter than 3NF.
iii) 3NF is normal form in which the table is in 2NF and every non-prime attribute is non-transitively dependent on every key in the table.	iii) BCNF is a normal form in which for every one of a table's non-trivial functional dependencies, is a super key.

8. What do you mean by closure? [WBUT 2013]

Answer:

The closure of a set F of functional dependencies is the set of all functional dependencies logically implied by F. We denote the closure of F by $F+$.

9. Suppose that we decompose the schema, $R=(A,B,C,D)$ into (A,B,C) and (A,D,E). [WBUT 2013]

Show that this decomposition is lossless decomposition, if the following set of F of FDs holds –

$A\rightarrow BC, CD\rightarrow E, B\rightarrow D, E\rightarrow A$.

Answer:

	A	B	C	D	E
R_1	a_1	a_2	a_3	b_{14}	b_{15}
R_2	a_4	b_{22}	B_{23}	a_5	a_6

Using the relation A→BC; implies A→B, A→C

The new matrix will be

	A	B	C	D	E
R_1	a_1	a_2	a_3	b_{14}	b_{15}
R_2	a_4	b_{22}	b_{23}	a_5	a_6
		a_7	a_8		

Thus R2 is fully transformed in terms of a_{ij} and hence the decomposition is loss less.

10. Discuss the properties of decomposition including attribute preservation, dependency preservation and lossless join with example. [WBUT 2014]

Answer:

Attribute preservation condition:

Each attribute in R will appear in at least one relation schema Ri in the decomposition so that no attributes are "lost".

Another goal of decomposition is to have each individual relation Ri in the decomposition D be in BCNF or 3NF.

Additional properties of decomposition are needed to prevent from generating spurious tuples.

Lossless Decomposition

Any relation when decomposed in to two or more relations the primary objective is to ensure that it should be Loss less. In order to do that the following functional dependencies are checked. Let us assume that the parent relation R be decomposed into R1 and R2. F be a set of functional dependencies in R. F^+ is the closure of R holding any of the following functional dependencies: ∩

R1 ∩ R2 →R1

R1 ∩ R2 →R2

The relational expressions ensures that the attributes participates in the natural join (R1 ∩ R2) is a candidate key for one of the two relations and the chance of generating unwanted tuples is also ruled out. For example;

ContractNo	HotelNo	Location
C1024	H25	Delhi
C1024	H25	Delhi
C1025	H04	Mumbai
C1025	H04	Mumbai

Let R={contractno, hotelno, location}

Let R1={**Contractno**, Hotelno}, R2= {**HotelNo**, location}, where ContractNo and Hotelno are the primary keys of the decomposed relations R1 and R2 respectively.

The closure F^+ is {R} above.

Let us apply the dependency rule above

R1 ∩ R2= {contractno, hotelno} ∩ {HotelNo, location}→R2→ Hotelno

Is the primary key of R2. Finally the loss less decomposed relations are:

ContractNo	HotelNo
C1024	H25
C1025	H04

R1

HotelNo	Location
H25	Delhi
H04	Mumbai

R2

Dependency Preservation

A decomposition having the property that $F'^{+} = F^{+}$ is a dependency preserving decomposition. Let us now work out with the algo:

```
Compute F+
For each schema Ri in D do
Begin
Fi' = only the fd's of of F+ in Ri;
End
F' = ϕ
For each  restricted Fi do
Begin
F'= F' ∪ Fi ;
End
Compute F'+;
If (F'+ = F+) then
return (T)
Else
return (F);
```

11. Consider the relation $R=\{A,B,C,D,E,F,G,H,I,J\}$ and the set of functional dependencies:

$$F=\{AB\rightarrow C, A\rightarrow DE, B\rightarrow F, F\rightarrow GH, D\rightarrow IJ\}$$

Decompose R into 3NF.

Answer: **[WBUT 2016]**

i) The core of all the keys is AB (i.e., AB are the attributes not on the right hand side of any given functional dependency).

Since {AB}+ = ABCDEFGHI, then AB is the only key for R.

ii) First, transform the FDs into standard form

$AB \rightarrow C$, $A \rightarrow D$, $A \rightarrow E$, $B \rightarrow F$, $F \rightarrow G$, $F \rightarrow H$, $D \rightarrow I$

Second, decompose R into BCNF (lossless-join):

ABCDEFGHI

$| A \rightarrow D$

ABCEFGHI and AD

$| A \rightarrow E$

ABCFGHI and AE $| B \rightarrow F$

ABCGHI and BF

So the BCNF decomposition is: ABCGHI, BF, AE, AD

And BCNF is 3NF. So it's done.
(Note: If you apply a different set of FDs for the BCNF decomposition, you may get a different result.)
[Remark: In case the question also requires the decomposition to be dependency-preserving, then:
The FDs in standard form above are already minimal cover.
Since FDs $F \rightarrow G$, $F \rightarrow H$, $D \rightarrow I$ are not preserved by the above BCNF decomposition, we add FG, FH and DI into the final 3NF decomposition:
ABCGHI, BF, AE, AD, FG, FH and DI.]

12. What are the advantages of normalization? **[WBUT 2016]**

Answer:

Advantages of normalization

1. Smaller database: By eliminating duplicate data, you will be able to reduce the overall size of the database.
2. Better performance:

a. Narrow tables: Having more fine-tuned tables allows your tables to have less columns and allows you to fit more records per data page.
b. Fewer indexes per table mean faster maintenance tasks such as index rebuilds.
c. Only join tables that you need.

13. What is closure and minimal cover? What is inclusion dependency? **[WBUT 2017]**

Answer:

1st part:

Closure: Refer to Question No. 8 of Short Answer Type Questions.

Minimal Cover:

$R(A\ B\ C\ D),\ F = \{A \rightarrow BC,\ B \rightarrow C,\ AB \rightarrow D\}$

$A \rightarrow BC$

$A \rightarrow B,\ A \rightarrow C,\ B \rightarrow C,\ AB \rightarrow D$ ✓

✓ ✗ ✓

$A^+ = ABCD \qquad B^+ = BC$

$A^+ = A\ C\ D$
$A^+ = ABCD$
$B^+ = B$
$A^+ = ABC$

$A \rightarrow B,\ B \rightarrow C$
$A \rightarrow D$

Thus minimal cover can be summarily defined as:

a) RHS should not contain more than one attribute
b) If LHS contains more than one attribute remove one extra attribute
c) Remove extra FD's one by one

2nd part:
Inclusion dependencies represent one-to-many relationships. The right hand side (RHS) of a relation indicates any candidate key of the specified relation variable, not necessarily its primary key. The left hand side (LHS) and RHS of a dependency relationship are not required to be a foreign key and a candidate key, respectively. This is merely required to write a correct inclusion dependency expression of a referential integrity relationship.

e.g.

parts

PK				
part_num	part_name	color	weight	city

supplier_parts

PK		
FK	FK	
supp_num	part_num	quantity

Using the notation R.A, where R is the name of a relation variable and A is the name of one of its attributes, you can write the following inclusion dependency:

supplier_parts.part_num → parts.part_num

This inclusion dependency states that the set of values appearing in the attribute part_num of relation variable supplier_parts must be a subset of the values appearing in the attribute part_num of relation variable parts.

14. What is partial functional dependency? Explain BCNF with a suitable example. **[WBUT 2018]**

Answer:
1st part: ***Refer to Question No. 10(b) of Long Answer Type Questions.***

2nd part: ***Refer to Question No. 13(c) of Long Answer Type Questions.***

15. Explain Armstrong axiom in brief. **[WBUT 2018]**

Answer:
Refer to Question No. 24(a) of Long Answer Type Questions.

Long Answer Type Questions

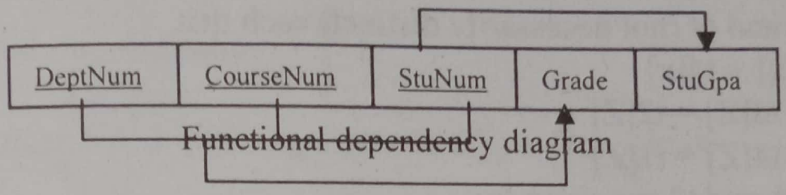
1. Discuss the *"insertion anomalies"*, *"updation anomalies"* and *"deletion anomalies"* with respect to normal forms with suitable example and suggest a method to overcome them. **[WBUT 2006, 2008]**

Answer:
An anomaly is a variation with respect to a database maintenance. In databases when **update, insert, delete** operations are performed, it is desirable that these operations should be in a minimum time, cost effective, predictable and efficient.
However, there are some aspect of the relation that may be difficult to maintain. In order to change the value of one attribute of an entity, ideally that change require only one tuple to be updated. When the relations are not fully normalized they exhibit update anomalies.

Let us consider the relation structure and sample records:

DeptNum	CourseNum	StuNum	Grade	StuGpa
CSE	CS101	742101	A	8.04
CSE	CS101	662771	E	8.75
CSE	EC222	662771	E	8.75
CSE	EE101	742101	A	8.04

Relation: STUDENT



Functional dependency diagram

The relation STUDENT keeps the information about the student enrolled in the courses, the grade assigned for the course, and the student's overall grade point average. The functional dependency diagram above shows relations among the attributes:

Update Anomaly: What if the student's stuGpa changes? We then need to change **stuGpa** in several records. We call this type of difficulty as an **update anomaly** – the simple change of a student's stuGpa affects, not just one record, but potentially several records in the database. The update operation is more complex than necessary, and this means it is more expensive to do, resulting in slower performance.

Delete Anomaly: Let us assume that the above relation STUDENT is the only information about stuGpa of a student and for stuNum equals to **'662771'**, the stuGpa is required to be deleted. What happens to the student's stuGpa? The delete operation has to be executed two times and both the course information EC222 and CS101 will be deleted from the relation. We refer the case as **deletion anomaly**.

Insertion Anomaly: Suppose we need to add a new student and assume a new student's GPA is 0. How do we add this information? Before we could add a row to this relation, we need to enrol the student into a particular course. i.e. course number is a must. This type of inefficient database design is referred to as Insertion Anomaly.

e.g.,

Name	Course	Phone#	Major	Prof	Grade
X1	353	2556-1818	CSE	P1	O
X 2	329	2592-0584	IT	P2	B
X1	328	2556-1818	CSE	P3	B
X3	456	2563-3193	ECE	P4	E
X1	492	2556-1818	CSE	P8	D
X5	379	2357-5684	Eng.	P9	E

Relation: STUDENT

Normalization is a process by which such anomalies can be removed.

2. a) Define MVD with suitable example. **[WBUT 2006, 2007]**

Answer:

Formally, the multivalued dependency **X →→ Y** is said to hold for a relation R(X, Y, Z) if for a given set of value (set of values if X is more than one attribute) for attributes X, there is a set of (zero or more) associated values for the set of attributes Y and the Y values depend only on X values and have no dependence on the set of attributes Z.

Now, more formally, X→→Y is said to hold for R(X, Y, Z) if t1 and t2 are two tuples in R that have the same values for attributes *X* and therefore with *t1[x] = t2[x]* then *R* also contains tuples *t3* and *t4* (not necessarily distinct) such that

t1[x] = t2[x] = t3[x] = t4[x]

t3[Y] = t1[Y] and t3[Z] = t2[Z]

t4[Y] = t2[Y] and t4[Z] = t1[Z]

In other words if t1 and t2 are given by

t1 = [X, Y1, Z1], and

t2 = [X, Y2, Z2]

then there must be tuples t3 and t4 such that

t3 = [X, Y1, Z2], and

t4 = [X, Y2, Z1]

Fourth normal form

e.g.,

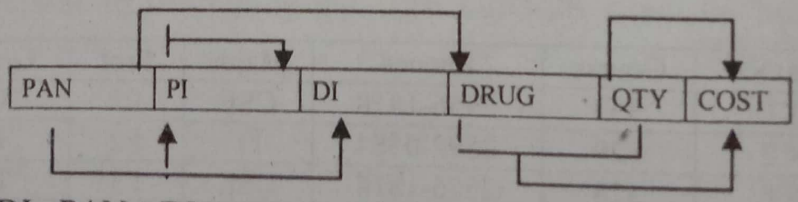
List the non-trivial multi-valued dependencies

A	B	C
a1	b1	c1
a1	b1	c2
a2	b1	c1
a2	b1	c1

The FDs and MVDs are: **A→B, A→→C, B→→C**

b) Find a loss-less joint decomposition of
R = {PAN, PI, DI, DRUG, QTY, COST} into Boyce-Cod normal form with set of functional dependencies.
F={PAN → PI, PI → DI, PI, DRUG → QTY, QTY → COST} **[WBUT 2006, 2007]**

Answer:



PAN->PI , PI -> DI , PAN->DI----------(1)

PI, DRUG-> QTY-----(2)

QTY->COST ------(3)

Decomposition is loss less.

3. a) Explain the following terms:
i) Normalization **[WBUT 2007]**

Answer:

Normalization

Relations resulting from a well-written E-R design may still have some undesirable properties, e.g. composite attributes, multi-valued attributes, data redundancy etc. In order to remove these undesirables we apply a process called **normalization**. The central concept is the notion of *functional dependency*.

ii) Full functional dependency **[WBUT 2007]**

Answer:

Refer to Question No. 5 of Long Answer Type Questions.

iii) Transitive dependency **[WBUT 2007]**

Answer:

A transitive dependency is an indirect functional dependency, one in which $X \rightarrow Z$ only by virtue of $X \rightarrow Y$ and $Y \rightarrow Z$.

iv) Partial dependency **[WBUT 2007]**

Answer:

A partial dependency is a dependency where A is functionally dependant on B (A → B), but there is some attribute on A that can be removed from A and yet the dependacy stills holds. For instance if the relation existed

StaffNo, sName → branchNo

Then you could say that for every StaffNo, sName there is only one value of branchNo, but since there is no relation between branchNo and staffNo the relation is only partial.

b) Explain different types of Normal Form with examples (upto 5 NF). [WBUT 2007]

Answer:

1NF

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

Example: 1NF but not 2NF

FIRST (supplier_no, status, city, part_no, quantity)

Functional Dependencies:

(supplier_no, part_no) → quantity

(supplier_no) → status

(supplier_no) → city

city → status (Supplier's status is determined by location)

Non-key attributes are not mutually independent (city ® status).

Non-key attributes are not fully functionally dependent on the primary key (i.e., status and city are dependent on just part of the key, namely supplier_no).

2NF

A relation R is in second normal form (2NF) if and only if it is in 1NF and every non-key attribute is fully dependent on the primary key

SECOND (supplier_no, status, city)

Functional Dependencies:

supplier_no → status

supplier_no → city

city → status

3NF

A relation R is in third normal form (3NF) if and only if it is in 2NF and every non-key attribute is non-transitively dependent on the primary key. An attribute C is transitively dependent on attribute A if there exists an attribute B such that: A → B and B → C. Note that 3NF is concerned with transitive dependencies which do not involve candidate keys. A 3NF relation with more than one candidate key will clearly have transitive dependencies of the form: primary_key → other_candidate_key → any_non-key_column

SUPPLIER_PART (supplier_no, supplier_name, part_no, quantity)

Functional Dependencies:

We assume that supplier_name's are always unique to each supplier. Thus we have two candidate keys:

(supplier_no, part_no) and (supplier_name, part_no)

Thus we have the following dependencies:

(supplier_no, part_no) → quantity

(supplier_no, part_no) → supplier_name

(supplier_name, part_no) → quantity

(supplier_name, part_no) → supplier_no

supplier_name → supplier_no

supplier_no → supplier_name

BCNF

Refer to Question No. 2(1st part) of Short Answer Type Questions.

4NF

A multivalued dependency is best illustrated using an example. In a table containing a list of three things - college courses, the lecturer in charge of each course and the recommended book for each course - these three elements (course, lecturer and book) are independent of one another. Changing the course's recommended book, for instance, has no effect on the course itself. This is an example of multivalued dependency: An item depends on more than one value. In this example, the course depends on both lecturer and book.

Thus, 4NF states that a table should not have more than one of these dependencies. 4NF is rarely used outside of academic circles.

5NF

The 5th normal form is also known as project-join normal form. A relation is in fifth normal form (5NF), if it is in 4NF (fourth Normal Form) and won't have lossless decomposition in smaller tables.

c) Given R(A,B,C,D) with FD's F={A->B,A->C,C->D} Consider the decomposition of R into R1 (A, B, C) with FD F1={A-> B, A-> C } and
R2 (C, D) with FD F2= {C—> D}.
Is this decomposition lossless and dependency preserving? Explain. [WBUT 2007]

Answer:

Lossless Decomposition:

Let R having the decompositions R1 and R2 with given set of FD's say F , then R1 and R2 to become lossless then at least one of the following functional dependencies should hold in F^+

$R1 \cap R2 \longrightarrow R1$

$R1 \cap R2 \longrightarrow R2$

From the given FDs $C \longrightarrow D$,

$C \rightarrow CD \longrightarrow R2$ [decomposition is loss less]

Dependency Preservation

```
Result = (Result ∩ R1)+ ∩R2
F= {A->B,A->C,C->D}
F+'= {((A->B,A->C,C->D,A->D) ∩ ( A-> B, A-> C) ) ∩ ( C—> D)  }
      = {A->B,A->C,C->D }
Thus F= F+'[Hence dependency Preserving]
```

4. If 'D' be the set of all functional and multivalued dependencies then write the rules to compute the'D+ (Closureof D). [WBUT 2007, 2010]

Answer:

We can find all of D+ by applying Armstrong's Axioms:

- if $\beta \subseteq \alpha$, then $\alpha \rightarrow \beta$ **(reflexivity)**
- if $\alpha \rightarrow \beta$, then $\gamma\ \alpha \rightarrow \gamma\ \beta$ **(augmentation)**
- if $\alpha \rightarrow \beta$, and $\beta \rightarrow \gamma$, then $\alpha \rightarrow \gamma$ **(transitivity)**

5. Explain the following terms 'Fully functional dependency' and 'non transitive dependency' with examples. [WBUT 2008]

Answer:

An attribute is **fully functionally** dependent on a set of attributes X if it is

i. functionally dependent on X, and
ii. not functionally dependent on any proper subset of X. {Employee Address} has a functional dependency on {Employee ID, Skill}, but not a *full* functional dependency, because it is also dependent on {Employee ID}.

A **non-transitive dependency** is one in which an attribute is dependent on a non-key attribute in that table. There is no dependency between non-prime key attributes.

Consider a relation student having prime key attribute (Roll) and non-prime key attributes(Name, semester, hostel) where for each semester there is a different hostel. Here Name is non-transitively dependent on Roll.

6. What is MVDs? **[WBUT 2008, 2011]**
What do you mean by lossless (or non-additive) join property of decomposition? **[WBUT 2008]**

Answer:

1st Part: ***Refer to Question No. 2(a) of Long Answer Type Questions.***

2nd part:

Lossless Decomposition:

Let R having the decompositions R1 and R2 with given set of FD's say F , then R1 and R2 to become lossless then at least one of the following functional dependencies should hold in F^+

$R1 \cap R2 \longrightarrow R1$

$R1 \cap R2 \longrightarrow R2$

7. Use the definition of functional dependency to argue that each of Armstrong's axioms (reflexivity, augmentation, transitivity, union and decomposition) is sound. **[WBUT 2008, 2010, 2012]**

Answer:

A set of inference rule is said to be sound if we cannot obtain any dependency that is not in F^+

Let us consider a relation, $R=\{A_1, A_2, A_3, \ldots\ldots, A_n\}$ and let r be a instance of R

Reflexivity

Let $Y \subseteq X \subseteq R$. Then for any two tuples t_1 and t_2 or r

$t_1[x] = t_2[x]$ and we can also say $t_1[Y] = t_2[Y]$ hence $X \rightarrow Y$

Augmentation

Let us suppose that *r* satisfies $X \rightarrow Y$ and let $Z \subseteq W \subseteq R$.

Then for any two tuples t_1 and t_2 or *r* such that $t_1[xW] = t_2[xW]$

Since *r* satisfies $X \rightarrow Y$ and X values of t_1 and t_2 are equal $t_1[Y] = t_2[Y]$.

Also W values of t_1 and t_2 being $t_1[Z] = t_2[Z]$

Therefore, $XW \rightarrow YZ$ holds in r.

Union

To prove that: if $\alpha \rightarrow \beta$ and $\alpha \rightarrow \gamma$ then $\alpha \rightarrow \beta\gamma$

We derive:

$\alpha \rightarrow \beta$ given

$\alpha\alpha \rightarrow \alpha\beta$ augmentation rule

$\alpha \rightarrow \alpha\beta$ union of identical sets

$\alpha \rightarrow \gamma$ given

$\alpha\beta \rightarrow \gamma\beta$ augmentation rule

$\alpha \rightarrow \beta\gamma$ transitivity rule and set union commutativity

Transitivity

Suppose that r satisfies X→Y and Y→Z, then for any two tuples t_1 and t_2 of *r*, if $t_1[x] = t_2[x]$ by X→Y, $t_1[Y] = t_2[Y]$

Again Y→Z, requiers $t_1[Z] = t_2[Z]$. Hence r satisfies X→Z

Reflexivity rule leads to trivial dependencies. Trivial dependencies does not depend on the given set of dependencies.

The use of augmentation and the transitivity depends on the given set of dependencies and these rules can be repeatedly used to infer new non-trivial dependencies.

Decomposition Rule

If $\alpha \rightarrow \beta\gamma$ then $\alpha \rightarrow \beta$ and $\alpha \rightarrow \gamma$

Let $\alpha \rightarrow \beta\gamma$ --------- (A)

Now, $\gamma \subseteq \beta\gamma$ $\therefore \beta\gamma \rightarrow \gamma$ --------- (B)

(From Reflexivity Rule)

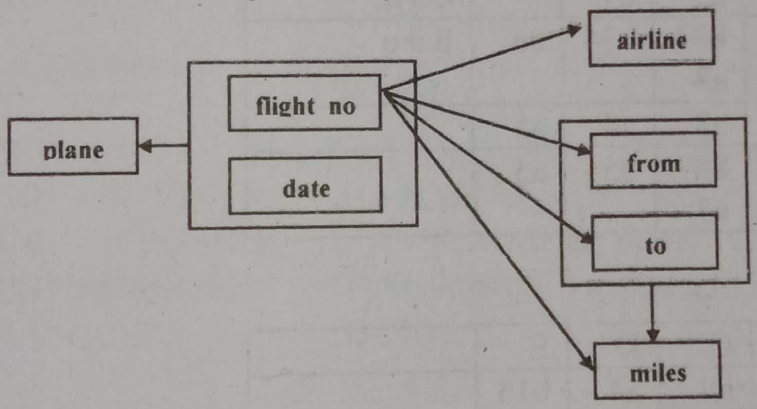
from (A) and (B) through Transitivity Rule

$\alpha \rightarrow \gamma$

In manner in can be shown $\alpha \rightarrow \beta$

Since, $\beta \subseteq \beta\gamma$.

8. Given a database schema named PLANE_INFO (flight_no, date, plane, airline, from, to, miles), the functional dependency diagram is given below:



Decompose it up to Boyce-Codd Normal Form (BCNF). **[WBUT 2009, 2010]**

Answer:

Flight_no,date→Plane (1)

Flight_No→airline,from,to,miles (2)

From,to→miles (3)

Relation (2) holds a transitive dependencies

So the revised FD'd may be represented as:

Flight_no,date→Plane (1)

Flight_No→airline,from,to (4)

From,to→miles (3)

From (1), (3) and (4) it is apparent that all the determinants are candidate keys. Hence the FD's are in BCNF.

9. Consider the relation R (A, B, C, D, E) with the set of F = {A → C, B → C, C → D, DC → C, CE → A}. Suppose the relation has been decomposed by the relations R1 (A, D) R2 (A, B) R3 (B, E) R4 (C, D, E), R5 (A, E). Is this decomposition lossy or lossless? Justify your answer. **[WBUT 2009, 2010, 2012]**

Answer:

	A	B	C	D	E
R1	a1	b12	b13	a4	b15
R2	a1	a2	b23	b24	b25
R3	b31	a2	b33	b34	a5
R4	b41	b42	a3	a4	a5
R5	a1	b52	b53	b54	a5

In order to study the lossy or loss less decomposition, we tabulate the decompositions into a (5 × 5) matrix. Against each available decomposed component we fill with 'a' otherwise with 'b'. Now we proceed to show any of the rows from R1 to R5 should be converted with all 'a' to conclude that the decompositions are lossless.

Step 1

	A	B	C	D	E	
R1	a1	b12	~~b13~~ a3	a4	b15	a→c
R2	a1	a2	~~b23~~ a3	~~b24~~ ~~a4~~	b25	a→c c→d
R3	b31	a2	~~b33~~ ~~a3~~	b34	a5	b→c
R4	b41	b42	a3	a4	a5	
R5	a1	b52	~~b53~~ a3	b54	a5	

Step 2

	A	B	C	D	E	
R1	a1	b12	a3	a4	b15	
R2	a1	a2	a3	~~a4~~	b25	
R3	b31	a2	a3	b34	a5	
R4	~~b41~~ a1	b42	a3	a4	a5	ce→a
R5	a1	b52	a3	b54	a5	

The decomposition is lossy as none of the row operations leads to all 'a's.

10. a) For a given Relvar $R=\{A,B,C,D,E,F\}$ and set of functional dependencies $F=\{AB\rightarrow C, C\rightarrow A, BC\rightarrow D, ACD\rightarrow B, BE\rightarrow C, CE\rightarrow FA, CF\rightarrow BD, D\rightarrow EF\}$, find the irreducible set & candidate keys.
b) Explain the following terms 'partial functional dependency' and 'non-transitive dependency' with examples.

c) Consider the following proposed rule for functional dependencies: If $A \rightarrow B$ and $C \rightarrow B$, then $A \rightarrow C$. Prove that this rule is not sound by showing a relation r that satisfies $A \rightarrow B$ and $C \rightarrow B$, but does not satisfies $A \rightarrow C$.

[WBUT 2010]

Answer:

a) Steps to find irreducible set

1. Using decomposition rule to rewrite the FDs such that each has a singleton right-hand side:

R=

AB→C D →E CF→D
C →A D →F CE →A
BC→D BE →C CE →F
ACD →B CF→B

2. For each FD f in R, if deleting f from R has no effect on the closure R^+, we delete f from R.

AB→C {AB}$^+$={AB} C ∉{a,b}
C→A {C}$^+$={C} A ∉{C}
BC→D {BC}$^+$={BCA} D ∉{BCA}
ACD→B {ACD}$^+$={ABCDEG} B ∈{ABCDEG}
delete ACD→B
C→A {C}$^+$={C} A ∉{C}
D→E {D}$^+$={DG} E ∉{DG}
D→G {D}$^+$={DE} G∉{DE}
BE→C {BE}$^+$={BE} C∉{BE}
CG→B {CG}$^+$={CGDA} B∉{CGDA}
CE→A {CE}$^+$={CEGABD} A ∈{CEGABD} delete CE→A
CG→D {CG}$^+$={CGBADE} D∈{CGBADE} delete CG→D
CE→G {CE}$^+$={CEA} G∉{CEA}

R=

AB→C D →G
C →A BE →C
BC→D CG →B
D →E CE →G

3. For each FD f in S, we examine each attribute A in the left-hand side of f; if deleting A from the left-hand side of f has no effect on the closure R^+, we delete A from the left-hand side of f.

R1 is a irreducible sets of dependencies.

If we compute step three before the step two, then:

ACD→B {CD}$^+$={CDAEGB} B ∈{CDAEGB}
ACD→B CD→B

R2 =

AB→C D→E CG→D
C→A D→G CE→A
BC→D BE→C CE→G
CD→B CG→B

Compute step two:

CG→B {CG}$^+$={CGDBAE} B∈{CGDBAE} delete CG→B
CE→A {CE}$^+$={CEGADB} A∈{CEGADB} delete CE→A

R=

AB→C D→E CE→G
C→A D→G
BC→D BE→C
CD→B CG→D

b) Partial Functional Dependency:

Refer to Question No. 3(iv) of Long Answer Type Questions.

Non-transitive dependency:

Refer to Question No. 5 of Long Answer Type Questions.

c) The following relation r is a counterexample to the rule.

r:

A	B	C
a_1	b_1	c_1
a_1	b_1	c_2

Note: $A \rightarrow B$ and $C \rightarrow B$, (since no 2 tuples have the same C value, $C \rightarrow B$ is true trivially). However, it is not the case that $A \rightarrow C$ since the same A value is in two tuples, but the C value in those tuples disagree.

11. Suppose you are given a relation *R* with four attributes, *ABCD*. For each of the following sets of FD's, assuming those are the only dependencies that hold for *R*, do the following:

i) Identify the candidate key(s) for *R*.

ii) Identify the best normal form that *R* satisfies (1NF, 2NF, 3NF, BCNF).

iii) If *R* is not in BCNF, decompose it into a set of BCNF relations that preserve the dependencies:

I) $C \rightarrow D, C \rightarrow A, B \rightarrow C$

II) $B \rightarrow C, D \rightarrow A$

III) $ABC \rightarrow D, D \rightarrow A$

IV) $A \rightarrow B, BC \rightarrow D, A \rightarrow C$

V) $AB \rightarrow C, AB \rightarrow D, C \rightarrow A, D \rightarrow B$.

[WBUT 2010]

Answer:

(i)
I) Candidate keys: B
II) Candidate keys: BD
III) Candidate keys: ABC, BCD
IV) Candidate keys: A
V) Candidate keys: AB, BC, CD, AD

(ii)
I) R is in 2NF but not 3NF
II) R is in 1NF but is not 2NF
III) R is in 3NF but not BCNF.
IV) R is in 2NF but not 3NF (because of the FD: BC → D).
V) R is in 3NF but not BCNF (because of C → A).

(iii)
I) both C → D and C → A violate BCNF. One way to obtain a lossless-join decomposition is to decompose R into AC, BC, and CD.
II) Both B → C and D → A are BCNF violations. The decomposition: AD, BC and BD (obtained by first decomposing to AD, BCD) is BCNF, lossless, dependency preserving.
III) ABCD is not in BCNF because of D → A since D is not a key. However if we split up R as AD, BCD we cannot preserve the dependency ABC → D. So there is no BCNF decomposition that preserves the dependencies.
IV) BC → D violates BCNF as BC is not a (super)key (does not contain a key). We split R into BCD and ABC.
V) C → A and D → B both cause violations. So decompose into: AC and BCD but this does not preserve the FDs AB → C and AB → D. Moreover, BCD is still not BCNF because D → B. So we need to decompose further into: AC, BD, CD. Now, we could attempt to revive the lost functional dependencies by adding the relations ABC and ABD. But we can see that these relations are not in BCNF form. Therefore, there is no BCNF decomposition.

12. a) Consider the relation R = {A, B, C, D, E, F, G , H, I, J} and the set of functional dependencies:
F= {AB→C, A→DE, B→F, F→GH, D→IJ}
Decompose R into 3NF. **[WBUT 2011]**

Answer:
3NF

Decomposed Relations	Result	using
R1	ABC	AB→C
R2	ADE	A→DE
R3	BF	B→F
R4	FGH	F→GH
R5	DIJ	D→IJ

The five 3NF decompositions are presented above in the 'Result' column.

b) What do you mean by lossless and dependency preserving decomposition? **[WBUT 2011]**

Answer:

Lossless decomposition:

Refer to Question No. 5 of Short Answer Type Questions.

Dependency preserving decomposition:

Dependency preservation is a desirable issue in database design. While updating the database, it is necessary to check that spurious tuples should not generate. To test the dependency preservation, let us consider the schema R={R1, R2,.....,Rn} and F be the set of corresponding functional dependencies. The functional dependencies of F^+ (Closure of F in R) should hold only the FD's in R.

Let F' = {F1, F2, F3,.....Fn}, Where F' is a set of functional dependencies on R. But in general $F' \neq F$. However, it may be $F'^+ = F^+$

If it is so, then every functional dependency in F is implied by F' and F' is satisfied, then F is also satisfied. A decomposition which satisfies the equation $F'^+ = F^+$ is dependency preserving decomposition.

13. a) When do we call a relation is in 3NF? **[WBUT 2013]**

Answer:

A relation R is in third normal form (3NF) if and only if it is in 2NF and every non-key attribute is non-transitively dependent on the primary key. An attribute C is transitively dependent on attribute A if there exists an attribute B such that: A → B and B → C. Note that 3NF is concerned with transitive dependencies which do not involve candidate keys. A 3NF relation with more than one candidate key will clearly have transitive dependencies of the form: primary_key → other_candidate_key → any_non-key_column

SUPPLIER_PART (supplier_no, supplier_name, part_no, quantity)

Functional Dependencies:

We assume that supplier_name's are always unique to each supplier. Thus we have two candidate keys:

(supplier_no, part_no) and (supplier_name, part_no)

Thus we have the following dependencies:

(supplier_no, part_no) → quantity

(supplier_no, part_no) → supplier_name

(supplier_name, part_no) → quantity

(supplier_name, part_no) → supplier_no

supplier_name → supplier_no

supplier_no → supplier_name

b) Consider the relation assignment {worker_id, building_id, startdate, name skilltype} and FDs are {worker_id->name, (worker_id, building_id)->startdate}. Is the relation in 2NF? If not, then make it in 2NF. **[WBUT 2013]**

Answer:

worker_id →name

worker_id, building_id)→startdate

These FD's indicates partial dependencies of non key attributes on the key attribute Worker_id in relation (1). In the 2nd relation startdate is dependent on the composite key.

It is not in 2NF. The 2NF relations are:

R1

worker_id	name	skilltype

R2

worker_id	building_id	startdate

c) Describe Boyce-Codd normal form with example. **[WBUT 2013]**

Answer:

Boyce-Codd normal form (or BCNF or 3.5NF) is a normal form used in database normalization. It is a slightly stronger version of the third normal form (3NF). A table is in Boyce-Codd normal form if and only if for every one of its nontrivial dependencies $X \rightarrow Y$, X is a super key—that is, X is either a candidate key or a superset thereof.

In the definition of BCNF each FD X →Y, X *must always* be a superkey. For 3NF, however, X must be a superkey *only* if Y is not part of any other relation key (i.e. if Y is non-prime).

Only in rare cases does a 3NF table not meet the requirements of BCNF. A 3NF table which does not have multiple overlapping candidate keys is guaranteed to be in BCNF. Depending on what its functional dependencies are, a 3NF table with two or more overlapping candidate keys may or may not be in BCNF. A relation R is in 3NF if and only if every dependency A->B satisfied by R meets at least ONE of the following criteria: 1. A->B is trivial (i.e. B is a subset of A)

2. A is a superkey 3. B is a subset of a candidate key BCNF doesn't permit the third of these options. Therefore BCNF is said to be stronger than 3NF because 3NF permits some dependencies which BCNF does not.

d) What is Query Tree? Why we need query tree? Consider the query "SELECT EMP_NAME FROM EMPLOYEE, WORK_ON, PROJECT WHERE PROJECT_NAME='ASSEMBLY' AND PRJ_NO='P1' AND JPOIN_DATE='21-12-12'. Construct a query tree for this query. **[WBUT 2013]**

Answer:

Query Tree is an internal depiction of SQL statements. The rule system is located between the query parser and the planner. It takes the output of the parser and each built of an SQL is stored separately.

14. a) Prove or disprove the following:
i) if AB→C, A→D, CD→EF then AB→F
ii) if XW→Y, and XY→Z, then X→(Z–W) **[WBUT 2014]**

Answer:

i) Given AB→C => AB->AC [Augmentation] ... (1)
A→D => AC->CD ... (2)
By Transitivity from (1) & (2)
AB→CD ... (3)
Given CD → EF ... (4)
By Transitivity from (3) & (4) AB→EF
==> AB→E, AB→F ---[By Decomposition] ... (5)
Thus AB→F proved

ii) if XW→Y, and XY→Z, then X→(Z–W)
XW→Y => XW→XY ... (1)
Given XY→Z ... (2)
By pseudo transitivity Xw → Z
The inference X→(Z–W) is false

b) A table R has attributes P, Q, R, S, T, U, V, W and satisfies the following functional dependencies:
PQR→S, PQ→U, Q→V, R→W
i) What are the candidate keys?
ii) Is this an irreducible set of functional dependencies? **[WBUT 2014]**

Answer:

PQ→U, Q→V,
Given PQR→S
Augmented PQ→U as PQR→UR
Augmented R→W as UR→UW
Hence PQR→UW
Thus PQR→SUW
PQ→U
Q→V
Are the candidate keys and the sets are irreducible

c) Consider the relation R(A, B, C, D, E) with the set of F = {A→C, B→C, C→D, DC→C, CE→A}. Suppose the relation has been decomposed by the relations R1(A, D), R2(A, B), R3(B, E), R4(C, D, E) and R5(A, E) is this decomposition lossy or lossless? Justify your answer.

Answer: **[WBUT 2014]**

Refer to Question No. 9 of Long Answer Type Questions.

d) If 'α' be the set of attributes then write the algorithm to compute 'α+', (Closure of attribute set). **[WBUT 2014]**

Answer:

Initialise $\alpha^+_{<current>} = \alpha$

Repeat until

$\alpha^+_{<current>} = \alpha^+_{<new>}$

For all $X \rightarrow Y$ in α such that $X \subseteq \alpha^+$

$\alpha^+_{<new>} \leftarrow \alpha^+_{<current>}$ XY /* Return the closure of α */

e.g., 1

Let x= {BCD} and the following functional dependencies holds in α

{A → BC, CD → E, E→C, D→AEH, ABH→BD, DH→BC}

compute the closure α^+ of X under α

Initialise $\alpha^+_{<current>} = \{BCD\}$

From the relation CD → E in α s.t $E \subseteq \alpha^+_{<current>}$

$X^+_{<new>} = \{BCDE\}$

From the relation D→AEH in F s.t $AEH \subseteq X^+_{<current>}$

$X^+_{<new>} = \{ABCDEH\}$

Thus X^+ is {ABCDEH}

15. a) Why do we normalize a relation? Describe the anomalies. **[WBUT 2014]**

Answer:

1st Part:

Normalization is a technique for producing a set of relations with desirable properties, given the data requirements of an enterprise. The process of normalization is a formal method that identifies relations based on their primary or candidate keys and the functional dependencies among their attributes.

2nd Part: ***Refer to Question No. 1 of Long Answer Type Questions.***

b) Consider each order has unique order_id for each order, following information are stored:

order_id, order_dt, customer_name, customer_address, salesman_name, salesman_address and for each requested item store item_code, item_name, quantity and rate.

Further assume, following functional dependencies:

salesman_name→ salesman_address

customer_name→ customer_address

order_id→order_dt, salesman_name, customer_name

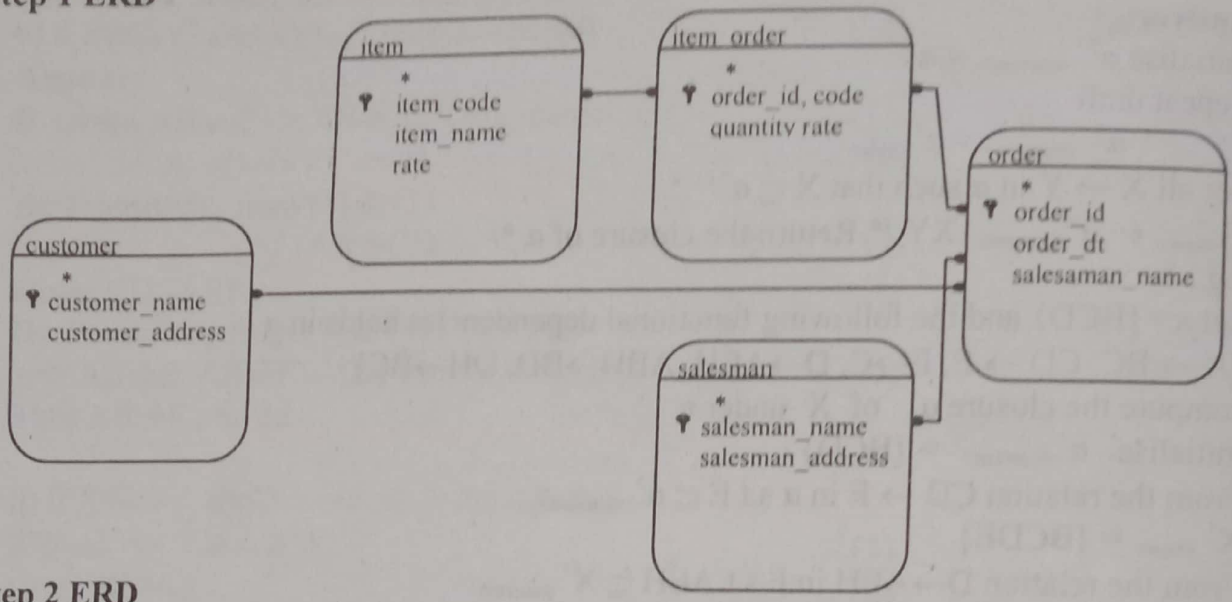
order_id, item_code→quantity

item_code→item_name, rate

Normalize the database up to 3NF showing the steps. Indicate PRIMARY & FOREIGN KEYS. **[WBUT 2014]**

Answer:

Step 1 ERD



Step 2 ERD

Primary Key & Foreign Key relations

Table Name	Primary key	Foreign Key	Reference Table	Normal Form
Salesman	salesman_name	salesman_name	Orderfile	3NF
Customer	customer_name	customer_name	Orderfile	3NF
Item	Item_Code	Order_id, item_code	Item_Order	3NF
Order	Order_It	Salesman_Name	Order	3NF
Item_Oder	Order_IT,IT_Code	Order_it	Order	3NF

c) Is a table with only two columns that is in 3NF, always is in BCNF. What about a table with three columns that is in 3NF? Explain your answer and give examples to support it.

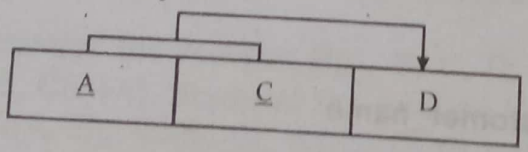
Answer: **[WBUT 2014]**

Definition-3NF

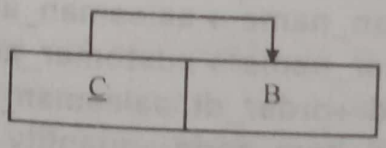
A relation R is in third normal form (3NF) if it is in 2NF and every non-key attribute of R is non-transitively dependent on each candidate key of R.

Definition-BCNF

A relation R is said to be in BCNF whenever X → A holds in R, and A is not in X, then X is a candidate key for R.



3NF and BCNF



3NF and BCNF

d) What is MVD's? What do you mean by lossless (or non-additive) join property of decomposition?

Answer:

Refer to Question No. 6 of Long Answer Type Questions.

16. a) Consider the following two sets of functional dependencies: [WBUT 2015]
X = {A→C, AC→D, E→AD, E→H} and Y = {A→CD,E→AH}.
Check whether or not they are equivalent.
b) Consider R = (A, B, C, D, E) and set of functional dependencies F = {A→BC, CD→E, B→D, E→A} to answer the following questions:
(i) List the candidate keys for R.
(ii) Show that the following decomposition of the schema R into (A,B,C) and (A, D, E) is lossless-join decomposition if the above given set F of functional dependencies holds.
(iii) Compute the canonical cover F_C.
c) What normal form is the following relation in? Discuss stuff (D, O, N, T, C, R, Y) FD's are DO→NTCRY, CR→D, D→N.

Answer:

a) It is to be proved that X is covered by Y

$\{A\}^+=\{A,C,D\}$ (wrt to Y). Since $\{C\} \subseteq A^+$, A→C can be inferred from Y.

$\{AC\}^+=\{A,C,D\}$ (wrt to Y). Since $\{D\} \subseteq \{AC\}^+$, AC→D is covered by Y.

$\{E\}^+=\{E,A,H,C,D\}$ (wrt Y), Since $\{AD\} \subseteq \{E\}^+$, E→AD is covered by Y.

Since $\{H\} \subseteq \{E\}^+$, E→H is covered by Y.

Prove that Y is covered by X:

- $\{A\}^+=\{A,C,D\}$ (wrt X),

Since $\{CD\} \subseteq \{A\}^+$, A→CD is covered by X.

- $\{E\}^+=\{E,A,D,H,C\}$ (with respect to X)

$\{A,H\} \subseteq \{E\}^+$, E→AH is covered by X.

Since X covers Y and Y covers X, X and Y are equivalent

b) i) Since, $A \to BC$, So, $A \to B$ and $A \to C$

Again $B \to D$

So, $A \to D$

$\therefore A \to BCD$

Again $CD \to E$ & $A \to CD$

So $A \to E$.

$\therefore A \to ABCDE$

$\therefore$ So A is a Candidate Key.

ii) $R1 \cap R2 = A$; $(A \to BC)$

Implies $(A \to ABC)$

Implies $(R1 \cap R2 \to R1)$

Imples this is lossless-join decomposition.

iii) Starting with $A \to BC$, we can conclude: $A \to B$ and $A \to C$.

Since $A \to B$ and $B \to D$, $A \to D$ (decomposition is transitive)

Since $A \to CD$ and $CD \to E$, $A \to E$ (union, decomposition, transitive)

Since $A \to A$, we have (reflexive)

$A \rightarrow ABCDE$ from the above steps (union)
Since $E \rightarrow A$, $E \rightarrow ABCDE$ (transitive)
Since $CD \rightarrow E$, $CD \rightarrow ABCDE$ (transitive)
Since $B \rightarrow D$ and $BC \rightarrow CD$, $BC \rightarrow ABCDE$ (augmentative, transitive)
Also, $C \rightarrow C$, $D \rightarrow D$, $BD \rightarrow D$, etc.
Therefore, any functional dependency with A, E, BC, or CD on the left hand side of the arrow is in $F+$, no matter which other attributes appear in the FD. Allow * to represent any set of attributes in R, then $F+$ is
$BD \rightarrow B$, $BD \rightarrow D$, $C \rightarrow C$, $D \rightarrow D$, $BD \rightarrow BD$, $B \rightarrow D$, $B \rightarrow B$, $B \rightarrow BD$.

c) The prime attributes are D, and O and non-prime attributes are N,T,C,R, and Y.
A relation scheme is in 2NF, if all the non-prime attributes are fully functionally dependent on the relation key(s).
From the set of FDs we can see that the non-prime attributes (N,T,C,R, and Y) are fully functionally dependent on the prime attributes, therefore, the relation is in 2NF.
A relation scheme is in 3NF, if for all the non-trivial FDs in F+ is of the form **CR→D**, D is a prime attribute and in FD , **D→N , D is a superkey**. Thus from the set of FDs we see that for all the FDs, this is satisfied, therefore, the relation is in 3NF.
A relation scheme is in BCNF, if for all the non-trivial FDs in F+ the form **CR→D, D→N, D** is a superkey but not CR. Hence, the given relation scheme is in BCNF.

17. Consider the relation $R = \{A, B, C, D, E\}$ and the set of functional dependencies:

$$F = \{AD \rightarrow B, B \rightarrow C, C \rightarrow D\}$$

Find out the candidate key. **[WBUT 2016]**

Answer:
The attribute E is non participative attribute, so no functional dependency is provided.
So $(AD)^+ = \{B, C, D\}$ using (AD→B, B→C, C→D)
$(AD)^+ = \{A, B, C, D\}$
$(B)^+ = (C, D)$ using (B→C, C→D)
$(C)^+ = (D)$
Candidate Key is **minimal set of attributes** of a relation which can be used to identify a tuple uniquely. So AD is the candidate key (Either include attribute E or exclude it)

18. a) Find out the closure of attribute set (AD) i.e. $(AD)^+$ in the R. Set of FD's F are as given below: **[WBUT 2017]**

$R = \{A, B, C, D, E\}$

$FD = \{B \rightarrow CD, D \rightarrow E, B \rightarrow A, E \rightarrow C, AB \rightarrow B\}$.

b) Find out the candidate keys for R.
c) Consider the following two sets of FDs:

$F = \{A \rightarrow C, AC \rightarrow D, E \rightarrow AD, E \rightarrow H\}$

$G = \{A \rightarrow CD, E \rightarrow AH\}$.

Check whether they are equivalent. Justify your answer.

Answer:

Closure of (AD)

$FD = \{B \to CD, D \to E, B \to A, E \to C, AB \to B\}$

Given:

D=ED {As D→E}

(AD)+=EDABC {By augmentation and E→C, AB→B}

(AB)+=BACDE

(A)+=A

(B)+=CDE

(D)+=EC

Closure of (AD)+ is [ABCDE]..... (a)

There can be 24 Super keys, however,

AD and AB are candidate keys..... (b)

$F = \{A \to C, AC \to D, E \to AD, E \to H\}$

Closure of F^+=(EH) considering E→H

=(ACDEH) considering E→AD, A→C

Similarly

Closure of G^+=(EAH) considering E→AH

=(ACDEH) considering A→CD

Thus F^+ and G^+ are equivalent..... (c)

19. a) If $R = (A, B, C, D)$ and the FDs are $\{AB \to CE, E \to AB, C \to D\}$. Why R is in 2NF, but not in 3NF? Explain.

b) Show that if a relation schema is in BCNF, then it is in 3NF but if a relation schema is in 3NF then it is not necessary in BCNF. Give examples. [WBUT 2017]

Answer:

a) AB→CE, E→AB, C→D

Applying Armstrong:

AB→C ... (1)

AB→E ... (2)

E→AB ... (3)

C→D ... (4)

Closure of (AB) is

(AB)+ = CEABD, which implies AB is a candidate key.

No (4) shows transitive dependency exists between C and D.

Thus it is in 2NF but not in 3NF

b) Relations that are in 3NF are also in BCNF. But, a 3NF relation is not in BCNF.A schema R is in BCNF with respect to a set F of functional dependencies, if for all functional dependencies in F+ of the form α→β, where α ⊆R and β ⊆ R, at least one of the following holds:

α →β is trivial (i.e.,β ⊆α).

α is a super key for R.

A relational schema R is in 3NF if for every functional dependency
$\alpha \rightarrow \beta$ (where α is a subset of the attributes and β is a single attribute) either $\alpha \rightarrow \beta$ is trivial
(i.e., $\beta \subseteq \alpha$)
α is a super key,
β is part of some key for R.
Thus the definition of 3NF permits a few additional functional dependencies involving key attributes that are prohibited by BCNF. Or, we can say that All BCNF is in 3NF, but all 3NF is not in BCNF.
A 3NF relation may be in BCNF only if (a) the candidate keys in the relation are composite keys (that is, they are not single attributes),(b) there is more than one candidate key in the relation, and(c) the keys are not disjoint, that is, some attributes in the keys are common.
Example:
R = (ABCD)
F = {AB→C, B→D; C→A}

First, let's compute the attribute closure:
A+ = A
B+ = BD
C+ = AC
D+ = D
AB+ = ABCD
BC+ = ABCD

So our candidate keys are AB and BC.
Now we start the algorithm:
AB→C
Does the first violate BCNF, i.e. is AB a key for R? The answer is yes, so no violation.
B→D
Does this violate BCNF? Yes, because B is not a key. So we create 2 relations:
(BD) (ABC)
For (BD) the candidate key is B. The only FD that applies here is B→D, so it is in BCNF.
For (ABC), the candidate keys are AB and BC. The first FD applies, AB→C, and AB is a key so it is in BCNF.
The second FD doesn't apply (there is no D in it), and the third FD does apply (C→A).
Is C a key? No, so we need to decompose by creating a new relation schema made up of the FD, and pulling out the right side of the FD from the original
(BD)(CA)(BC)
BD is still in BCNF as before CA has C as the candidate key, and the only FD that applies is C→ A. It is in BCNF. BC has BC as the candidate key, and no FDs apply, so it is in BCNF
Our final decomposition is:
(BD)(CA)(BC)
(BD)(ABC)

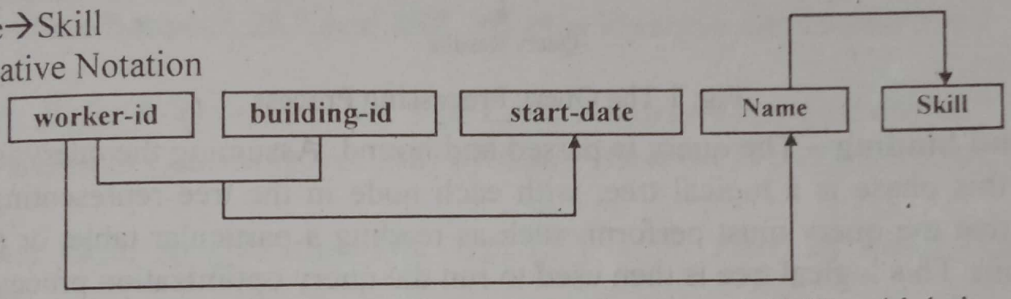
In this situation, BD was in BCNF, so it is in 3NF by definition. (ABC) still has 2 candidate keys, AB and BC. The first FD applies, AB → C, and AB is a key so it is in BCNF (and by definition 3NF). The second FD doesn't apply (there is no D in it), and the third FD does apply (C →A). While this one violated BCNF, it doesn't violate 3NF because A is a part of a key of (ABC). This Decomposition is then 3NF and dependency preserving.

20. a) Why do we need normalization? **[WBUT 2018]**
b) Relation assignment (worker-id, building-id, start-date, name, skill). FD's are {worker-id → name, {worker-id, building-id} → start date, name → skill}. Is the relation in 3NF, if not then make it in 3NF.

Answer:

a) ***Refer to Question No. 15(a) of Long Answer Type Questions.***

b) FD: WorkerID → Name
FD: {WorkerID, BLDG}→ StartDate,
FD: Name→Skill
FD Alternative Notation



This diagram clearly shows the Offending FD is Name→Skill, which is a transitive dependency and contrary to the hypothesis of 3NF.
To convert the given relations into 3nf, the following decompositions are necessary according to the definition of 3NF, the determinant is an attribute (or collection of attributes) that determines another attribute i.e. for every X →Y, X is a key (where X is a single or a combination of attributes).
Hence, the relations 's in 3NF are:
{WorkerID, Bldg-ID, Start-Date}
{worker-id, Name}
{Name, Skill}

21. a) What is the utility of Query Tree? **[WBUT 2018]**
b) Describe different phase of a Query Tree.

Answer:

a) A tree data structure that corresponds to a relational algebra expression. It represents the input relations of the query as *leaf nodes* of the tree, and represents the relational algebra operations as *internal nodes*.
An execution of the query tree consists of executing an internal node operation whenever its operands are available and then replacing that internal node by the relation that results from executing the operation.

b) In order to arrive at what it believes to be the best plan for executing a query, the Query Processor performs a number of different steps; the entire query processing process tree is shown Fig: 1.

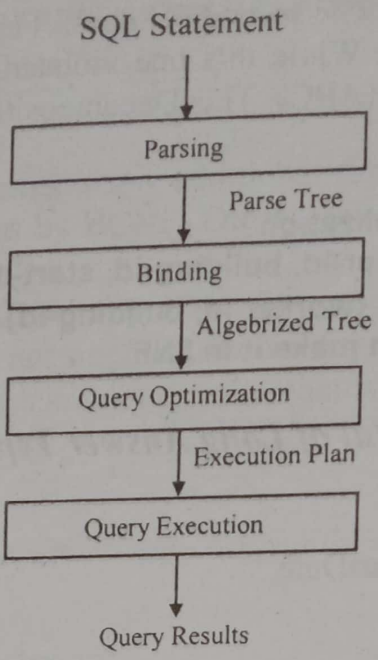


Fig: 1 The Query Processing Process

Parsing and binding – The query is parsed and bound. Assuming the query is valid, the output of this phase is a logical tree, with each node in the tree representing a logical operation that the query must perform, such as reading a particular table, or performing an inner join. This logical tree is then used to run the query optimization process.

A **query optimizer** generates one or more **query** plans for each **query**, each of which may be a mechanism used to run a **query**. The most efficient **query** plan is selected and used to run the **query**. Database users do not typically interact with a **query optimizer**, which **works** as follows:

Parsing and translation

Translate the query into its internal form. This is then translated into relational algebra. Parser checks syntax, verifies relations

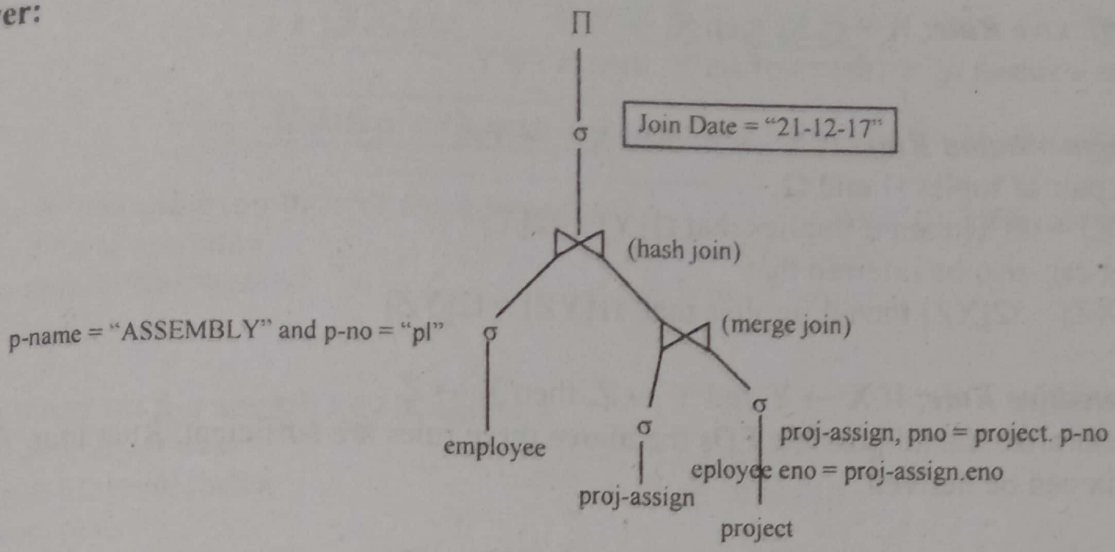
Evaluation

The query-execution engine takes a query-evaluation plan, executes that plan, and returns the answers to the query.

22. Consider the employee query "Select work-on, e-name, from employee, proj-assign project where p-name = "ASSEMBLY" and p-no = "p1" and Join-Date = "21-12-17" and employee.eno = proj-assign. eno and proj-assign, pno=project. p-no.

[WBUT 2018]

Answer:



23. a) Explain the terms 'full functional dependency' and 'multivalued dependency' with example. **[WBUT 2019]**

b) Differentiate between 2NF and 3NF. What is lossless decomposition?

Answer:

a) 1st Part: *Refer to Question No. 5 of Long Answer Type Questions.*

2nd Part: *Refer to Question No. 2(a) of Long Answer Type Questions.*

b) 1st Part: *Refer to Question No. 3(b) (2nd & 3rd Part) of Long Answer Type Questions.*

2nd Part: *Refer to Question No. 10 [2nd Part] of Short Answer Type Questions.*

c) What is closure? Explain with example.

Answer:

1st Part: *Refer to Question No. 8 of Short Answer Type Questions.*

2nd Part: *Refer to Question No. 3 of Short Answer Type Questions.*

Rest Part:

Let, R = (A, B, C, D, E)

$A \rightarrow BC$, $CD \rightarrow E$, $B \rightarrow D$, $E \rightarrow A$

24. Write short notes on the following:

a) Armstrong's axioms **[WBUT 2009, 2017]**

b) Functional dependency **[WBUT 2013]**

c) BCNF **[WBUT 2018]**

Answer:

a) Armstrong's Axioms (Inference Rules for FDs):

For any given relation, there can be many FDs. Let us denote F as the set of FDs of a given relation R. There can be many other FDs that can be derived (or, inferred) from F.

1. Reflexive Rule: If $Y \subseteq X$, then $X \rightarrow Y$
If Y is a subset of attributes of set X, then $X \rightarrow Y$

2. Augmentation Rule: If $X \rightarrow Y$, then $XZ \rightarrow YZ$
For a pair of tuples t1 and t2,
If $t1[X] = t2[X]$ then it implies that $t1[Y] = t2[Y]$
And it can also be inferred that
If $t1[XZ] = t2[XZ]$ then it implies that $t1[YZ] = t2[YZ]$

3. Transitive Rule: If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
To generalize the all possible FDs the above three rules are sufficient. Rule four, five and rule six can be derived.

4. Decomposition: If $X \rightarrow YZ$ then $X \rightarrow Y$ and $X \rightarrow Z$

5. Union: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$

6. Pseudo transitivity rule: If $X \rightarrow Y$ holds and $ZY \rightarrow W$ holds then $XZ \rightarrow W$ holds

b) Functional Dependency:
A functional dependency (FD) is a constraint between two sets of attributes in a relation from a database.
Given a relation R, a set of attributes X in R is said to functionally determine another attribute Y, also in R, (written $X \rightarrow Y$) if and only if each X value is associated with precisely one Y value. Customarily we call X the determinant set and Y the dependent attribute. Thus, given a tuple and the values of the attributes in X, one can determine the corresponding value of the Y attribute. For the purposes of simplicity, given that X and Y are sets of attributes in R, $X \rightarrow Y$ denotes that X functionally determines each of the members of Y—in this case Y is known as the dependent set. Thus, a candidate key is a minimal set of attributes that functionally determine all of the attributes in a relation.

c) BCNF: ***Refer to Question No. 2 of Short Answer Type Questions.***

STORAGE STRATEGIES

Multiple Choice Type Questions

1. One of the shortcomings of file system is [WBUT 2009, 2011, 2017]
   a) data availability
   b) fixed records
   c) sequential records
   d) lack of security

Answer: (d)

2. An index on the search key is called a [WBUT 2012]
   a) primary index
   b) secondary index
   c) multi-level index
   d) all of these

Answer: (a)

3. Which one of the following is not an indexing technique? [WBUT 2019]
   a) Primary index
   b) Secondary index
   c) Multilevel index
   d) Sequential index

Answer: (d)

Short Answer Type Questions

1. What are the four main differences between a file-processing system and a DBMS? [WBUT 2006]

OR,

Write down the differences between DBMS and Traditional File Processing System. [WBUT 2009]

OR,

Distinguish between file management system and database management system. [WBUT 2017]

OR,

What are the advantages of database over traditional file system? [WBUT 2018]

Answer:

In file system approach, each user defines and implements the needed files for a specific application to run. For example in sales department of an enterprise, One user will be maintaining the details of how many sales personnel are there in the sales department and their grades, these details will be stored and maintained in a separate file. Another user will be maintaining the salesperson salary details working in the concern, the detailed salary report will be stored and maintained in a separate file. Although both of the users are interested in the data's of the salespersons they will be having their details in a separate files and they need different programs to manipulate their files. This will lead to wastage of space and redundancy or replication of data's, which may lead to confusion, sharing of data among various users is not possible, data inconsistency may occur. These files will not be having any inter-relationship among the data's stored in these files.

Therefore in traditional file processing every user will be defining their own constraints and implement the files needed for the applications.
In database approach, a single repository of data is maintained that is defined once and then accessed by many users. The fundamental characteristic of database approach is that the database system not only contains data's but it contains complete definition or description of the database structure and constraints. These definitions are stored in a system catalog, which contains the information about the structure and definitions of the database. The information stored in the catalog is called the metadata, it describes the primary database. Hence this approach will work on any type of database for example, insurance database, Airlines, banking database, Finance details, and Enterprise information database. But in traditional file processing system the application is developed for a specific purpose and they will access specific database only.
The other main characteristic of the database is that it will allow multiple users to access the database at the same time and sharing of data is possible. The database must include concurrency control software to ensure that several users trying to update the same data at the same time, it should maintain in a controlled manner. In file system approach many programmers will be creating files over a long period and various files have different format, in various application languages.
Therefore there is possibility of information getting duplicated, this redundancy is storing same data multiple times leads to higher costs and wastage of space. This may result in data inconsistency in the application, this is because update is done to some of the files only and not all the files. Moreover in database approach multiple views can be created which will not occupy any space.
A multi-user database whose users have variety of applications must provide facilities for defining multiple views. In traditional file system, if any changes are made to the structure of the files if will affect all the programs, so changes to the structure of a file may require changing of all programs that access the file. But in case of database approach the structure of the database is stored separately in the system catalog from the access of the application programs.
Database can be used to provide persistent storage for program objects and data structures that resulted in object oriented database approach. Traditional systems suffered from impedance mismatch problem and difficulty in accessing the data, which is avoided in object oriented database system. Database can be used to represent complex relationships among data's as well as to retrieve and update related data easily and efficiently.
It is possible to define and enforce integrity constraints for the data's stored in the database. The database also provides facilities for recovering hardware and software failures. The backup and recovery subsystem is responsible for recovery. It reduces the application development time considerably when compared to the file system approach and availability of up-to-date information of all the users. It also provides security to the data's stored in the database system.

Long Answer Type Questions

1. What is the difference between Primary index, Secondary index and Clustering index? **[WBUT 2006, 2011]**

Answer:

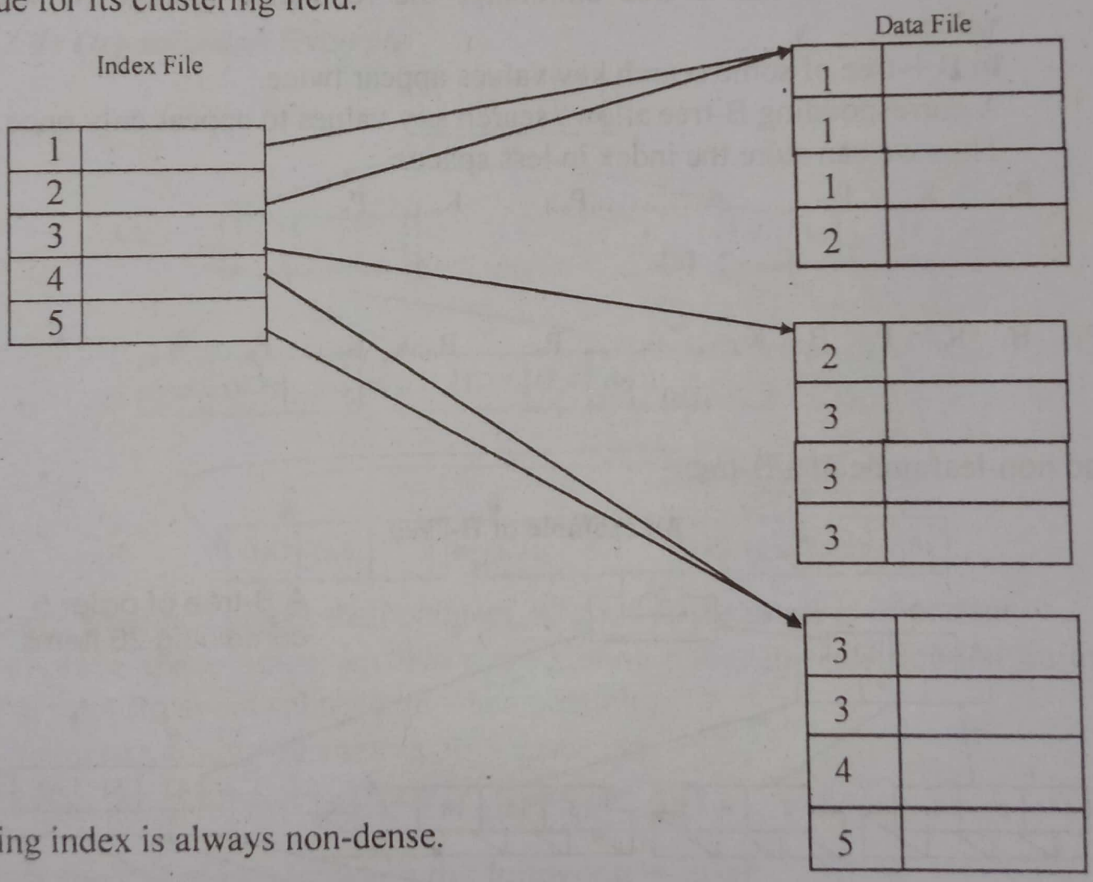
Primary & Secondary Index:

Refer to Question No. 3(a) of Long Answer Type Questions.

Clustering Indexes

If records of a file are physically ordered on a nonkey field, called the clustering field. We can create a clustering index to speed up the retrieval of records that have the same value for the clustering field. This differs from a primary index, which requires that the ordering field of the data file have a distinct value for each record.

The clustering index file, like the primary index file, has two fields. The first field contains distinct values of the clustering field, and the second field contains block pointers. The block pointer points to the first block in the data file that has a record with that value for its clustering field.



Clustering index is always non-dense.

2. What is Blocking Factor? Explain the difference between B-tree and B^+ -tree indexing with proper example. **[WBUT 2008, 2011]**

Answer:

Blocking: refers to storing a number of records in one block on the disk.

Blocking factor (bfr) refers to the number of records per block. There may be empty space in a block and if an integral number of records do not fit in one block **Spanned *Records:*** refer to records that exceed the size of one or more blocks and hence span a number of blocks.

Difference between the B -Tree and B^+-Tree Are

A B-tree of order *m* is an *m*-way tree (i.e., a tree where each node may have up to *m* children) in which:

1. The number of keys in each non-leaf node is one less than the number of its children and these keys partition the keys in the children in the fashion of a search tree
2. All leaves are on the same level
3. All non-leaf nodes except the root have at least $\lceil m / 2 \rceil$ children
4. The root is either a leaf node, or it has from two to *m* children
5. A leaf node contains no more than $m - 1$ keys

- The number *m* should always be odd.
  1. B-tree indices are similar to B +-tree indices.
     - Difference is that B-tree eliminates the redundant storage of search key values.
     - In B +-tree of some search key values appear twice.
     - A corresponding B-tree allows search key values to appear only once.
     - Thus we can store the index in less space.

P_1 K_1 P_2 ……P_{n-1} K_{n-1} P_n

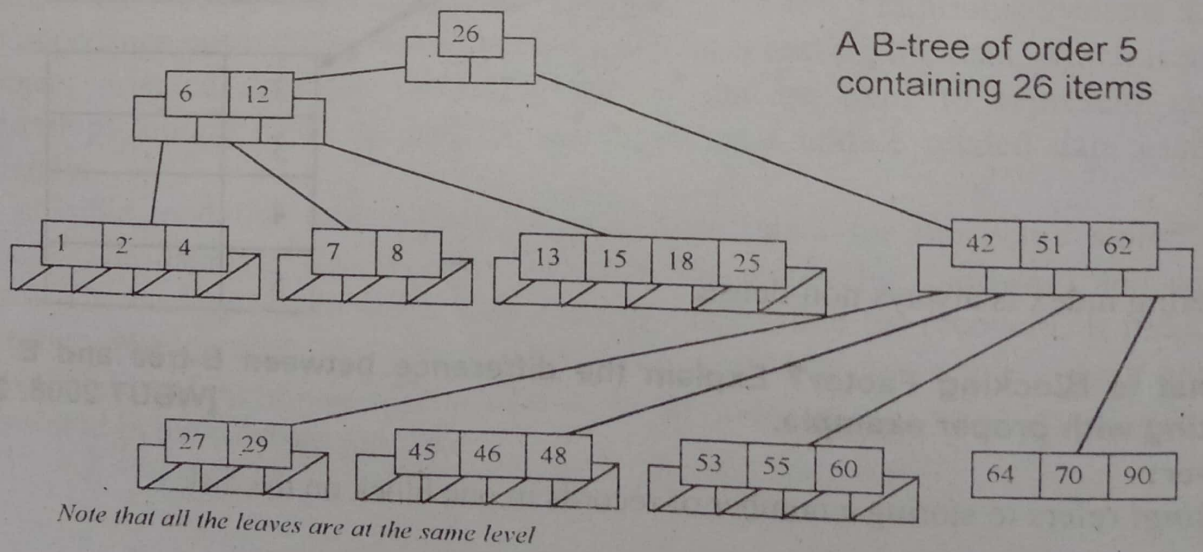
(a)

P_1 B_1 K_1 P_2 B_2 K_2 ……P_{n-1} B_{n-1} K_{n-1} P_n

(b)

Leaf and non-leaf node of a B-tree.

An example of B-Tree

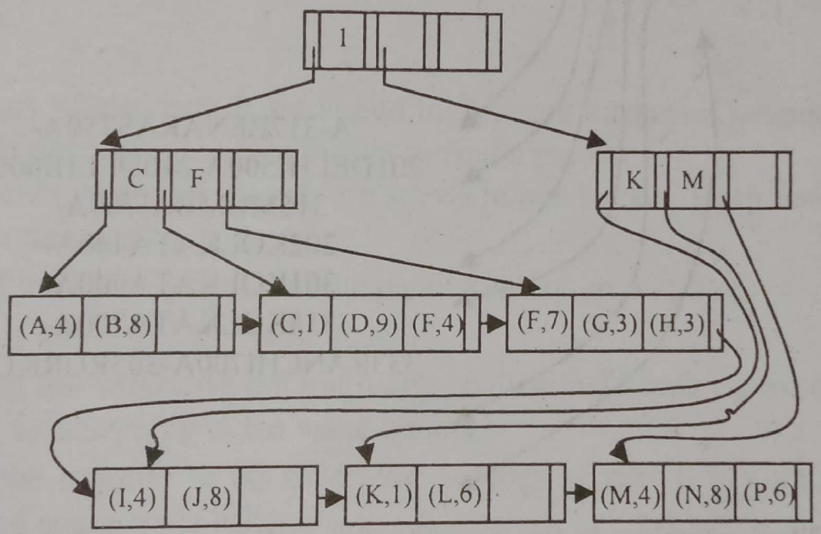


Note that all the leaves are at the same level

B+ -Tree File Organization

1. The B +-tree structure is used not only as an index but also as an organizer for records into a file.
2. In a B +-tree file organization, the leaf nodes of the tree store records instead of storing pointers to records. Since records are usually larger than pointers, the maximum number of records that can be stored in a leaf node is less than the maximum number of pointers in a non-leaf node.
3. However, the leaf node are still required to be at least half full.
4. Insertion and deletion from a B +-tree file organization are handled in the same way as that in a B +-tree index.
5. When a B +-tree is used for file organization, space utilization is particularly important. We can improve the space utilization by involving more sibling nodes in redistribution during splits and merges.
6. In general, if *m* nodes are involved in redistribution, each node can be guaranteed to contain at least [(m−1)n/m] entries. However, the cost of update becomes higher as more siblings are involved in redistribution.

B^+-tree File Organization Example



- Records are much bigger than pointers, so good space usage is important
- To improve space usage, involve more sibling nodes in redistribution during splits and merges (to avoid split/merge when possible)
- involving one sibling guarantees 50% space use
- involving two guarantees at least 2/3 space use, etc.

3. a) State the difference between the following in brief:
Primary Index versus Secondary Index. **[WBUT 2011]**

Answer:

An **index file** consists of records (called **index entries**) of the form

search-key	pointer

Index files are typically much smaller than the original file.

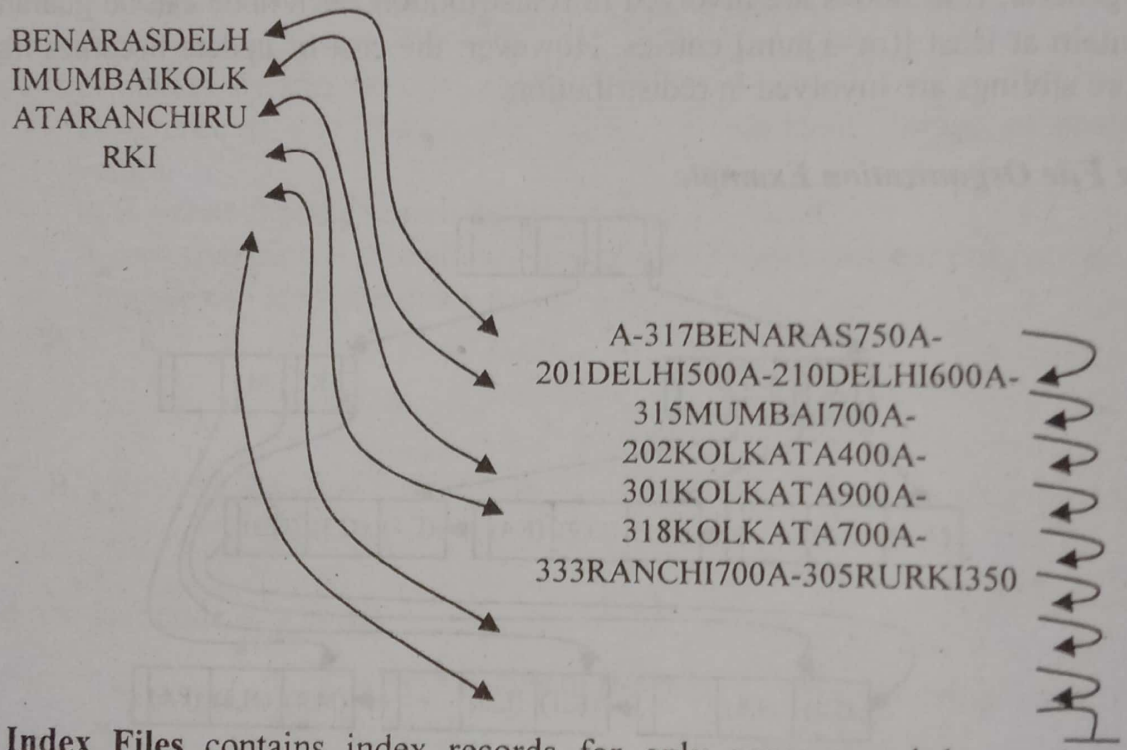
In an **ordered index,** index entries are stored sorted on the search key value. E.g., author catalog in library.

Primary index: in a sequentially ordered file, the index whose search key specifies the sequential order of the file also called **clustering index.** The search key of a primary index is usually but not necessarily the primary key

Secondary index: an index whose search key specifies an order different from the sequential order of the file. It is also called non-clustering index. ***Index-sequential file:*** Ordered sequential file with a primary index. This is also called ISAM - indexed sequential access method.

Dense Index Files

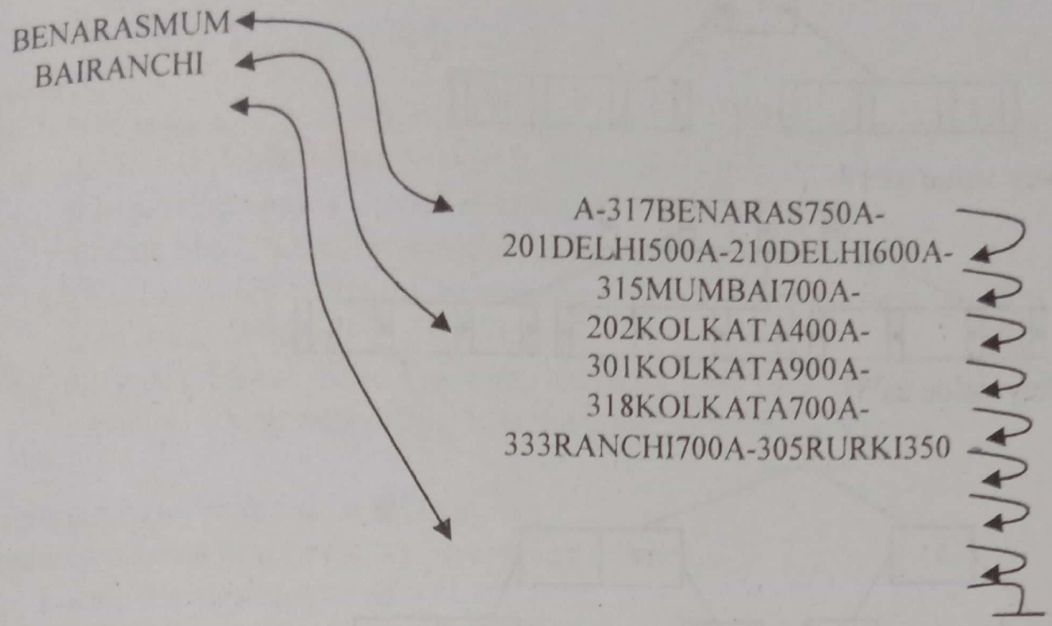
Dense index – Index record appears for every search-key value in the file (in this example we have clustering index).



Sparse Index Files contains index records for only some search-key values. It is applicable when records are sequentially ordered on search-key (i.e., only applicable to clustering indexes). To locate a record with search-key value *K* we:

Find index record with largest search-key value <= *K. Then s*earch file sequentially starting at the record to which the index record points. It involves less space and less maintenance overhead for insertions and deletions. Generally slower than dense index for locating records.

Example of Sparse Index Files



b) What are the causes of bucket overflow in a hash file organization? **[WBUT 2011]**

Answer:

Bucket Overflow:

Open hashing occurs where records are stored in different buckets. Compute the hash function and search the corresponding bucket to find a record.

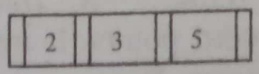
Closed hashing occurs where all records are stored in one bucket. Hash function must be chosen at implementation time.

- Number of buckets is fixed, but the database may grow.
- If number is too large, we waste space.
- If number is too small, we get too many "collisions" resulting in records of many search key values being in the same bucket.
- Choosing the number to be twice the number of search key values in the file gives a good space trade off.

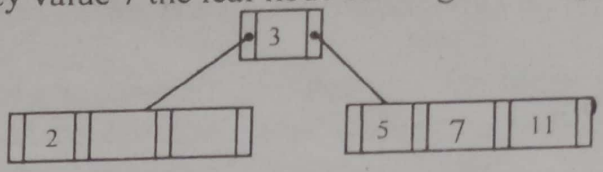
c) Construct a B+ tree for the following set of key values: **[WBUT 2011]**
(2, 3, 5, 7, 11, 17, 19, 23, 29, 31)
Assume that the tree is initially empty and values are added in ascending order. Construct B+ trees for the cases where the numbers of pointers that will fit in one node is as follows:
i) Four **ii) Six** **iii) Eight**

Answer:

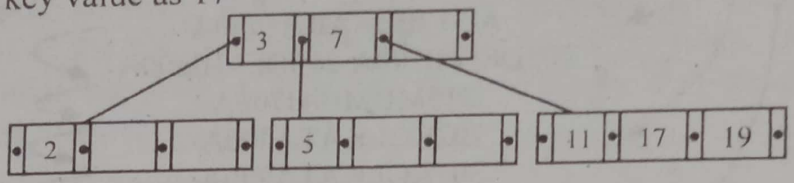
i) With four pointers



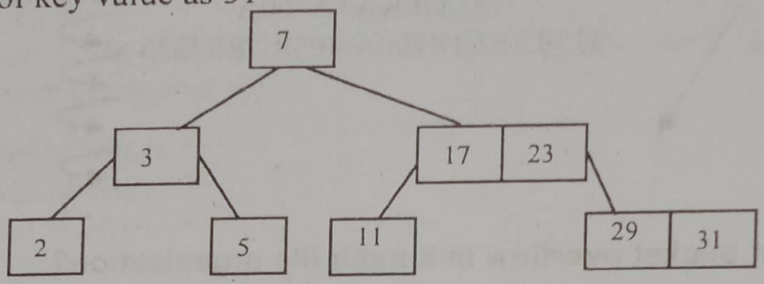
On arrival of key value 7 the leaf node status gets changed



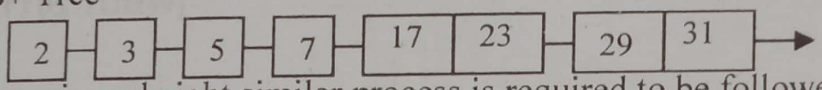
On arrival of key value as 17



On arrival of key value as 31

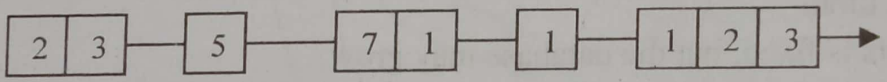


B+ Tree

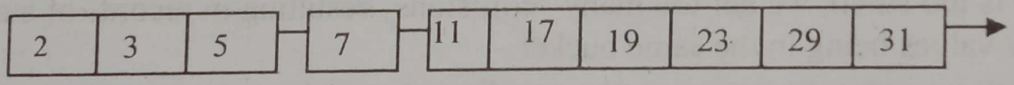


For six and eight similar process is required to be followed The median selection will be done by the formula:(n+1)/2.

ii) B+ tree with 6 pointers:



iii) B+ tree with 8 pointers:



All the key values will be available at the leaf.

4. a) What are dense and sparse indexing? Explain with an example. [WBUT 2012]

Answer:

Dense index — Index record appears for every search-key value in the file (in this example we have clustering index).

Sparse Index Files contains index records for only some search-key values. It is applicable when records are sequentially ordered on search-key (i.e., only applicable to clustering indexes). To locate a record with search-key value *K* we:

Find index record with largest search-key value <= *K*. *Then s*earch file sequentially starting at the record to which the index record points. It involves less space and less maintenance overhead for insertions and deletions. But generally slower than dense index for locating records.

b) Create B^+ tree for the following key: **[WBUT 2012]**

Order = 3

Key : 8, 5, 1, 7, 3, 12, 9, 6.

Answer:

A B^+-tree of order m is a m-way search tree with the following characteristics.

i) the tree is either empty or is of height greater than or equal to 1,
ii) the root node index block has at least 2 children,
iii) all data block nodes (leaf nodes) must be at the same level.,
iv) all index block nodes, other than the root node and the data block nodes, have at least [m/2] children,
v) all index blocks have the same capacity. All data blocks have also the same capacity, which may differ from the capacity of index blocks.

Inserting a New Node into a B^+-tree

In order to insert a new node, we proceed as follows,

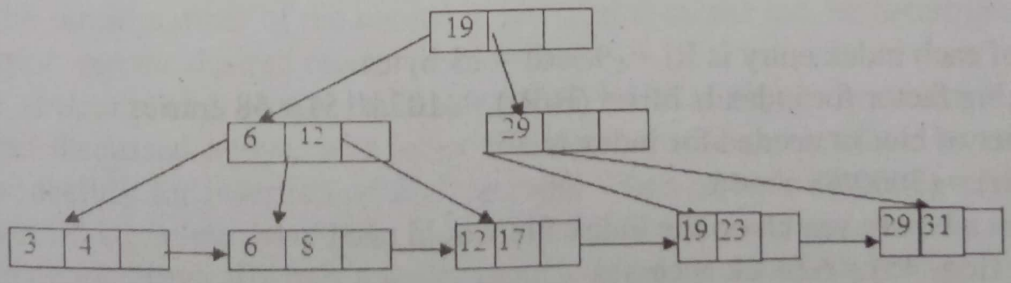
1. Locate the appropriate data block
2. If there is room in the data block, the new record will be inserted in sequence by moving subsequent records if necessary,
3. Splitting a block: If there is no room, a new data block has to be acquired first. Some of the records from the old block are moved into the newly acquired block so that after insertion of the new record the old and the new data block contain nearly an equal number of records.

A new entry in the new data block is inserted into the appropriate index block. If the index block is already full then a new index block is obtained and some of the entries from the old index block are to be moved to the new index block so that they contain nearly an equal number of entries. Similarly, the index block which is at a level immediately above this index block is updated and so on.

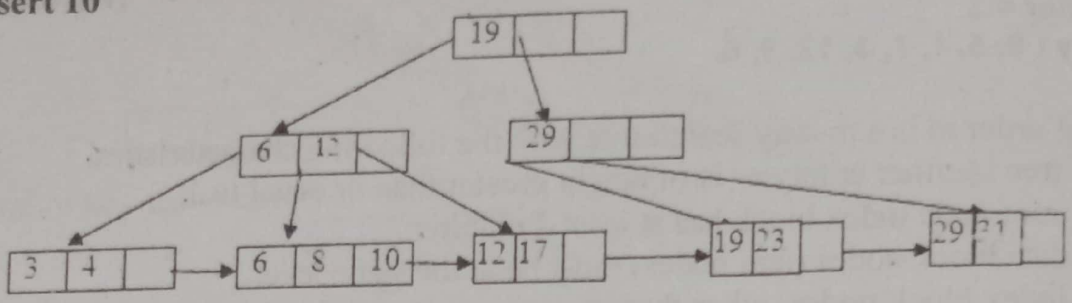
5. Construct a B+ tree for the following set of key values– (3,4,6,8,12,17,19,23,29,31) assume that the tree is initially empty, values are added in ascending order and the number of pointer in one node is 3. Also perform the following operations of B+ tree: (i) insert 10 (ii) insert 11 (iii) delete 29. **[WBUT 2015]**

Answer:

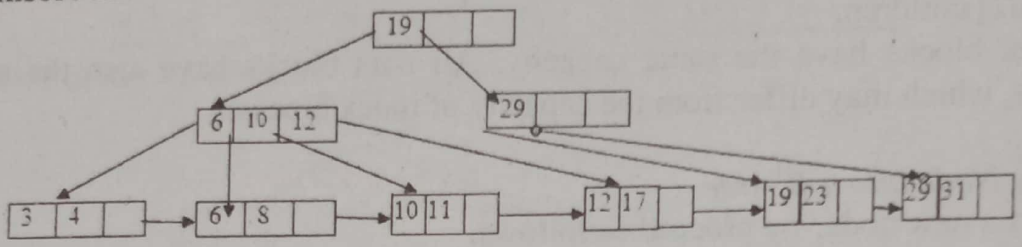
The B+ tree is generated in ascending order. A node (which is not a root) is never allowed to have fewer than [n/2] values/pointers.



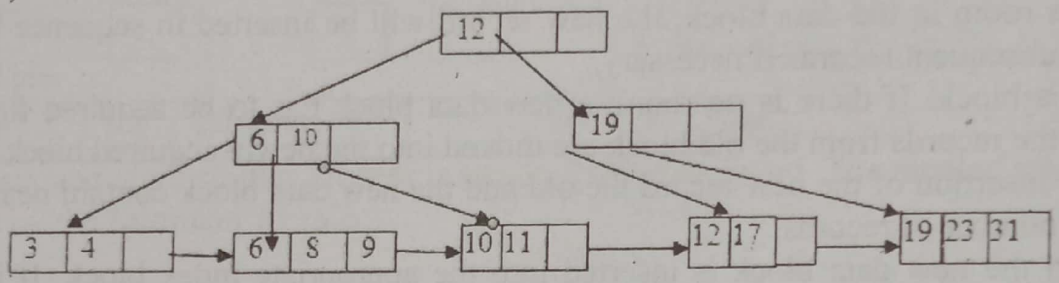
(i) insert 10



(ii) insert 11



(iii) Delete 29



6. Consider the file with $r = 3000$ records (fixed length) of size R = 100 bytes stored on a disk with block size B = 1024 bytes. Suppose each index entry in index file takes 15 bytes (9 bytes for index value, 5 bytes for pointer). What is the number of accessing blocks for clustering index? [WBUT 2015]

Answer:

Block Factor bfr = (B/R) = (1024/100) = 10 records per block

The number of blocks needed for the file

b = (r/bfr) = (30,000/10)= 3000 blocks

A binary search on the data file would need approximately

$\log_2 b = \log_2 3000) = 12$ block accesses

Now if the ordering key field of the index file is V= 9 bytes, and a block pointer is P = 6 bytes

The size of each index entry is Ri = (9 + 6) = 15 bytes

The blocking factor for index is bfri = (B/Ri) = (1024/15) = 68 entries

The number of blocks needed for index is

bi = (ri/bfri) = (3000/68) = 45

To perform a binary search on the index file would need

$(\log_2 bi) = (\log_2 45) = 6$ block accesses

Searching a record using the index, one additional data block access needed. Thus total 6 + 1 = 7 block accesses are needed.

7. Write short notes on the following:

a) B-tree **[WBUT 2013, 2019]**

b) File indexing **[WBUT 2016]**

c) B+ tree **[WBUT 2016]**

OR,

B+ tree file organization **[WBUT 2018]**

d) Ordered Index **[WBUT 2017]**

e) Primary Index and Secondary Index **[WBUT 2018]**

Answer:

a) B-tree:

A B-tree of order m is an m-way tree (i.e., a tree where each node may have up to m children) in which the number of keys in each non-leaf node is one less than the number of its children and these keys partition the keys in the children in the fashion of a search tree. All leaves are on the same level. All non-leaf nodes except the root have at least $\lceil m / 2 \rceil$ children. The root is either a leaf node, or it has from two to m children. A leaf node contains no more than $m - 1$ keys. The number m should always be odd. A corresponding B-tree allows search key values to appear only once. Thus we can store the index in less space.

b) File indexing:

The sorting of a large database file is a very consuming operation. Also, since record searching is done in main memory, all the records, in the case of a sequential search, must be read in the main memory one by one. This process incurs huge I/O time. In general, data in a base file are ordered according to an ordering attribute. A file can have only one ordering attribute which is the primary key of the concerned table. But an application programmer may wish to sort the records according to any of the available attributes. In order to avoid huge sorting and searching times, *indexed sequential access method* (ISAM) has been developed. For indexed sequential search, an auxiliary file called an *index* file is created beforehand. A *file index* is a data structure that is used for speeding up the processes of record search and retrieval in a database. An index is generally created on a key attribute of a database file (index on other attributes are also possible).

In the simplest case, an index file contains only one field containing the set of values of the key attribute. During a record search operation, the index file in main memory and the search process is applied over it (for the required key value). Once the desired key value is found, the serial number of the record in physical database can be determined from the index location and the desired record is read directly.

Index file structure and access mechanism can be of different types. In the simplest situation, as discussed above, and index file comes with an overflow area in order to provide the facility for insertion of new records. When an index entry in maintained for each record of complete base file, it is called *full indexing*. Sometimes, actual base records are divided logically into smaller groups (blocks) and the index file contains only one entry for each record separately. Here search is done for blocks of records only. Since disk I/O is generally performed on a block basis (i.e. one block of records are read

in one access), if a block holds one group of records, both the searching effort and number of disk accesses will be reduced. However, in this case, after a disk access, the read block or records needs to be searched sequentially in the main memory. This method is called partial indexing.

c) B+ tree: ***Refer to Question No. 2 (2nd Part) of Long Answer Type Questions.***

d) ***Refer to Question No. 3(a) of Long Answer Type Questions.***

e) Primary Index and Secondary Index:
Refer to Question No. 3(a) of Long Answer Type Questions.

TRANSACTION PROCESSING

Multiple Choice Type Questions

1. In 2-phase locking a transaction must: [WBUT 2006, 2007, 2008, 2019]
   a) release all its locks at the same time
   b) NOT obtain any new locks once it has started releasing locks
   c) Only obtain locks on items not used by any other transactions
   d) Ensure that deadlocks will never occur

Answer: (d)

2. Transaction follows [WBUT 2007, 2015, 2016]
   a) ACID properties
   b) Non-preemption property
   c) Preemption property
   d) Starvation property

Answer: (a)

3. The concurrency control has the problem of [WBUT 2010]
   a) lost updates
   b) dirty read
   c) unrepeatable read
   d) all of these

Answer: (d)

4. Advantage of locking algorithms in concurrent execution of DB transaction is ..... [WBUT 2011]
   a) deadlock b) concurrency c) consistency d) none of these

Answer: (b)

5. Which phase is not part of a two phase locking protocol? [WBUT 2012]
   a) Growing phase
   b) Shrinking phase
   c) Stabilization phase
   d) None of these

Answer: (c)

6. Which is not an ACID property? [WBUT 2012]
   a) Atomicity b) Integrity c) Consistency d) Durability

Answer: (b)

7. Three transactions attempt to book seats on a flight that has 12 seats available. The transactions are transaction T1 for 3 seats, transaction T2 for 5 seats and transaction T3 for 7 seats. If a schedule that is serializable is executed, the number of seats sold cannot be [WBUT 2014]
   a) 7 b) 8 c) 10 d) 12

Answer: (b)

8. Which of the following is the size of the data item chosen as the unit of protection by a concurrency control program? [WBUT 2014]
   a) Lock
   b) Blocking factor
   c) Granularity
   d) none of these

Answer: (c)

[WBUT 2015]

9. Check-pointing is associated with
a) log based recovery
b) non-log based recovery
c) both (a) and (b)
d) none of these

Answer: (a)

[WBUT 2015]

10. Which of the following guarantees that, : A transaction is either performed in its entirety or not performed at all"?
a) consistency
b) durability
c) isolation
d) atomicity

Answer: (d)

[WBUT 2016]

11. Serializability of concurrent transaction is ensured by
a) locking
b) time stamping
c) both (a) and (b)
d) none of these

Answer: (c)

[WBUT 2017]

12. Serializability of concurrent transactions are ensured by
a) Locking
b) Drop
c) both of these
d) none of these

Answer: (a)

[WBUT 2017]

13. Which index is specified on the non-ordering fields of a file?
a) Primary
b) Clustering
c) Secondary
d) none of these

Answer: (c)

[WBUT 2018]

14. Which of the following is not a property of transaction?
a) atomicity
b) concurrency
c) isolation
d) durability

Answer: (b)

[WBUT 2018]

15. Conflict serializability can be detected by
a) WFG
b) precedence graph
c) spanning tree
d) none of these

Answer: (b)

[WBUT 2018]

16. Check points are part of
a) recovery measure
b) security measure
c) concurrency measure
d) authorization measure

Answer: (a)

[WBUT 2019]

17. Which of the following protocols ensures conflict serializability and safety from deadlocks?
a) Two-phase locking protocol
b) Time stamp ordering protocol
c) Graph based protocol
d) None of these

Answer: (b)

Short Answer Type Questions

1. What is two phase locking protocol? How does it guarantee serializability? [WBUT 2008, 2012, 2017] [WBUT 2008, 2017]

OR,

Briefly explain two phase locking protocol. [WBUT 2015]

OR,

Describe different 2PL protocol in brief. [WBUT 2018]

Answer:

Two-phase locking

According to the **two-phase locking** protocol, a transaction handles its locks in two distinct, consecutive phases during the transaction's execution:

1. **Expanding phase** (aka Growing phase): locks are acquired and no locks are released (the number of locks can only increase).
2. **Shrinking phase:** locks are released and no locks are acquired.

The serializability property is guaranteed for a schedule with transactions that obey the protocol. The 2PL *schedule class* is defined as the class of all the schedules comprising transactions with data access orders that could be generated by the 2PL protocol (or in other words, all the schedules that the 2PL protocol can generate).

Typically, without explicit knowledge in a transaction on end of phase-1, it is safely determined only when a transaction has entered its *ready* state in all its processes (processing has ended, and it is ready to be committed; no additional data access and locking are needed and can happen). In this case phase-2 can end immediately (no additional processing is needed), and actually no phase-2 is needed. Also, if several processes (two or more) are involved, then a synchronization point (similar to atomic commitment) among them is needed to determine end of phase-1 for all of them (i.e., in the entire distributed transaction), to start releasing locks in phase-2 (otherwise it is very likely that both 2PL and Serializability are quickly violated). Such synchronization point is usually too costly (involving a distributed protocol similar to atomic commitment), and end of phase-1 is usually postponed to be merged with transaction end (atomic commitment protocol for a multi-process transaction), and again phase-2 is not needed. This turns 2PL to SS2PL (see below). All known implementations of 2PL in products are SS2PL based, and whenever 2PL (or Strict 2PL, S2PL) practical utilization has been mentioned in the professional literature, the intention has been SS2PL.

2. Describe ACID properties in DBMS. [WBUT 2009]

OR,

What is ACID property? [WBUT 2013, 2019]

OR,

Discuss the ACID properties of a Database transaction. [WBUT 2015]

OR,

Explain the ACID properties of transactions. [WBUT 2016]

OR,

Discuss the ACID properties of transactions. [WBUT 2017]

Answer: *Refer to Question No. 1 of Long Answer Type Questions.*

3. With suitable examples, show how recovery in a database system can be done using LOG files with –
a) immediate updation
b) deferred updation. [WBUT 2010]

Answer:

a) Immediate Update

Immediate update, or UNDO/REDO, is another algorithm to support ABORT and machine failure scenarios.

- While a transaction runs, changes made by that transaction can be written to the database at any time. However, the original and the new data being written must both be stored in the log BEFORE storing it on the database disk.
- On a commit:
- All the updates which has not yet been recorded on the disk is first stored in the log file and then flushed to disk.
- The new data is then recorded in the database itself.
- On an abort, REDO all the changes which that transaction has made to the database disk using the log entries.
- On a system restart after a failure, REDO committed changes from log.

Example

Using immediate update, and the transaction TRAN1 again, the process is:

Time	Action	LOG
t1	START	-
t2	read(A)	-
t3	write(10,B)	Was B == 6, now 10
t4	write(20,C)	Was C == 2, now 20
t5	COMMIT	COMMIT

b) *Deferred Update*

Deferred update, or NO-UNDO/REDO, is an algorithm to support ABORT and machine failure scenarios.

- While a transaction runs, no changes made by that transaction are recorded in the database.

On a commit:

- The new data is recorded in a log file and flushed to disk
- The new data is then recorded in the database itself.
- On an abort, do nothing (the database has not been changed).
- On a system restart after a failure, REDO the log.

Example

Consider the following transaction TRAN1

Transaction TRAN1
read(A)
write(10,B)
write(20,C)
Commit

Using deferred update, the process is:

Time	Action	Log
t1	START	-
t2	read(A)	-
t3	write(10,B)	B = 10
t4	write(20,C)	C = 20
t5	COMMIT	COMMIT

	Before			After		
DISK			B = 6			B = 10
	A = 5	C = 2		A = 5	C = 20	

If the DMBS fails and is restarted:

- The disks are physically or logically damaged then recovery from the log is impossible and instead a restore from a dump is needed.
- If the disks are OK then the database consistency must be maintained. Writes to the disk which was in progress at the time of the failure may have only been partially done.
- Parse the log file, and where a transaction has been ended with `COMMIT' apply the data part of the log to the database.
- If a log entry for a transaction ends with anything other than COMMIT, do nothing for that transaction.
- flush the data to the disk, and then truncate the log to zero.
- the process or reapplying transaction from the log is sometimes referred to as 'rollforward'.

4. Explain deferred update technique for recovery with its advantages and disadvantages. **[WBUT 2011]**

Answer:

Recovery Techniques Use the Following Operations

UNDO –> similar to ROLLBACK but on a single operation.

REDO –> a single operation is redone.

Consider the following example transaction from before:
Read (Sno = x, Salary)
Salary = Salary * 1.1
Write (Sno = x, Salary)
For the Read operation, the DBMS performs the following steps:
- Find the address of the disk block containing the record with key value x.
- Transfer this disk block into a database buffer in main memory.
- Copy the salary data from the database buffer into the variable Salary.

For the Write operation, the following steps occur:
- Find the address of the disk block containing the record with key value x.
- Transfer this disk block into a database buffer in main memory.
- Copy the salary data from the variable Salary into the database buffer.
- Write the database buffer back to disk.

The database buffers function as an intermediate storage location (or cache) to minimize the cost of reading and writing data on disk. So, it is common to delay writing them to disk until absolutely necessary. It is only once the buffers have been **flushed** to disk that any update operations can be regarded as permanent.
The flushing of buffers can be triggered by special commands (e.g., transaction commit), or automatically when the buffer becomes full. Explicit flushing of the buffers is called **force-writing**. Failure may occur between writing to the buffers and flushing the buffers to disk. In this case, the recovery manager must determine the status of the transaction that performed the write at the time of failure. If the transaction that wrote prior to failure had issued its commit, then the recovery manager must **redo** the transaction's updates to ensure.

5. What is the single most significant difference between two-phase locking and the time-stamping technique concurrency control? Explain briefly. [WBUT 2014]

OR,

Explain two-phase locking protocol. [WBUT 2016]

OR,

What is the main difference between two-phase locking and the time stamping technique concurrency control? Explain briefly. [WBUT 2019]

Answer:

Two Phase Locking - 2PL

This locking protocol divides transaction execution phase into three parts. In the first part, when transaction starts executing, transaction look for grant of locks. Second part is where the transaction acquires all locks and no other lock is required. Transaction keeps executing its operation. As soon as the transaction releases its first lock,

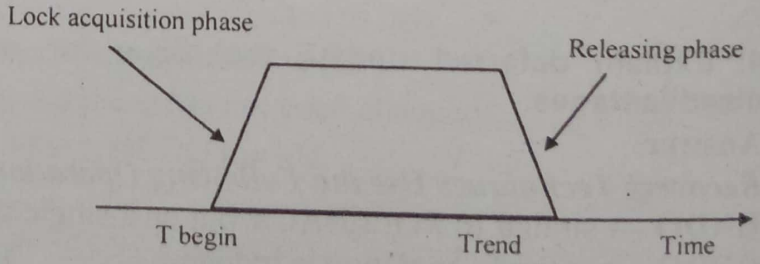


Fig: Two Phase Locking

the third phase starts. In this phase a transaction cannot demand for any lock but only releases the acquired locks.
Two phase locking has two phases, one is growing; where all locks are being acquired by transaction and second one is shrinking, where locks held by the transaction are being released. To claim an exclusive (write) lock, a transaction must first acquire a shared (read) lock and then upgrade it to exclusive lock.

Time stamp based protocols
The most commonly used concurrency protocol is time-stamp based protocol. This protocol uses either system time or logical counter to be used as a time-stamp.
Every transaction has a time-stamp associated with it and the ordering is determined by the age of the transaction. A transaction created at 0004 clock time would be older than all other transaction, which come after it. For example, any transaction 'y' entering the system at 0006 is two seconds younger and priority may be given to the older one.
The major difference between 2PL and Time stamp is that in lock based protocols manage the order between conflicting pairs among transaction at the time of execution whereas time-stamp based protocols start working as soon as transaction is created.
Besides, every data item is given the latest read and write-timestamp. This lets the system know, when last read was and write operation made on the data item.

6. What do you mean by transaction? Explain the transaction states. [WBUT 2016]

Answer:
1st Part: ***Refer to Question No. 8(a) of Long Answer Type Questions.***
2nd Part: ***Refer to Question No. 13(c) of Long Answer Type Questions.***

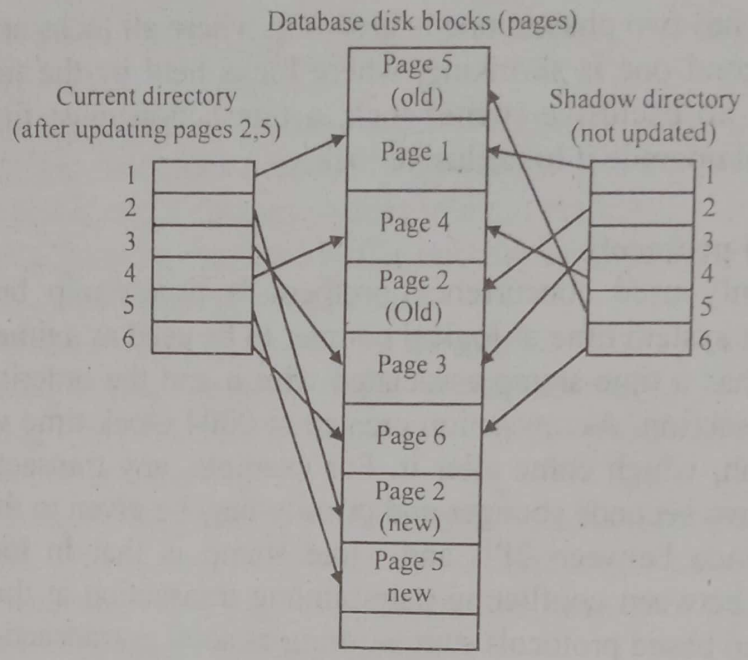
7. Explain log based recovery and checkpoints. [WBUT 2016]

Answer:
Log based recovery: In this method, log of each transaction is maintained in some stable storage, so that in case of any failure, it can be recovered from there to recover the database. But storing the logs should be done before applying the actual transaction on the database.
Every log in this case will have informations like what transaction is being executed, which values have been modified to which value, and state of the transaction. All these log information will be stored in the order of execution.

Checkpoints: Checkpoint acts like a bookmark. During the execution of transaction, such checkpoints are marked and transaction is executed. The log files will be created as usual with the steps of transactions. When it reaches the checkpoint, the transaction will be updated into database and all the logs till that point will be removed from file. Log files then are updated with new steps of transaction till next checkpoint and so on. Here care should be taken to create a checkpoint because, if any checkpoints are created before any transaction is complete fully, and data is updated to database, it will not meet the purpose of the log file and checkpoint. If checkpoints are created when each transaction is complete or where the database is at consistent state, then it will be useful.

8. What do you mean by shadow paging? **[WBUT 2016]**

Answer:



An example of shadow paging

- Shadow paging is an alternative to log-based recovery; this scheme is useful if transactions execute serially
- Idea: maintain *two* page tables during the lifetime of a transaction –the current page table, and the shadow page table
- Store the shadow page table in nonvolatile storage, such that state of the database prior to transaction execution may be recovered.
- Is a State-based approach
- Save a checkpoint of the state before performing transaction operations
- If transaction aborts, restore old state
- Can be expensive if state is large
- Shadow paging limits the cost
- Maintain two page tables during life of a transaction: *current* page and *shadow* page table.

When transaction starts, two pages are the same. Shadow page table never changed thereafter and used to restore database in event of failure. During T, current page table records all updates to database. When transaction completes, current page table becomes shadow page table.

- To start with, both the page tables are identical. Only current page table is used for data item accesses during execution of the transaction.
- Whenever any page is about to be written for the first time
- A copy of this page is made onto an unused page.
- The current page table is then made to point to the copy
- The update is performed on the copy

9. What do you mean by deadlock handling? Explain in detail. **[WBUT 2016]**

Answer:

A set of transactions are considered to be in a deadlock state, if the transactions are waiting for one another to release the data items needed for them that are held by others. In a deadlock state no transaction will proceed.

The deadlock can be handled by rolling back a transaction which would be chosen as the victim.

Deadlock can be handled in the following ways:

Deadlock Prevention: This concept ensures that the system never enters a deadlock state. It chooses the transaction which would probably cause the deadlock and rolls-back the transaction.

Deadlock Detection: This identifies the deadlock if any happened and recovers the system from deadlock.

Deadlock Recovery: recovers the system from deadlock state. It chooses the identified transaction which caused the deadlock and rolls-back it.

10. Consider the following two transactions: **[WBUT 2018]**

T1 : read (A); read (B); if A = 0, then B = B + 1; write (B);

T2 : read (B); read (A); if B = 0, then A = A + 1; write (A);

Add lock and unlock instructions to transaction T1 and T2. Is there any dead lock observed?

Answer:

Lock and unlock instructions:

lock-S(A)	
read(A)	
lock-X(B)	
read(B)	
if A = 0, then B = B + 1	
write(B)	
unlock(A)	
unlock(B)	
	lock-S(B)
	read(B)
	read(B)
	read(A)
	if B = 0, then A = A + 1; write (A);
	read(A)
	unlock(B)
	unlock(A)

If this schedule is followed, the transactions are in exclusion and hence there is no question of dead lock.

11. Define schedule and conflict serializable schedule. **[WBUT 2019]**

Answer:
1st Part: *Refer to Question No. 2(a) of Long Answer Type Questions.*
2nd Part: *Refer to Question No. 8.d) of Long Answer Type Questions.*

Long Answer Type Questions

1. Discuss the ACID properties of a database transaction. [WBUT 2007]

Answer:

Properties of Transactions

With the help of the abbreviated word ACID the properties of a transaction can be explained as follows:

1. ***Atomicity:*** **A transaction is either performed in its entirety or not performed at all.** The system guarantees this using logging, shadowing, distributed commit.
2. ***Consistency (Preservation):*** The complete execution of a transaction takes the database from one consistent state to another. The application guarantees this by correctly marking transaction boundaries.
3. ***Isolation:*** The execution of a transaction is not interfered with by any other (concurrent) transactions. It should not make its update values visible to other transactions until it is completed successfully. This is guaranteed through locking or timestamp-based concurrency control.
4. ***Durability (Permanency):*** Changes applied to the database by a committed transaction must persist in the database even if the system crashes before all changes are reflected by writing updates to stable storage.

2. a) What is a schedule? [WBUT 2007]

Answer:

Schedule

A schedule S of n transactions T1, T2, T3,……., Tn is an ordering of operations of the transactions ,subject to the constraints that for each transaction T1 that participates in S, the operation of T1 in S must appear in the same order in which they occur in T1.

For example;

Let there be three transactions T1, T2 and T3

T1 : R1(x) | R1(y) | W1(x) |W1(y)|

T2 : R2(x) | R2(y) | W2(x) |W2(y)|

T3 : R3(x) | R3(z) | W3(x)|W3(z)| R3(x)| W3(y)

Schedules will be

S1: R1(x) , R1(y) , W1(x) ,W1(y), R2(x) , R2(y) , W2(x) ,W2(y), R3(x) , R3(z) , W3(x),W3(z), R3(x), W3(y)

or

S1:{T1},{T2}, {T3}

S2: R1(x) ,R2(x), R3(x), R1(y) ,R2(y), R3(z), W1(x), W2(x) W3(x), W1(y), W2(y),

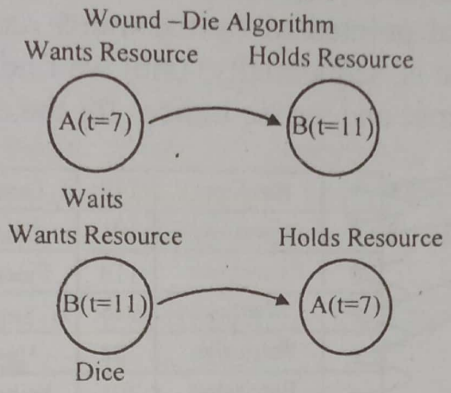
W3(z), R3(x), W3(y)
S3: R1(x) , R1(y), W2(x) , W2(y), R2(x) ,
R2(y) , W1(x) , W1(y), R3(x), W3(y) ,
R3(x) , R3(z) , W3(x), W3(z)
Here S1 and S2 are schedules but not S3('___ ' indicates the non schedule sequence)

b) Describe the wait-die and wound-wait protocols for deadlock prevention.
[WBUT 2007, 2010, 2011, 2013, 2019]

Answer:

1. The ***Wait-Die*** algorithm:

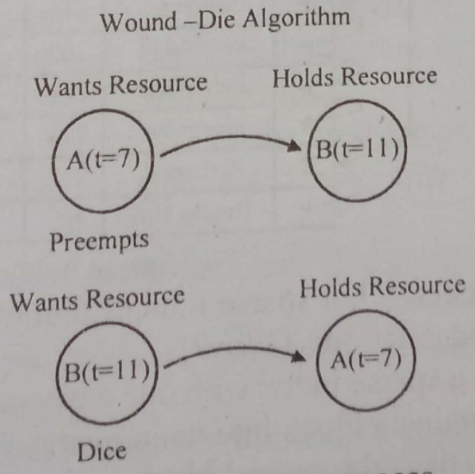
Allow wait only if waiting process is older.
Since timestamps increase in any chain of waiting processes, cycles are impossible.



The Wait-Die algorithm kills the younger process.
When the younger process restarts and requests the resource again, it may be killed once more.
This is the less efficient of these two algorithms.

2. The ***Wound-Wait*** algorithm:

Otherwise allow wait only if waiting process is younger.
Here timestamps decrease in any chain of waiting process, so cycles are again impossible.
It is wiser to give older processes priority.



The Wound-Wait algorithm preempts the younger process.
When the younger process re-requests resource, it has to wait for older process to finish.
This is the better of the two algorithms.

3. Write the concepts of dense index and sparse index with example. When is it preferable to use a dense index rather than a sparse index? Explain.

[WBUT 2008, 2010]

Answer:

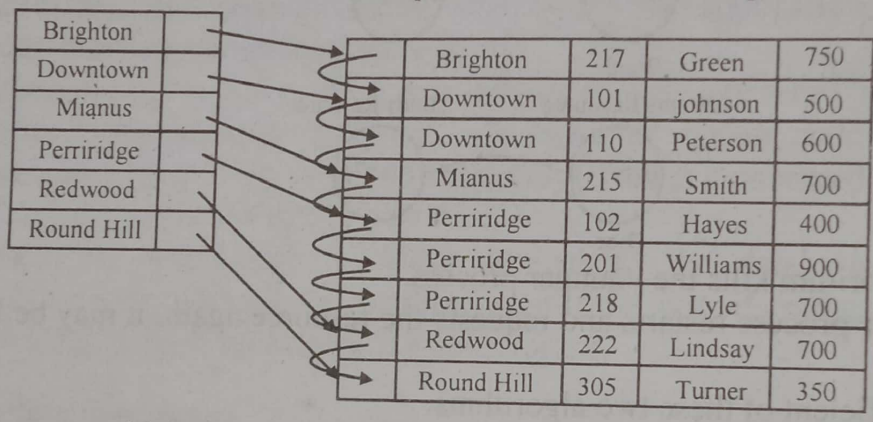
Dense Index:

- An index record appears for **every** search key value in file.
- This record contains search key value and a pointer to the actual record.

Sparse Index:

- Index records are created only for **some** of the records.
- To locate a record, we find the index record with the largest search key value less than or equal to the search key value we are looking for.
- We start at that record pointed to by the index record, and proceed along the pointers in the file (that is, sequentially) until we find the desired record.

The following figures show dense and sparse indices for the deposit file.



Notice how we would find records for Perryridge branch using both methods.

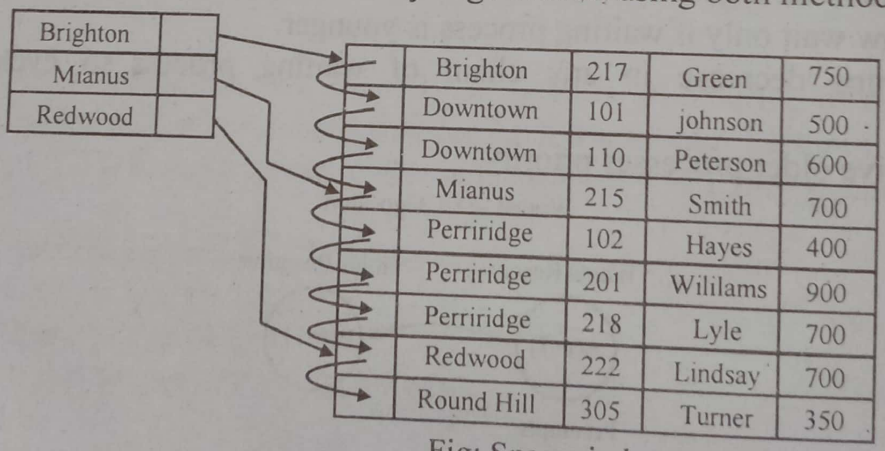


Fig: Sparse index

Dense indices are faster in general, but sparse indices require less space and impose less maintenance for insertions and deletions. (Why?)

A good compromise: to have a sparse index with one entry per block.

- Biggest cost is in bringing a block into main memory.
- We are guaranteed to have the correct block with this method, unless record is on an overflow block (actually could be **several** blocks).
- Index size still small.

4. a) Distinguish between locking and timestamp protocols for concurrency controls. Explain multi-version two-phase locking. [WBUT 2010]

b) Define three concurrency problems, dirty read, non-repeatable read, phantoms.

c) Consider the following *two* transactions: [WBUT 2010, 2011, 2013, 2019]

T_1 : read (A) ;
read (B) ;
if $A = 0$, then $B : B + 1$;
write (B)

T_2 : read (B) ;
read (A) ;
If $B = 0$, then $A : A + 1$;
Write (A)

Add lock and unlock instructions to transactions T_1 and T_2, so that they observe the two-phase locking protocol. Can the execution of these transactions result in a deadlock? [WBUT 2010]

Answer:

a) Locking data items to prevent multiple transactions from accessing the items concurrently is the most common method used to ensure serializability. In this section, we examine the basic idea behind the locking concepts and introduce two-phase locking protocol.

A lock is a variable associated with data items that describe the status of the item with respect to possible operations that can be applied to it.

There are various types of locks are used in concurrency control like **Binary Locks and Shared/Exclusive Locks**. In the locking protocols, the order between every pair of conflicting transactions is determined at execution time by the first lock that they both request that involves incompatible modes.

Another method for determining the serializability order is select an ordering among transactions in advance. Timestamp – ordering scheme is the typical example of this method.

A timestamps is a unique variable associated with a transaction. Timestamp of transaction T is denoted by TS(T). This timestamp is assigned by the database system in the order in which the transaction submitted to the system. If Ti has been assigned timestamp TS(Ti) and a new transaction Tj enters the system then TS(Ti) < TS(Tj)

There are two simple ways for generating timestamp:

1. Use the current value of system clock as the timestamps
2. Use a logical counter that is incremented each time a value is assigned to a transaction as timestamp.

In order to use timestamp techniques, each data item Q is associtated with two timestamp values:

- read_TS(Q). This is the largest timestamp of any transaction that have successfully read data item Q.

read_TS(Q) = TS(T) where T is the youngest transaction that has read Q successfully

- write_TS(Q). This is the largest timestamp of any transaction that sucessfully written data item Q.

write_TS(Q) = TS(T) where T is the youngest transaction that has written Q successfully

Multi-version two-phase locking

Introduction to Multi-Version Concurrency

The aim of Multi-Version Concurrency is to avoid the problem of Writers blocking Readers and vice-versa, by making use of multiple versions of data.

The problem of Writers blocking Readers can be avoided if Readers can obtain access to a previous version of the data that is locked by Writers for modification.

The problem of Readers blocking Writers can be avoided by ensuring that Readers do not obtain locks on data. Multi-Version Concurrency allows Readers to operate without acquiring any locks, by taking advantage of the fact that if a Writer has updated a particular record, its prior version can be used by the Reader without waiting for the Writer to Commit or Abort. In a Multi-version Concurrency solution, Readers do not block Writers, and vice versa.

Requirements of Multi-Version Concurrency systems

As its name implies, multi-version concurrency relies upon multiple versions of data to achieve higher levels of concurrency. Typically, a DBMS ordering multi-version concurrency (MVDB), needs to provide the following features:

1. The DBMS must be able to retrieve older versions of a row.
2. The DBMS must have a mechanism to determine which version of a row is valid in the context of a transaction. Usually, the DBMS will only consider a version that was committed prior to the start of the transaction that is running the query. In order to determine this, the DBMS must know which transaction created a particular version of a row, and whether this transaction committed prior to the starting of the current transaction.

b) Dirty read—Dirty reads occur when one transaction reads data that has been written but not yet committed by another transaction. If the changes are later rolled back, the data obtained by the first transaction will be invalid.

Nonrepeatable read—Nonrepeatable reads happen when a transaction performs the same query two or more times and each time the data is different. This is usually due to another concurrent transaction updating the data between the queries.

Phantom reads—Phantom reads are similar to nonrepeatable reads. These occur when a transaction (T1) reads several rows, and then a concurrent transaction (T2) inserts rows. Upon subsequent queries, the first transaction (T1) finds additional rows that were not there before.

c)

T31: **lock-S**(A)
read(A)
lock-X(B)
read(B)
if A=0

```
        then B:=B+1
        write(B)
        unlock(A)
        unlock(B)

T32:    lock-S(B)
        read(B)
        lock-X(A)
        read(A)
        if B=0
        then A:=A+1
        write(A)
        unlock(B)
        unlock(A)
```

Execution of these transactions can result in deadlock. For example, consider the following partial schedule:

T31	T32
lock-S(A)	lock-S(B)
read(A)	read(B)
lock-X(B)	lock-X(A)

The transactions are now deadlocked.

5. What are the roles of the Analysis, Redo and Undo phases in the recovery algorithm 'ARIES'? **[WBUT 2010, 2011, 2013, 2014, 2019]**

Answer:

The ARIES recovery procedure consists of three main steps:

- **Analysis:** It identifies the dirty (updated) pages in the buffer and the set of transactions active at the time of crash. The appropriate point in the log where REDO operation should start is also determined.
- **REDO phase:** It actually reapplies updates from the log to the database. Generally the REDO operation is applied to only committed transactions. However, in ARIES, this is not the case. Certain information in the ARIES log will provide the start point for REDO, from which REDO operations are applied until the end of the log is reached. Thus only the necessary REDO operations are applied during recovery.
- **UNDO phase:** The log is scanned backwards and the operations of transactions that were active at the time of the crash are undone in reverse order. The information needed for ARIES to accomplish its recovery procedure includes the log, the transaction table, and the dirty page table. In addition, check pointing is used.

6. a) State Two-phase commit protocol and discuss the implications of a failure on the part of
i) the coordinator
ii) a participant, during each of the two phases. **[WBUT 2011, 2013, 2019]**

Answer:
The Two-Phase Commit Protocol (2PC) is a distributed algorithm used in computer networks and distributed database systems. It is used when a simultaneous data update should be applied within a distributed database. In this protocol, one node acts as the coordinator, which is also called master and all the other nodes in the network are called participants.
Its 1st phase, all these participants agree or disagree with the coordinator to commit, i.e., vote yes's or no's and
In 2nd phase they complete the transaction simultaneously by getting the commit or the abort signal from the coordinator.
i) If the Coordinator fails in the midst of execution of a transaction T, then the participating site must decide the fate of T.
The Two-Phase Commit Protocol goes to a blocking state by the failure of the coordinator.

ii) When a participating site, S_k recovers from a failure, it examines its log to determine the fate of those transactions T that were in the midst of execution during failure.

1st phase:

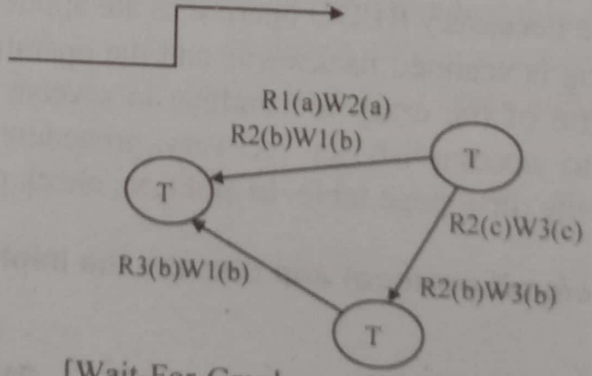
- If the log contains <Ready T> record, the site must consult C_i to determine the fate of T:
- If the log contains <Abort T> record, the site executes Undo(T)

2nd phase:

- If the log contains <Commit T> record, the site executes Redo(T)
- If the log contains <Abort T> record, the site executes Undo(T)

b) Let T_1, T_2 and T_3 be transactions that operate on the same data items A, B and C. Let r_1 (A) mean that T_1 reads A, w_1 (A) means that T_1 writes A and so on for T_2 and T_3. Consider the following schedule
S1; $r_2(C)$, $r_2(B)$, $w_2(B)$, $r_3(B)$, $r_3(C)$, $r_1(A)$, $w_1(A)$, $w_3(B)$, $w_3(C)$, $r_2(A)$, $r_1(B)$, $w_1(B)$, $w_2(A)$
Is the schedule serializable? **[WBUT 2011, 2013, 2018, 2019]**

Answer:
From the given schedule, using the read-before-you write scheme the wait for graph is drawn.



[Wait-For-Graph over T1, T2, T3]

The graph forms a cycle between the nodes T1 and T2 and hence the schedule is not conflict seralizable.

7. What do you mean by integrity constraint? [WBUT 2013, 2019]

Answer:

An integrity constraint defines a business rule for a table column. When enabled, the rule will be enforced by oracle (and so will always be true.) To create an integrity constraint all existing table data must satisfy the constraint.

Default values are also subject to integrity constraint checking (defaults are included as part of an INSERT statement before the statement is parsed.)

If the results of an INSERT or UPDATE statement violate an integrity constraint, the statement will be rolled back.

Integrity constraints are stored as part of the table definition, (in the data dictionary.)

If multiple applications access the same table they will all adhere to the same rule.

8. a) What is transaction? [WBUT 2013, 2019]

Answer:

A transaction is a set of changes that must all be made together. Transaction is executed as a single unit. If the database was in consistent state before a transaction, then after execution of the transaction also, the database must be in a consisted.

b) Explain with example serial and serializable schedule. [WBUT 2013, 2019]

Answer:

A schedule S is said to be a serial if for every participating transaction T in the schedule, executes consecutively. Here in the example S1 is a serial schedule but not S2. A schedule S of n transactions is serializable, if is equivalent to some serial schedule of the same n transactions.

c) What are the problems of concurrent execution of transaction? [WBUT 2013, 2019]

Answer:

If locking is not available and several users access a database concurrently, problems may occur if their transactions use the same data at the same time. Concurrency problems include:

- Lost or buried updates.
- Uncommitted dependency (dirty read).
- Inconsistent analysis (non-repeatable read).
- Phantom reads

Lost Updates

Lost updates occur when two or more transactions select the same row and then update the row based on the value originally selected. Each transaction is unaware of other transactions. The last update overwrites updates made by the other transactions, which results in lost data.

For example, two editors make an electronic copy of the same document. Each editor changes the copy independently and then saves the changed copy, thereby overwriting the original document. The editor who saves the changed copy last overwrites changes made by the first editor. This problem could be avoided if the second editor could not make changes until the first editor had finished.

Uncommitted Dependency (Dirty Read)

Uncommitted dependency occurs when a second transaction selects a row that is being updated by another transaction. The second transaction is reading data that has not been committed yet and may be changed by the transaction updating the row.

For example, an editor is making changes to an electronic document. During the changes, a second editor takes a copy of the document that includes all the changes made so far, and distributes the document to the intended audience. The first editor then decides the changes made so far are wrong and removes the edits and saves the document. The distributed document contains edits that no longer exist, and should be treated as if they never existed. This problem could be avoided if no one could read the changed document until the first editor determined that the changes were final.

Inconsistent Analysis (Non-repeatable Read)

Inconsistent analysis occurs when a second transaction accesses the same row several times and reads different data each time. Inconsistent analysis is similar to uncommitted dependency in that another transaction is changing the data that a second transaction is reading. However, in inconsistent analysis, the data read by the second transaction was committed by the transaction that made the change. Also, inconsistent analysis involves multiple reads (two or more) of the same row and each time the information is changed by another transaction; thus, the term non-repeatable read.

For example, an editor reads the same document twice, but between each reading, the writer rewrites the document. When the editor reads the document for the second time, it has changed. The original read was not repeatable. This problem could be avoided if the editor could read the document only after the writer has finished writing it.

Phantom Reads

Phantom reads occur when an insert or delete action is performed against a row that belongs to a range of rows being read by a transaction. The transaction's first read of the range of rows shows a row that no longer exists in the second or succeeding read, as a result of a deletion by a different transaction. Similarly, as the result of an insert by a different transaction, the transaction's second or succeeding read shows a row that did not exist in the original read.

For example, an editor makes changes to a document submitted by a writer, but when the changes are incorporated into the master copy of the document by the production department, they find that new unedited material has been added to the document by the author. This problem could be avoided if no one could add new material to the document until the editor and production department finish working with the original document.

d) Explain with the help of precedence graph the conflict and non-conflict serializability.

[WBUT 2013, 2019]

Answer:

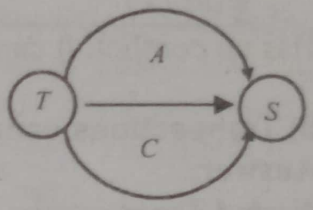
Let S be a schedule. We construct a directed graph, called a precedence graph from S. This graph consists of a pair G = (V,E), where V is a set of vertices, and E is a set of edges. The set of vertices consists of all the transactions participating in the schedule. The set of edges consists of all edges Ti → Tj for which one of the following three conditions holds

1. Ti executes write (Q) before Tj executes read (Q).
2. Ti executes read (Q) before Tj executes write (Q).
3. Ti executes write (Q) before Tj executes write (Q).

If an edge Ti → Tj exists in the precedence graph, then, in any serial schedule S' equivalent to S, Ti must appear before Tj.

To determine if a schedule is serializable, make a directed graph:

- Add a node for each committed transaction.
- Add an arc from T to S if any equivalent serial schedule must order T before S. T must commit before S iff the schedule orders some operation of T before some operation of S. The schedule only defines such an order for conflicting operations so this means that a pair of accesses from T and S conflict over some item x, and the schedule says T "wins" the race to x.



The schedule is conflict-serializable if the graph has no cycles (winner take all).

The Graph Test Example:

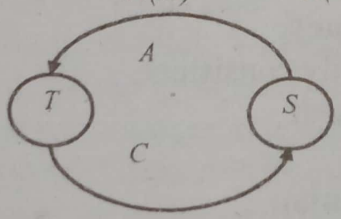
Consider two transactions T and S:

T: transfer Rs.100 from A to C: R(A) W(A) R(C) W(C)

S: compute total balance for A and C: R(A) R(C)

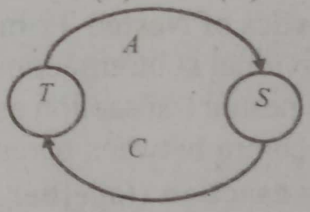
T: *R(A)* *W(A)* *R(C)* *W(C)*

S: *R(A)* *R(C)*



T: *R(A)* *W(A)* *R(C)* *W(C)*

S: *R(A)* *R(C)*

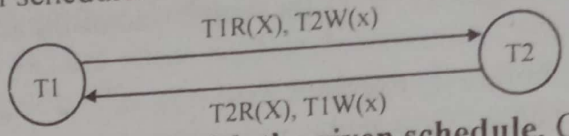


9. a) Consider the following transactions and find how many different schedules of the two transactions are possible? How many of these are serializable?

Transaction 1	Traction 2
Read (X)	Read (X)
X:=X–N	X:=X+M
Write (X)	Write (X)
Read (Y)	------
Y:=Y+N	------
Write (Y)	------

[WBUT 2014]

Answer:
A **serial schedule** is when all the operations of one transactions appear together (not mixed with the operations of any other transactions on the schedule).
A **serializable schedule** is a weaker term -- it is a schedule where the operations of different transactions may be mixed together on the schedule, so long as they are conflict-equivalent to some serial schedule.



There is a loop between T1 and T2 with the given schedule. Conflict will occur.
Schedule can execute in exclusive mode to become serializable.
However,

Read (Y)
Y:=Y+N
Write (Y)

Has no conflict. It can be a serial schedule.

b) Transactions cannot be nested inside one another. Why not? **[WBUT 2014]**

Answer:
Nested Transactions
Problem: Lack of mechanisms that allow:
- A top-down, functional decomposition of a transaction into sub transactions.
- Individual sub transactions to abort without aborting the entire transaction.
- Although a nested transaction looks similar to a distributed transaction, it is not conceived of as a tool for accessing a multi database.

Characteristics of Nested Transactions Parent:
- creates children to perform subtasks
- either sequentially or concurrently waits until all children complete

Characteristics of Nested Transactions Consistency:
- Individual subtransactions are not necessarily consistent,
- But nested transaction as a whole is consistent
- No comm between parent and children.

Each subtransaction (together with its descendants):
- is isolated w.r.t. each sibling (and its descendants).
- hence, siblings are serializable,
- but order is not determined i.e. NT is *non-deterministic*.

Concurrent nested transactions are serializable.
A sub transaction is atomic:
- It can abort or commit independently of other sub transactions.
- Commit is conditional on commit of parent (since child task is a subtask of parent task).
- Abort causes abort of all sub transaction's children.

Nested transaction commits when root commits:
- At that point updates of committed sub transactions are made durable.

c) Systems do not allow transaction to commit changes to database on an individual basis. i.e., without simultaneously committing changes to all other database. Why not?
[WBUT 2014]

Answer:

It doesn't allow a given transaction to commit changes to files on an individual basis because it must make sure that the system remains in a consistent state before and after a transaction. So it has to make sure that the system as a whole is reliable before making changes on an individual basis.

10. a) Explain the purpose of the checkpoint mechanism. How often should checkpoints be performed?
b) How does 'strict two-phase' and 'consecutive two-phase' locking protocol differ.
[WBUT 2015]

Answer:

a) Checkpointing is done with log-based recovery schemes to reduce the amount of searching that needs to be done after a crash. If there is no checkpointing, then the entire log must be searched after a crash, and all transactions "redone" from the log. If checkpointing is used, then most of the log can be discarded. Since checkpoints are very expensive, how often they should be taken depends upon how reliable the system is. The more reliable the system, the less often a checkpoint should be taken.

b) According to the two-phase locking protocol a transaction handles its locks in two distinct, consecutive phases during the transaction's execution:

- Expanding phase (aka Growing phase): locks are acquired and no locks are released (the number of locks can only increase).
- Shrinking phase: locks are released and no locks are acquired.

The two phase locking (2PL) rule can be summarized as: never acquire a lock after a lock has been released. The serializability property is guaranteed for a schedule with transactions that obey this rule.

Typically, without explicit knowledge in a transaction on end of phase-1, it is safely determined only when a transaction has completed processing and requested commit. In this case all the locks can be released at once (phase-2).

To comply with the Strict two phase locking protocol a transaction needs to comply with 2PL, and release its write (exclusive) locks only after it has ended, i.e., being either committed or aborted. On the other hand, read (shared) locks are released regularly during phase 2. This protocol is not appropriate in B-trees because it causes Bottleneck (while B-trees always starts searching from the parent root).

11. a) What are the various states of a transaction? Explain with a state diagram.
b) Consider the following schedule:
S1: r2 (C), r2 (B), w2 (B), r3 (B), r3 (C), r1 (A), w1 (A), w3 (B), w3 (C), r2 (A), r1 (B), w1 (B), w2(A).
Is the schedule serializable?
c) What is cascadeless schedule? Why is cascadeless of schedule desired?
d) Explain log based recovery.
[WBUT 2017]

Answer:

a) ***Refer to Question 13(c) of Long Answer Type Questions.***

b) The serialization graph is drawn by following the rule read before you write and is presented in the fig below:

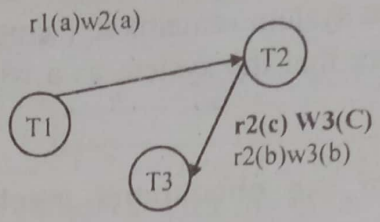


Fig: Serialization Graph

There are no directed cycles in this graph. Therefore, the given schedule S is conflict serializable.

c)

T10	T11	T12
read(A)		
read(B)		
write(A)		
	read(A)	
	write(A)	
		read(A)

Transaction T10 writes a value of A that is read by Transaction T11. Transaction T11 writes a value of A that is read by Transaction T12. Suppose at this point T10 fails. T10 must be rolled back, since T11 is dependent on T10, T11 must be rolled back, T12 is dependent on T11, T12 must be rolled back.

This phenomenon, in which a single transaction failure leads to a series of transaction rollbacks is called Cascading rollback. Thus cascade less schedule can be defined as:

If Tj reads value updated by Ti only after Ti is committed, the schedule will be cascade less recoverable.

A cascade less schedule is one in which failure of transaction results of which are to be used in other transactions, will not lead to necessity of rolling-back latter transactions. This is achieved by allowing to read results provided by some transaction, only after commitment of the latter. This restricts concurrency of execution of transactions, and if probability of transactions failure is small, it may be desirable to allow non-cascade less schedules.

d) ***Refer to Question No. 7 of Short Answer Type Questions.***

12. a) What is deadlock? What are necessary and sufficient condition for deadlock?
b) What are the necessary measurements for deadlock avoidance?
c) Let T_1, T_2 and T_3 be transactions that operate on the same data items A, B and C. Let $r_1(A)$ mean that T_1 read A, $W_1(A)$ means that T_1 writes A and so on for T_2 and T_3. Consider the following schedule:
$S_1 : r_2(C), r_2(B), W_2(B), r_3(B), r_3(C), r_1(A) W_1 : W_3(B), W_3(C), r_2(A), r_1(B), W_1(B), W_2(A)$ Is the schedule serializable? **[WBUT 2018]**

Answer:

a) 1st Part: *Refer to Question No. 9 of Short Answer Type Questions.*

2nd Part:

Four Necessary and Sufficient Conditions for Deadlock

1. Mutual exclusion

The resources involved must be unshareable; otherwise, the processes would not be prevented from using the resource when necessary.

2. Hold and wait or partial allocation

The processes must hold the resources they have already been allocated while waiting for other (requested) resources. If the process had to release its resources when a new resource or resources were requested, deadlock could not occur because the process would not prevent others from using resources that it controlled.

3. No pre-emption

The processes must not have resources taken away while that resource is being used. Otherwise, deadlock could not occur since the operating system could simply take enough resources from running processes to enable any process to finish.

4. Resource waiting or circular wait

A circular chain of processes, with each process holding resources which are currently being requested by the next process in the chain, cannot exist. If it does, the cycle theorem (which states that "a cycle in the resource graph is necessary for deadlock to occur") indicated that deadlock could occur.

b) Deadlock Avoidance

- It requires that each process declare the maximum number of resources of each type that it will need.
- The deadlock-avoidance algorithm dynamically checks the resource-allocation state to ensure the system can never be in a circular-wait condition.
- When a process requests a resource, the system must make sure that the allocation would leave the system in a safe state.
  1. [A state is ***safe*** if the system can allocate all resources requested by all processes (up to their stated maximums) without entering a deadlock state.

2. More formally, a state is safe if there exists a ***safe sequence*** of processes { P0, P1, P2, ..., PN } such that all of the resource requests for Pi can be granted using the resources currently allocated to Pi and all processes Pj where j < i. (I.e. if all the processes prior to Pi finish and free up their resources, then Pi will be able to finish also, using the resources that they have freed up.)
3. If a safe sequence does not exist, then the system is in an unsafe state, which ***MAY*** lead to deadlock. (All safe states are deadlock free, but not all unsafe states lead to deadlocks.)]

- It the system is in a safe state, then there would be no deadlock. However, if it is in an unsafe state, that there is a possibility of a deadlock.
- The avoidance approach requires that knowledge of all processes, all the resources available, the resources allocated presently and the future requests by the processes.
- For a single instance of a resource type we use the resource allocation graph.
- For multiple instance of a resource type, we use the banker's algorithm. [The Banker's Algorithm gets its name because it is a method that bankers could use to assure that when they lend out resources they will still be able to satisfy all their clients. (A banker won't loan out a little money to start building a house unless they are assured that they will later be able to loan out the rest of the money to finish the house.)]
- A major drawback of this method is that it is difficult to know at the beginning itself of the maximum resource required.

c) *Refer to Question No. 6.b) of Long Answer Type Questions.*

13. Write short notes of the following:

a) Two phase locking **[WBUT 2006, 2009]**

OR,

Two-phase locking protocol

b) Dead lock **[WBUT 2019]**

c) Transaction state diagram **[WBUT 2013, 2017, 2019]**

d) Time stamp based protocol for concurrency protocol **[WBUT 2013, 2019]**

e) Log based recovery **[WBUT 2017, 2018]**

f) Anomalies in concurrent execution of transaction **[WBUT 2018]**

Answer:

a) Two phase locking:

Two-phase locking is important in the context of ensuring that schedules are serializable. The **two-phase locking protocol** specifies a procedure each transaction follows. This protocol is important because, if observed by all transactions, it will guarantee a serializable, and thus correct, schedule. It may also help you understand why some methods of locking permit some types of inconsistencies.

The two-phase locking protocol

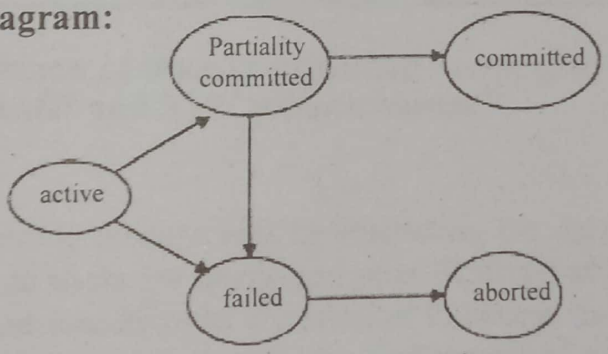
1. Before operating on any data item, a transaction must acquire a lock on that data item. This phase is called **growing phase.**

2. After releasing a lock, a transaction must never acquire any more locks. This phase is called **shrinking phase**.
In practice, a transaction normally holds locks until it terminates with either a COMMIT or ROLLBACK statement. Releasing locks before the end of the transaction disallows the operation of rolling back the changes whenever doing so would necessitate operating on rows to return them to an earlier state.

The two-phase locking theorem
If all transactions obey the two-phase locking protocol, then all possible interleaved schedules are serializable.

b) Dead lock:
Suppose that a transaction T1 places an exclusive lock on item A. Transaction T2 places an exclusive lock on item B. Transaction T1 attempts to place an exclusive lock on item B. Transaction T1 is placed into a wait state.
Transaction T2 attempts to place an exclusive lock on item A. Transaction T2 moves into a wait state. This is called as deadlock. Two main ways of dealing the dead locks are:
There are two main ways to deal with deadlock. They are:
Deadlock Prevention i.e. prevent deadlock in the first place by giving each transaction exclusive rights to acquire all locks needed before proceeding.
Deadlock Detection i.e., allow first the deadlock to occur, then break it by aborting one of the transactions.

c) Transaction state diagram:



States of transaction

A transaction must be in one of the following states:
Active: the initial state, the transaction stays in this state while it is executing.
Partially committed: after the final statement has been executed.
Failed: when the normal execution can no longer proceed.
Aborted: after the transaction has been rolled back and the database has been restored to its state prior to the start of the transaction.
Committed: after successful completion.

d) Time stamp based protocol for concurrency protocol:
Refer to Question No. 5 (2nd part) of Short Answer Type Questions.

e) Log based recovery:
Refer to Question No. 7 (1st part) of Short Answer Type Questions.

f) Anomalies in concurrent execution of transaction:
Refer to Question No. 8(c) of Long Answer Type Questions.

DATABASE SECURITY

Multiple Choice Type Questions

1. "Each user should provide use id and password to access the database" — This property is known as [MODEL QUESTION]

a) Authentication
b) Athorization
c) Access control
d) Privilege

Answer: (c)

2. Which of the following is the process by which a user's identity is checked? [MODEL QUESTION]

a) Authentication
b) Authorization
c) Access control
d) All of these

Answer: (c)

3. Which of the following is the command that reads SQL statements form DBRMs and produces a mechanism to access data as directed by the SQL statements? [MODEL QUESTION]

a) Compile b) DCLGEN c) BIND d) EXECUTE

Answer: (b)

Long Answer Type Questions

1. Explain the importance of view in providing security to the database. What are the importance of GRANT and REVOKE commands? [MODEL QUESTION]

Answer:

1st Part:

Views can serve as security mechanisms by restricting the data available to users. Some data can be accessible to users for query and modification, while the rest of the table or database is invisible and inaccessible. Permission to access the subset of data in a view must be granted, denied, or revoked, regardless of the set of permissions in force on the underlying table(s).

For example, the salary column in a table contains confidential employee information, but the rest of the columns contain information that should be available to all users. You can define a view that includes all of the columns in the table with the exception of the sensitive salary column. As long as table and view have the same owner, granting SELECT permissions on the view allows the user to see non-confidential columns in the view without having any permissions on the table itself.

By defining different views and granting permissions selectively on them, users, groups, or roles can be restricted to different subsets of data. For example:

- Access can be restricted to a subset of the rows of a base table. For example, define a view that contains only rows for business and psychology books and keep information about other types of books hidden from users.
- Access can be restricted to a subset of the columns of a base table. For example, define a view that contains all the rows of the titles table but omits the royalty and advance columns because this information is sensitive.
- Access can be restricted to a row-and-column subset of a base table.
- Access can be restricted to the rows that qualify for a join of more than one base table. For example, define a view that joins the titles, authors, and title author tables to display the names of authors and books they have written, This view hides personal data about the authors, and financial information about the books.
- Access can be restricted to a statistical summary of data in a base table. For example, define a view that contains only the average price of each type of book.
- Access can be restricted to a subset of another view or of some combination of views and base tables.

2nd Part:

SQL GRANT Command

SQL GRANT is a command used to provide access or privileges on the database objects to the users.

The Syntax for the GRANT command is:

```
GRANT privilege_name
ON object_name
TO {user_name |PUBLIC |role_name}
[WITH GRANT OPTION];
```

For Example: GRANT SELECT ON employee TO user1; This command grants a SELECT permission on employee table to user1.You should use the WITH GRANT option carefully because for example if you GRANT SELECT privilege on employee table to user1 using the WITH GRANT option, then user1 can GRANT SELECT privilege on employee table to another user, such as user2 etc. Later, if you REVOKE the SELECT privilege on employee from user1, still user2 will have SELECT privilege on employee table.

SQL REVOKE Command

The REVOKE command removes user access rights or privileges to the database objects.

The Syntax for the REVOKE command is:

```
REVOKE privilege_name
ON object_name
FROM {user_name |PUBLIC |role_name}
```

For Example: REVOKE SELECT ON employee FROM user1;This command will REVOKE a SELECT privilege on employee table from user1.When you REVOKE SELECT privilege on a table from a user, the user will not be able to SELECT data from that table anymore. However, if the user has received SELECT privileges on that table from more than one users, he/she can SELECT from that table until everyone who granted the permission revokes it. You cannot REVOKE privileges if they were not initially granted by you.

ADVANCED TOPICS

Multiple Choice Type Questions

1. There are two approaches to storing a data in distributed DBMS, ... & fragmentation. [MODEL QUESTION]
   a) availability b) replication c) transparency d) aliases

Answer: (b)

2. The first phase of query processing is [MODEL QUESTION]
   a) decomposition b) restructuring
   c) analysis d) none of these

Answer: (a)

3. Which of the following is a function of a Distribution Database Management System? [MODEL QUESTION]
   a) Distributed query processing b) Replicated data management
   c) Distributed data recovery d) All of the above

Answer: (d)

4. Block-interleaved distributed parity is RAID level [MODEL QUESTION]
   a) 2 b) 3 c) 4 d) 5

Answer: (c)

5. A distributed database is a [MODEL QUESTION]
   a) database that is distributed among a network of geographically separated locations
   b) collection of locations, each of which is operated as a local database system while accessing data at several locations
   c) user program which interacts with the DDBMSs
   d) process which cooperates in completing transactions
   e) software that manages the collection of storage locations and data structures

Answer: (a)

Short Answer Type Questions

1. Discuss the advantages of centralized and distributed databases.

Answer: [MODEL QUESTION]

Centralized database

Early data processing systems worked with a single centralized database.
Smaller local systems started to store local databases as well as do local data processing.
Move to central database distributed to local processors as long as:
Accurate updating of data

Integrity of Data
Sharing of Data
Central Administrative controls
Could all be assured and maintained.

For advantages of distributed database:

1. Enriched Modeling Capabilities.
2. Extensibility.
3. Removal of Impedance Mismatch.
4. More Expressive Query Language.
5. Support for Schema Evolution.
6. Support for Long Duration Transactions.
7. Applicability to Advanced Database applications.
8. Improved Performance

2. Discuss the advantages of distributed and centralized databases. **[MODEL QUESTION]**

Answer:

For advantages of centralized database

Refer to Question No. 1 of Short Answer Type Questions.

Disadvantages of centralized database are,

1. When the central system fails, then the database system fails totally.
2. Data is not easy to get when user at remote sites,
3. The cost of data communication may more high than the distributed database

3. Discuss the differences between DDBMS and OODBMS. **[MODEL QUESTION]**

Answer:

Differences between DDBMS and OODBMS:

Sl.	Distributed DBMS	Object Oriented DBMS
1.	Local Autonomy	Complex objects must be supported
2.	No reliance on a centralized site	Object identity must be supported.
3.	Continuous operation	Encapsulation must be supported
4.	Location independence	Types or Classes must be supported
5	Replication independence	Types or Classes must be able to inherit from their ancestors
6	Fragmentation independence	Dynamic binding must be supported
7	Distributed Query processing	The DML must be computationally complete.
8	Distributed transaction management	The set of data types must be extensible
9	Hardware independence	Data persistence must be provided.
10	Operating System independence	The DBMS must be capable of managing very large databases

Sl.	Distributed DBMS	Object Oriented DBMS
11	Network independence	The DBMS must support concurrent use
12	DBMS independence	DBMS must be able to recover from hardware/software failures and provide a simple way of querying data.

4. How is data warehouse different from a database? **[MODEL QUESTION]**

Answer:

A database is used to store data while a data warehouse is mostly used to facilitate reporting and analysis. Basically, database is just where the data is stored; in order to access this data or analyze it a database management system is required. However, a data warehouse does not necessarily require a DBMS. The purpose of a data warehouse is for easy access to the data for a user. The data warehouse may also be used to analyze the data; however the actual process of analysis is called data mining.

Some differences between a database and a data warehouse:

- A database is used for Online Transactional Processing (OLTP) but can be used for other purposes such as Data Warehousing.
- A data warehouse is used for Online Analytical Processing (OLAP). This reads the historical data for the Users for business decisions.
- In a database the tables and joins are complex since they are normalized for RDMS. This reduces redundant data and saves storage space.
- In data warehouse, the tables and joins are simple since they are de-normalized. This is done to reduce the response time for analytical queries.
- Relational modeling techniques are used for RDMS database design, whereas modeling techniques are used for the Data Warehouse design.
- A database is optimized for write operation, while a data warehouse is optimized for read operations.
- In a database, the performance is low for analysis queries, while in a data warehouse, there is high performance for analytical queries.
- A data warehouse is a step ahead of a database. It includes a database in its structure.

5. Define data mining. What is the advantages data mining over traditional approaches? **[MODEL QUESTION]**

Answer:

1st Part:

Data mining, which is also known as knowledge discovery, is one of the most popular topics in information technology. It concerns the process of automatically extracting useful information and has the promise of discovering hidden relationships that exist in large databases. These relationships represent valuable knowledge that is crucial for many applications. Data mining is not confined to the analysis of data stored in data warehouses. It may analyze data existing at more detailed granularities than the summarized data provided in a data warehouse. It may also analyze transactional, textual, spatial, and multimedia data which are difficult to model with current multidimensional database technology.

2nd Part:
With the help of data mining, organizations are in a better position to predict the future regarding the business trends, the possible amount of revenue that could be generated, the orders that could be expected and the type of customers that could be approached. The traditional approaches will not be able to generate such accurate results as they use simpler algorithms. One major advantage of data mining over a traditional statistical approach is its ability to deal directly with heterogeneous data fields.
The advantages of data mining helps the businesses grow help the customers be happy, and help in a lot of other areas like data management.

6. What do you understand by Web Mining? Compare Web Mining with Data Mining. **[MODEL QUESTION]**

Answer:
Web Mining is the use of the data mining techniques to automatically discover and extract information from web documents/services. It is used for discovering useful information from the World-Wide Web and its usage patterns. In web mining we use data mining techniques to make the web more useful and more profitable (for some) and to increase the efficiency of our interaction with the web.
Web mining can involve all of the traditional data mining processes of classification, segmentation, clustering, association, prediction, and modeling the only difference is that the analyzes result in immediate action. Unlike data mining, web mining is dependent on the use of software agent to trigger targeted offers as events take place in real time. Web mining unlike data mining which existed prior the Internet explosion involves a new paradigm of data collection, integration and analysis. Web mining involves pattern recognition via a seamless stream of activity taking place over a decision network and not a static warehouse. Web mining works by performing data analysis via networks, using software agents to mine, collaborate and discover conditions and features which can lead to increases in sales, cross-selling opportunities and the targeting of specific products or services.

Long Answer Type Questions

1. Write short note on Vertical and Horizontal Fragmentation. **[WBUT 2009]**

Answer:
Horizontal Fragmentation – defines a subset of a relation based on a selection predicate over relation attribute values;
Vertical Fragmentation – specifies fragments according to the subsets of the attributes accessed by applications.
Let us consider the following PROJ table. We show the horizontal and vertical fragmentations in the examples below.

PROJ

PNO	PNAME	BUDGET	LOCATION
P1	BIO INF	50000	MUMBAI
P2	E.LEARNING	30000	DELHI
P3	PLASMA	100000	CHENNAI
P4	AIRCRAFT	150000	KOLKATA

Example of horizontal fragmentation

PROJ1

PNO	PNAME	BUDGET	LOCATION
P1	BIO INF	50000	MUMBAI
P2	E.LEARNING	30000	DELHI

PROJ2

PNO	PNAME	BUDGET	LOCATION
P3	PLASMA	100000	CHENNAI
P4	AIRCRAFT	150000	KOLKATA

Here in the first fragment it is PROJ1= $SL_{budget <= 50000}$ (PROJ)
In the second fragment it is PROJ2= $SL_{budget > 50000}$ (PROJ)
We note: SL stands for the relational operator selection

Example of vertical fragmentation

PROJ1

PNO	BUDGET
P1	50000
P2	30000
P3	100000
P4	150000

PROJ1

PNO	PNAME	LOCATION
P1	BIO INF	MUMBAI
P2	E.LEARNING	DELHI
P3	PLASMA	CHENNAI
P4	AIRCRAFT	KOLKATA

Here in the first fragment it is PROJ1= $PJ_{PNO,BUDGET}$ (PROJ)
In the second fragment it is PROJ2 = $PJ_{PNO,PNAME,LOCATION}$ (PROJ)
We note: PJ stands for the relational operator "PROJECTION"

2. What are the advantages of distributed database management systems over centralized database management systems? [MODEL QUESTION]

Answer:

Advantages & Disadvantages of Object Oriented Model:

The object oriented model is based on a collection of objects. The data describing the object are called **attributes** of the object. An object also contains **codes** that operates on the attributes are called **methods**. An object that contains the same type of attributes and same types of methods are grouped into **classes**. Methods in the object manipulate the data or perform computations on data or monitor the object. The data of an object can only be accessed by another object through the method of that object. The object model should support the following features:

- Complex objects
- Object identity
- Object encapsulation
- Object type or Class
- Inheritance

- Avoid premature binding
- Must be computationally complete
- Should be extensible

Advantages:

1. Enriched Modeling Capabilities.
2. Extensibility.
3. Removal of Impedance Mismatch.
4. More Expressive Query Language.
5. Support for Schema Evolution.
6. Support for Long Duration Transactions.
7. Applicability to Advanced Database applications.
8. Improved Performance

Disadvantages:

1. Lack of Universal Data Model.
2. Lack of Experience.
3. Lack of Standards.
4. Query Optimization compromises Encapsulation.
5. Object Level Locking may impact Performance.
6. Complexity.
7. Lack of Support for Views.
8. Lack of Support for Security.

3. a) Identify the basic features in an object oriented data model.
b) Compare inheritance in the EER model and in the OO model.

[MODEL QUESTION]

Answer:

a) Basic features of an object oriented data model are

- ***Extensibility***
  OODBMS allows new data types to be developed from existing types. For example, common properties of several classes can be identified and combined into a super class and thus reduce complexity and redundancy.
- ***Advanced modeling capabilities***
  OODBMS allow for a better representation of a problem, which can be modeled more closely to real-life objects.
- ***Removal of impedance mismatch***
  OODBMS implements a single interface between DML and the programming language, which in turn reduces unnecessary overhead associated with mapping declarative languages such as SQL to programming language like C.
- ***More Expressive query language***
  OODBMS implement a declarative query language based on object-oriented form of SQL, which is generally more usable than the navigational access.

- ***Support for schema evolution***
  Object oriented facilities such as inheritance and generalization provide for a better-structured schema, which again captures a better representation of the real-world objects.
- ***Improved performance***
  The facilities provided by the object-oriented environment have proved very effective. Previous research done in this field and many tests indicate that the performance of OODBMS can be up to a 30-fold improvement on the RDBMS performance.

b) EER model and in the OO model

i) schema comprehension: ternary relationships are significantly easier to comprehend in the EER model than in the OO model;
ii) the EER model supasses the OO model for designing unary and ternary relationships;
iii) time: it takes less time to design EER schemas;
iv) preferences: the EER model is preferred by designers.

We conclude that even if the objective is to implement an OO database schema, the following procedure is still recommended: 1) create an EER conceptual schema, 2) map it to an OO schema, and 3) augment the OO schema with behavioral constructs that are unique to the OO approach.

4. a) Explain the concept of Distributed Database. What are the advantages and functions of distributed database?
b) Describe briefly the following terms which are needed for Distributed Database Design:

Data Fragmentation.
Data Replication.
Data Allocation.

[MODEL QUESTION]

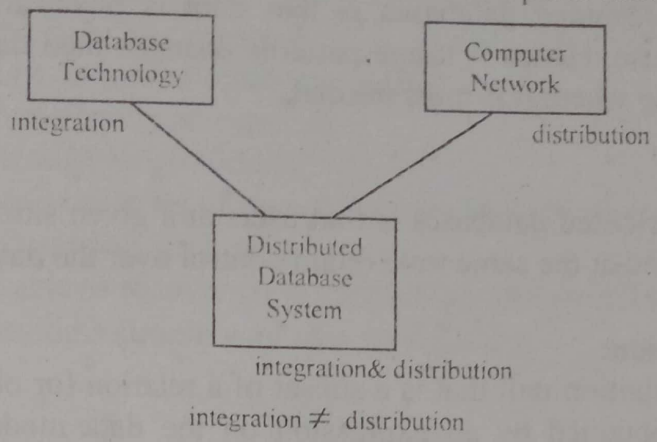
Answer:

a) A distributed database system (DDBMS) is a technology fostered by the development of database technology and computer network. Database systems provide data independence for user applications by offering a logical view of the data that is independent of physical implementation.

On the other side, computer networks allow physically distributed machines to communicate. This is relevant to database technology because one may consider that for various reasons like privacy, efficiency, reliability, scalability users may want their data to be closer to him/her or be replicated through different nodes and still obtain an integrated global view of it.

Thus, DDBMSs explores the communication facilities of a computer network to offer distributed users the same level of services obtained in a centralized database system: query processing, transaction management, recovery processing, security guaranties, and constraint support.

So it may be considered as a logically interrelated collection of shared data (and a description of this data), physically distributed over a computer network.



Types of Distributed Database

1. Homogeneous: Every site runs the same type of DBMS
2. Heterogeneous: Different sites run different DBMS (maybe even RDBMS and ODBMS)

Distributed DBMS Architectures

Client-Server

1. Client sends query to each database server in the distributed system
2. Client caches and accumulates responses

Collaborating Server

1. Client sends query to "nearest" Server
2. Server executes query locally
3. Server sends query to other Servers, as required
4. Server sends response to Client

There are a number of advantages to using a distributed DBMS. These include the following:

1. Capacity and incremental growth

As the volume of data grows at different locations, new sites can be added with little or no upheaval to the DBMS. Compare this to the situation in a centralized system, where growth entails upgrading with changes in hardware and software that effect the entire database.

2. Reliability and availability

An advantage of distributed databases is that even when a portion of a system (i.e. A local site) is down, the overall system remains available. With replicated data, the failure of one site still allows access to the replicated copy of the data from another site. The

remaining sites continue to function. The greater accessibility enhances the reliability of the system.

3. Efficiency and flexibility

An advantage of distributed databases is that data is physically stored close to the anticipated point of use. Hence if usage patterns changes then data can be dynamically moved or replicated to where it is most needed.

4. Sharing

An advantage of distributed databases is that users at a given site are able to access data stored at other sites and at the same time retain control over the data at their own site

b) Data Fragmentation:

A fragment is a distribution unit that is a subset of a relation (or object in a more general sense) that can be obtained by an expression on the data model language. Common expressions used for fragmentation of relations include: selections, projections and semi-joins.

In addition to reduce network transfer cost, fragmentation also maximizes concurrency by allowing concurrent transactions that access disjoint sets of a relation to proceed independently one of the other. In addition, a single query can also benefit from fragmentation by accessing in parallel different fragments

Data Replication:

Database replication is the creation and maintenance of multiple copies of the same database.

Advantages of Replication:

- **Availability**: failure of site containing relation *r* does not result in unavailability of *r* is replicas exist.
- **Parallelism**: queries on *r* may be processed by several nodes in parallel.
- **Reduced data transfer**: relation *r* is available locally at each site containing a replica of *r*.

Data Allocation: Data allocation is the process of defining the location a fragment should be placed in. The location transparency property defines that applications should bother about where a fragment has been placed. An extension of the location transparency includes fragment replication policy

5. Illustrate the features of object oriented database. What are the advantages and disadvantages of using object oriented database management system?

Answer: **[MODEL QUESTION]**

1st part:

13 mandatory features of an ORDBMS are,

1. Complex objects must be supported.

2. Object identity must be supported.
3. Encapsulation must be supported.
4. Types or Classes must be supported.
5. Types or Classes must be able to inherit from their ancestors.
6. Dynamic binding must be supported.
7. The DML must be computationally complete.
8. The set of data types must be extensible.
9. Data persistence must be provided.
10. The DBMS must be capable of managing very large databases.
11. The DBMS must support concurrent users.
12. DBMS must be able to recover from hardware/software failures.
13. DBMS must provide a simple way of querying data.

2nd part:

List of advantages and disadvantages of using an OODBMS over an RDBMS with an object oriented programming language

1. **Composite Objects and Relationships:** An OODBMS can outperform an RDBMS by ten to a thousand times depending on the complexity of the data being handled.
2. **Class Hierarchy:** Data in RDMS has hierarchical characteristics but it is easier to describe in an OODBMS. For example; an employee can be a manager or not in an RDBMS can be described by a type identifier or creating a separate table using foreign keys between Managers and Employees. In an OODBMS, the Employee class is simply a parent class of the Manager class.
3. **Necessity of a Query Language:** A SQL is not necessary for accessing data from an OODBMS unlike an RDBMS It can be translucently accessed through objects. However, queries can be used in OODBMS.
4. **Impedance mismatch** is completely avoided when using an OODBMS. Like; a typical application that uses an object oriented programming language and an RDBMS, a significant amount of time is usually spent mapping tables to objects and back and many a times atomic type data in a RDBMS do not map cleanly.
5. **No Primary Keys:** in a RDBMS for unique identification any two tuples can not have the same primary key values In an OODBMS, the unique identification of objects is done in abstraction via OIDs. Thus there is no limitation on the values that can be stored in an object.
6. **One Data Model**: A data model typically model entities and their relationships in RDBMS with E-R diagram. In OODBMS there is no disconnecting between the database model and the application model because the entities are just other objects in the system. An entire application can thus be comprehensively modelled in one UML diagram.

Disadvantages

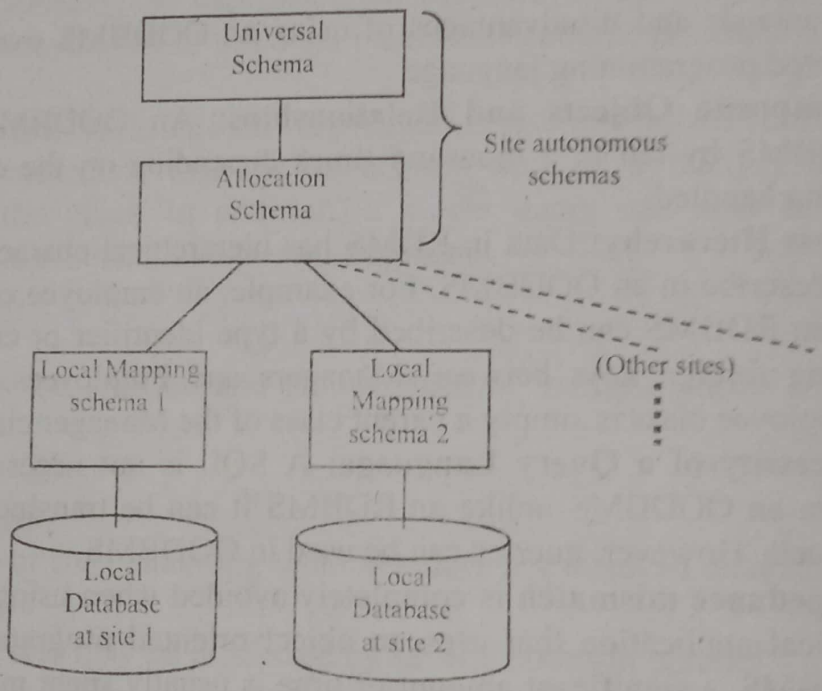
1. **Schema Changes:** In an RDBMS modifying the database schema is simple. While in an OODBMS modifying the schema means those changes have to be

made to the classes in that interact with instances of that class. This implies all schema changes in an OODBMS will involve a system wide recompilation.

2. **Language Dependence:** An OODBMS is typically use a specific language via the API, which is not the case with an RDBMS where simple SQL and ODBC can be used.
3. **Lack of Ad-Hoc Queries:** In an RDBMS ad-hoc queries can be used in joining existing tables then querying them. However, in an OODBMS there is a is a loss of flexibility and it cannot be performed on the data in an OODBMS

6. With a suitable diagram discuss the referential architecture of distributed database management system. Give a comparative study of the different levels of transparency in DDBMS. How does an auxiliary program help in fetching data in a DDBMS? [MODEL QUESTION]

Answer:



A reference architecture for distributed databases

The above figure shows a reference architecture for a distributed database. This reference architecture is not explicitly implemented in all distributed databases but conceptually relevant to understand the organization of any distributed database.

At the top level of figure is the universal or global schema. The global schema defines all the data which are contained in the distributed database. However, the data model used for the definition of a universal schema should be convenient for the definition of the mapping to the other levels of the distributed database. Each of the universal/global relation can be split into multiple non overlapping portions which are called fragments. There are several different ways in which to perform the splitting operation. The mapping between universal relations and fragments is defined in the fragmentation schema. This mapping is one to many; i.e., several fragments correspond to one universal relation. but

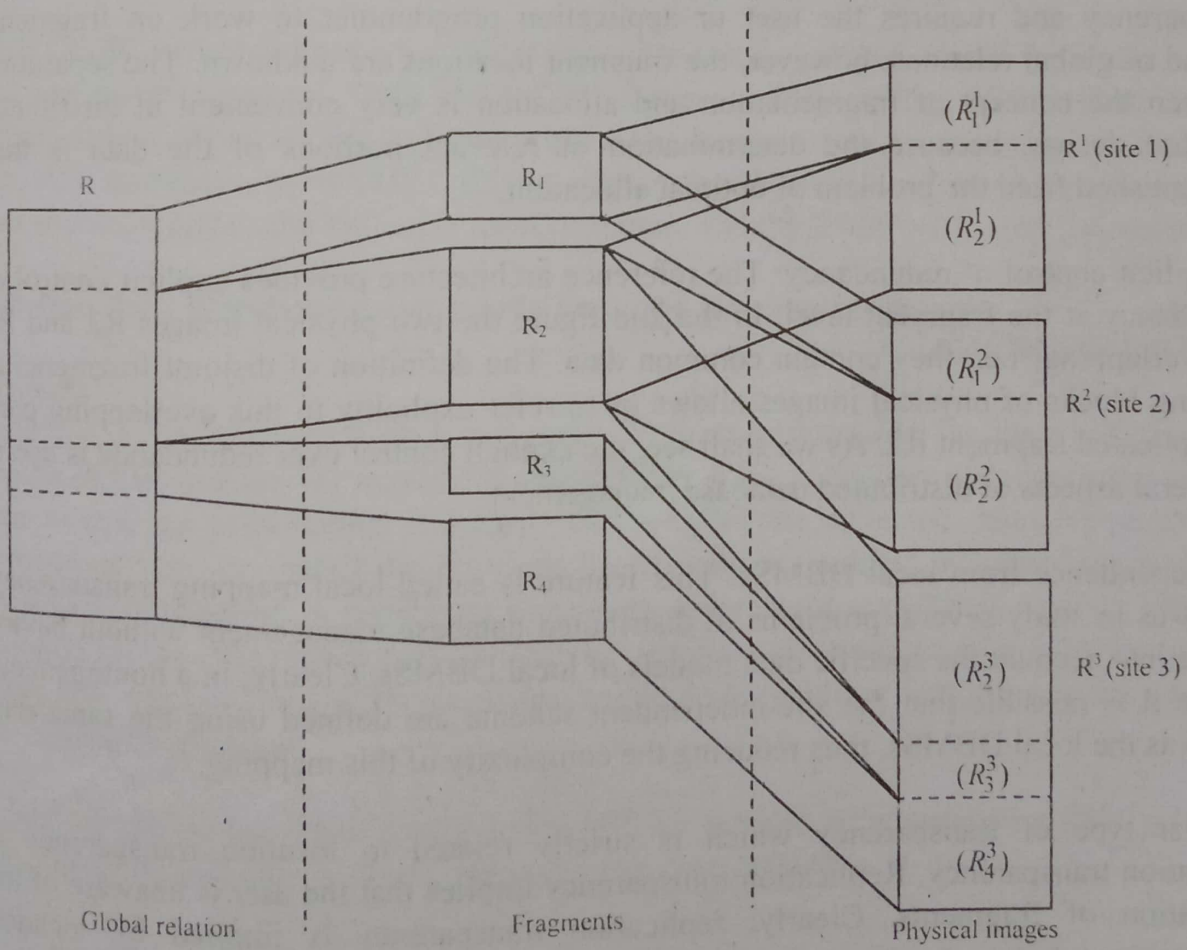
only one universal relation corresponds to one fragment. Fragments are indicated by a universal relation name with an index (fragment index); for example, Ri indicates the ith fragment of global relation R.

Fragments are logical portions of universal relations which are physically located at one or several sites of the network. The allocation schema defines at which site(s) a fragment is located. Note that the type of mapping defined in the allocation schema determines whether the distributed database is redundant or non-redundant; in the former case the mapping is one to many, while in the latter case the mapping is one to one. All the fragments which correspond to the same universal relation R and are located at the same site j constitute the physical image of global relation R at site j. Therefore a one to one mapping between a physical image and a pair (global relation, site); physical images can be indicated by a global relation name and a site index.

Give a comparative study of the different levels of transparency in DDBMS.

The example of the relationship between the object types defined above is shown in figure below. A global relation R is split into four fragments R1, R2, R3 and R4. These four fragments are allocated redundantly at the three sites of a computer network, thus building three physical images R1, R2 and R3.

Let us refer to a copy of a fragment at a given site by the two indexes a fragment index and a site index respectively. E.g., in figure below the notation R_2^3 indicates the copy of fragment R2 which is located at site 3. Finally the two physical images can be identical.



Fragments and physical images for global relations

In this case we say that a physical image is a copy of another physical image, In the figure below, R1 is a copy of R2.
Let us now go back to the figure of the reference architecture. It is described that the relationships between the objects at the top levels of this architecture. These levels are site independent; therefore, they do not depend on the data model of local DBMSs. At a lower level, it is necessary to map the physical images to the objects which are manipulated by the local DBMSs. This mapping is called a local mapping schema and depends on the type of local DBMS; therefore in a heterogeneous system we have different types of local mappings at different sites.

This architecture provides a very general conceptual framework for understanding distributed databases. The three most important objectives are the separation of data fragmentation and allocation, the control of redundancy and the independence from local DBMSs.

1. Separating the concept of data fragmentation from the concept of data allocation. This separation allows to distinguish two different levels of distribution transparency, namely fragmentation transparency and location transparency. Fragmentation transparency is the highest degree of transparency and consists of the fact that the user or application programmer works on global relations. Location transparency is a lower degree of transparency and requires the user or application programmer to work on fragments instead of global relations; however, the fragment locations are unknown. The separation between the concept of fragmentation and allocation is very convenient in distributed database design, because the determination of relevant portions of the data is thus distinguished from the problem of optimal allocation.

2. Explicit control of redundancy: The reference architecture provides explicit control of redundancy at the fragment level. In the 2nd figure the two physical images R2 and R3 are overlapping; i.e., they contain common data. The definition of disjoint fragments as building blocks of physical images allows us to refer explicitly to this overlapping part; the replicated fragment R2. As we shall see, the explicit control over redundancy is useful in several aspects of distributed database management.

3. Independence from local DBMSs: This feature is called local mapping transparency, allows us to study several problems of distributed database management without having to take into account the specific data models of local DBMSs. Clearly, in a homogeneous system it is possible that the site-independent schema are defined using the same data model as the local DBMSs, thus reducing the complexity of this mapping.

Another type of transparency which is strictly related to location transparency is replication transparency. Replication transparency implies that the user is unaware of the replication of fragments. Clearly, replication transparency is implied by location transparency; however in certain cases it is possible that the user has no location transparency but has replication transparency implies use of one particular copy and the

system makes appropriate actions on the other copies. Finally, we can say that it performs the same function as in traditional DBMSs, and it is not necessarily implemented by the existing systems, but it shows which levels and schema are conceptually relevant.

How does an auxiliary program help in fetching data in a DDBMS?
Install an ODBC support for a DBMS by defining the ODBC driver with the ODBC Data Source Administrator. Nothing to worry about this task because it is typically carried out by an installation program. Once the ODBC driver has been installed and a database created, create a DSN.

DSNs are used to map a specific data source to an ODBC driver. Following are the three valid types of DSNs supported:

- ***User data source:*** This data source is local to the computer on which it is created and can be used only by the user who creates it.
- ***System data source:*** This data source belongs to the computer on which it is created and belongs to that computer rather than the user who created it. Any users can access this data source as long as they have the correct privileges.
- ***File data source:*** This data source is specific to the underlying database file, in other words, any user that has the appropriate driver installed can use the data source.

Each ODBC driver defines what information is needed to create a DSN for a data source that the driver supports. Once the ODBC driver is selected, the ODBC Data Source Administrator, when adding a DSN, instructs the driver to display its configuration dialog box. At a minimum, most ODBC drivers require the physical filename that represents the data source, whether the system is local or remote and the folder where the file resides.

7. Define Data Warehouse and briefly describe its characteristics.

[MODEL QUESTION]

Answer:

A data warehouse consists of a computer database responsible for the collection and storage of information for a specific organization. This collection of information is then used to manage information efficiently and analyze the collected data. Although data warehouses vary in overall design, majority of them are subject oriented, meaning that the stored information is connected to objects or events that occur in reality. The data provided by the data warehouse for analysis provides information on a specific subject, rather than the functions of the company and is collected from varying sources into one unit having time-variant.

A data warehouse has significantly different features from other enterprise-wide systems, particularly in how data is stored, managed and manipulated.

There are four key characteristics which separate the data warehouse from other major operational systems:

1. **Subject Orientation:** Data organized by subject
2. **Integration:** Consistency of defining parameters
3. **Non-volatility:** Stable data storage medium
4. **Time-variance:** Timeliness of data and access terms

QUESTION 2015

GROUP – A
(Multiple Choice Type Questions)

1. Choose the correct alternatives for the following:

i) Check-pointing is associated with

✓a) log based recovery
b) non-log based recovery
c) both (a) and (b)
d) none of these

ii) The different levels of data abstraction are

a) physical level
b) logical level
c) view level
✓d) all of these

iii) The strategy for processing a query is improved by

a) query evaluation
b) decomposition
✓c) query optimization
d) none of these

iv) View is a

a) temporary table
✓b) virtual able
c) SQL statement
d) query

v) DML stands for

✓a) Data Manipulation Language
b) Data Media Language
c) both (a) & (b)
d) none of these

vi) Closure of F is

a) F
✓b) F+
c) F–
d) F++

vii) Transaction follows

✓a) acid properties
b) non-preemption property
c) preemption property
d) starvation property

viii) Relations produced from an ER-model will always be in

✓a) 1NF
b) 2NF
c) 3NF
d) 4NF

ix) BCNF stands for

a) Boyle Codd Normalization
b) Boyce Cold Normal Form
✓c) Boyce Codd Normal Form
d) none of these

x) Which of the following guarantees that, : A transaction is either performed in its entirety or not performed at all"?

a) consistency b) durability c) isolation ✓d) atomicity

xi) Which of the following is true?

a) a super key is always a candidate key

b) every 3NF schema is also in BCNF

✓c) generalization is a bottom-up design approach

d) none of these

GROUP – B

(Short Answer Type Questions)

2. a) What do you mean by degree, cardinality of relationship?

b) What do you mean by data abstraction? Explain three levels of data abstraction?

a) See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 10.

b) See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 11.

3. Discuss the ACID properties of a Database transaction.

See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 2.

4. a) What is Data dictionary?

b) What do you mean by unary and binary operations in Relational algebra? Give example.

a) See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 4.b) (OR).

b) See Topic: RELATIONAL MODEL, Short Answer Type Question No. 6.

5. Explain with example "BCNF is stricter than 3NF".

See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Short Answer Type Question No. 2.

6. Briefly explain two phase locking protocol.

See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 1.

GROUP – C

(Long Answer Type Questions)

7. Consider the following employee database, primary keys are underlined.

Employee(employee-name, street, city)

Works(employee-name, company-name, salary)

Company(company-name, city)

Manages(employee-name, manager-name)

Write SQL's for the queries given below:

(i) Find the names of all employees who work for XYZ.

(ii) Find all employees in the database who live in the same cities aas the companies for which they work.

(iii) Find all employees in the database who live in the same cities and on the same streets as do their managers.

(iv) Find all employees who earn more than the average salary of all employees of their company.

(v) Find the company that has the smallest payroll.

See Topic: SQL, Long Answer Type Question No. 6.

8. a) Consider the following two sets of functional dependencies

X = {A→C, AC→D, E→AD, E→H} and Y = {A→CD,E→AH}.

Check whether or not they are equivalent.

b) Consider R = (A, B, C, D, E) and set of functional dependencies

F = {A→BC, CD→E, B→D, E→A} to answer the following questions:

(i) List the candidate keys for R.

(ii) Show that the following decomposition of the schema R into (A,B,C) and (A, D, E) is lossless-join decomposition if the above given set F of functional dependencies holds.

(iii) Compute the canonical cover FC.

c) What normal form is the following relation in? Discuss stuff (D, O, N, T, C, R, Y)

FD's are DO→NTCRY, CR→D, D→N.

See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 16.

9. Supreme products manufacture products like pressure cookers, cook wares, water purifiers, food processors etc. The company markets its product to wholesalers all over the country and dealers sell them to customer. The company has five regional offices. Sales persons contact dealers and explain about products, incentives offered, panting programs for wholesalers and demonstration for customer etc. Dealers place orders with the sales persons attached with the regional office of their location. After receiving goods they make payments, which may be in installments. Company would like to develop a system to monitor sale of different products, performance of salespersons and order from wholesalers. Do the following:

i) Identify entities, attributes and relationship.

ii) Draw an E-R diagram.

iii) Convert this to relational tables.

See Topic: ENTITY-RELATIONSHIP MODEL, Long Answer Type Question No. 5.

10. a) Explain the purpose of the checkpoint mechanism. How often should checkpoints be performed?

b) How does 'strict two-phase' and 'consecutive two-phase' locking protocol differ.

c) Construct a B+ tree for the following set of key values – (3,4,6,8,12,17,19,23,29,31) assume that the tree is initially empty, values are added in ascending order and the number of pointer in one node is 3. Also perform the following operations of B+ tree: (i) insert 10 (ii) insert 11 (iii) delete 29.

a) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 10(a).
b) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 10(b).
c) See Topic: STORAGE STRATEGIES, Long Answer Type Question No. 5.

11. a) Describe Three-Schema architecture of DBMS. Define physical data independence and logical data independence.
b) Explain with examples the term super key, candidate key, primary key and alternate key.
c) Consider the file with $r = 3000$ records (fixed length) of size R = 100 bytes stored on a disk with block size B = 1024 bytes. Suppose each index entry in index file takes 15 bytes (9 bytes for index value, 5 bytes for pointer). What is the number of accessing blocks for clustering index?

a) 1st part: See Topic: DATABASE SYSTEM ARCHITECTURE, Short Answer Type Question No. 1.
2nd part: See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 1.
b) See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 2.
c) See Topic: STORAGE STRATEGIES, Long Answer Type Question No. 6.

QUESTION 2016

GROUP – A
(Multiple Choice Type Questions)

1. Choose the correct alternatives for the following:

i) Overall logical structure of a database can be graphically represented by
✓a) ER diagram b) Records c) Hierarchy d) Relation

ii) Which key cannot be null?
a) Unique key ✓b) Primary key c) Super key d) Foreign key

iii) Relational calculus is a
✓a) Query language b) procedural language
c) Non-procedural language d) None of these

iv) 2 NF is based on
a) full dependency b) transitive dependency
✓c) functional dependency d) partial dependency

v) The information about data in a database is called
✓a) Meta data b) Hyper data c) Tera data d) None of these

vi) A row from a table is selected by
✓a) selection operator b) projection operator
c) union operator d) none of these

vii) Which data type can store unstructured data?

✓a) Raw b) Char c) Numeric d) Varchar

viii) A normal form in which every non-prime attribute is fully dependent on prime attribute is

a) 1 NF ✓b) 2 NF c) 3 NF d) BCNF

ix) Serializability of concurrent transaction is ensured by

a) locking b) time stamping

✓c) both (a) and (b) d) none of these

x) Transaction follows

✓a) ACID properties b) Starvation properties

c) Preemption properties d) Non-preemption properties

GROUP – B

(Short Answer Type Questions)

2. What is Data dictionary? What do you mean by unary operations in Relational algebra? Give example.

1st part: See Topic: DATABASE SYSTEM ARCHITECTURE, Short Answer Type Question No. 8.

2nd part: See Topic: RELATIONAL MODEL, Short Answer Type Question No. 6.

3. Explain two-phase locking protocol.

See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 5.

4. Consider the relation $R = \{A, B, C, D, E, F, G, H, I, J\}$ and the set of functional dependencies:

$$F = \{AB \rightarrow C, A \rightarrow DE, B \rightarrow F, F \rightarrow GH, D \rightarrow IJ\}$$

Decompose R into 3NF.

See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Short Answer Type Question No. 11.

5. Discuss the different level of views.

See Topic: SQL, Short Answer Type Question No. 6.

6. What is Weak entity set? Explain with suitable example.

See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 12.

GROUP – C

(Long Answer Type Questions)

7. a) Draw the ER diagram of a hospital management system and explain.

b) Consider the relation $R = \{A, B, C, D, E\}$ and the set of functional dependencies:

$$F = \{AD \rightarrow B, B \rightarrow C, C \rightarrow D\}$$

Find out the candidate key.

a) See Topic: ENTITY-RELATIONSHIP MODEL, Long Answer Type Question No. 6.

b) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 17.

8. a) What do you mean by transaction? Explain the transaction states.

b) Explain log based recovery and checkpoints.

c) What do you mean by shadow paging?

d) What do you mean by deadlock handling? Explain in detail.

a) See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 6.

b) See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 7.

c) See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 8.

d) See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 9.

9. Consider the employee database:

Employee (emp_name, street, emp_id)

Works (emp_name, company_name, salary)

Company (company_name, city)

Manages (emp_name, manager_name)

Write the appropriate SQL statement on the basis of the above table:

a) Find the names and cities of residence of all employees who work for the UBI.

b) Find the names, street addresses and cities of residence of all employees who work for the UBI and earn more than Rs. 50,000.

c) Find all employees in the database who do not work for UBI.

d) Find the 2nd highest salary for employees in UBI.

e) Find the company that has the most employees.

See Topic: SQL, Long Answer Type Question No. 7.

10. a) What are the advantages of normalization?

b) How does BCNF differ from 3rd normal form?

c) Explain the ACID properties of transactions.

a) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Short Answer Type Question No. 12.

b) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Short Answer Type Question No. 2.

c) See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 2.

11. Write short notes on any three of the following:

a) File indexing

b) B+ tree

c) Advantages of DBMS

d) Database models

e) Inner join and Outer join

a) See Topic: STORAGE STRATEGIES, Long Answer Type Question No. 7(b).

b) See Topic: STORAGE STRATEGIES, Long Answer Type Question No. 7(c).

c) See Topic: DATABASE SYSTEM ARCHITECTURE, Long Answer Type Question No. 1.

d) See Topic: RELATIONAL MODEL, Long Answer Type Question No. 3(b).

e) See Topic: RELATIONAL MODEL, Long Answer Type Question No. 3(c).

QUESTION 2017

GROUP – A

(Multiple Choice Type Questions)

1. Choose the correct alternatives for the following:

i) The information about data in a database is called

✓a) Metadata b) Teradata c) Hyperdata d) None of these

ii) What is the highest normal form for the relational schema Bank?

a) First b) Second c) Third ✓d) Boyce code

6

v) Serializability of concurrent transactions are ensured by

✓a) Locking b) Drop c) both of these d) none of these

vi) Which index is specified on the non-ordering fields of a file?

a) Primary b) Clustering ✓c) Secondary d) none of these

vii) One of the shortcomings of the file system is

a) Data availability b) Fixed records

c) Sequential records ✓d) Lack of security

viii) In the E-R diagram the term 'Cardinality' is synonymous to

a) Attribute ✓b) Degree c) Entities d) Cartesian

ix) The employee salary should not be greater than Rs. 20,000. This is

✓a) integrity constraint
b) referential constraint
c) over-defined constraint
d) feasible constraint

x) What is the name of a trigger that triggers itself?

a) Triggering trigger
✓b) Cascading trigger
c) Mutating trigger
d) none of these

GROUP – B
(Short Answer Type Questions)

2. Discuss the ACID properties of transactions.

See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 2.

3. a) Distinguish between file management system and database management system.
b) Discuss the role of DBA.

a) See Topic: STORAGE STRATEGIES, Short Answer Type Question No. 1.
b) See Topic: DATABASE SYSTEM ARCHITECTURE, Short Answer Type Question No. 2.

4. What is Cardinality ratio? What is the difference between procedural and non-procedural DML? Describe different types of attribute.

1st part: See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 8.
2nd part: See Topic: DATABASE SYSTEM ARCHITECTURE, Short Answer Type Question No. 3.
3rd part: See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 13.

5. What is closure and minimal cover? What is inclusion dependency?

See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Short Answer Type Question No. 13.

6. What is 2-phase locking protocol? How does it guarantee serializability?

See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 1.

GROUP – C
(Long Answer Type Questions)

7. a) Find out the closure of attribute set (AD) i.e. $(AD)^+$ in the R. Set of FD's F are as given below:

$R = \{A, B, C, D, E\}$.

$FD = \{B \rightarrow CD, D \rightarrow E, B \rightarrow A, E \rightarrow C, AB \rightarrow B\}$.

b) Find out the candidate keys for R.

c) Consider the following two sets of FDs:

$F = \{A \to C, AC \to D, E \to AD, E \to H\}$

$G = \{A \to CD, E \to AH\}$.

Check whether they are equivalent. Justify your answer.

See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 18.

8. Consider the relational database:

Employee (person-name, street, city)

Works (person-name, company name, salary)

Company (company name, city)

Manages (person-name, manager-name)

Write down appropriate SQL statement for the following queries:

a) Find the name of all employees who work for 'SBI bank'.

b) Find name, street address, cities of residence of all employees who work for 'UBI Bank' and earn more than Rs. 5,00,000 per annum.

c) Find the second highest salary for employees in 'SBI bank'.

d) Find the names of all employees who live in the same city and on the same street as do their manager.

e) Find the company that has the most employees.

See Topic: SQL, Long Answer Type Question No. 7.

9. a) What are the various states of a transaction? Explain with a state diagram.

b) Consider the following schedule:

S1: r2 (C), r2 (B), w2 (B), r3 (B), r3 (C), r1 (A), w1 (A), w3 (B), w3 (C), r2 (A), r1 (B), w1 (B), w2(A).

Is the schedule serializable?

c) What is cascadeless schedule? Why is cascadeless of schedule desired?

d) Explain log based recovery.

See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 11.

10. a) If $R = (A, B, C, D)$ and the FDs are $\{AB \to CE, E \to AB, C \to D\}$. Why R is in 2NF, but not in 3NF? Explain.

b) Show that if a relation schema is in BCNF, then it is in 3NF but if a relation schema is in 3NF then it is not necessary in BCNF. Give examples.

c) What are metadata and data dictionary?

d) Explain the terms candidate key, primary key, foreign key and super key.

a) & b) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 19.

c) See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 4.

d) See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 2.

11. Write short notes on any three of the following:

a) Armstrong's axioms

b) Time stamp based protocol for concurrency protocol

c) Log based recovery

d) Ordered Index

e) Deadlock

a) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 24.(a).

b) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 13(d).

c) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 13(e).

d) See Topic: STORAGE STRATEGIES, Long Answer Type Question No. 7(d).

e) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 13(b).

QUESTION 2018

GROUP – A

(Multiple Choice Type Questions)

1. Choose the correct alternatives for the following:

i) Which of the following is not a property of transaction?

a) atomicity ✓b) concurrency c) isolation d) durability

ii) Check points are part of

✓a) recovery measure b) security measure

c) concurrency measure d) authorization measure

iii) Tree structures are used to store data in

a) network model ✓b) relational model

c) hierarchical model d) file based system

iv) The language that requires a user to specify the data to be retrieved without specifying how to get it is

a) procedural DML ✓b) non procedural DML

c) procedural DDL d) non-procedural DDL

v) The rule that a value of a foreign key must appear as a value of some specific table is called a

✓a) referential constraint b) index

c) integrity constraint d) functional dependency

vi) The clause in SQL that specifies that the query result should be sorted in ascending or descending order based on the values of one or more column is

a) view ✓b) order by c) group by d) having

vii) Conflict serializability can be detected by

a) WFG ✓b) precedence graph c) spanning tree d) none of these

viii) According to the levels of abstractions the schema at the intermediate level is called

✓a) logical schema b) physical schema c) subschema d) super schema

ix) The operation which is not considered as basic operation in relational algebra

✓a) join b) selections c) union d) cross product

x) It is an abstraction through which relationships are treated as higher level entities?

a) generalization b) specialization ✓c) aggregation d) inheritance

GROUP – B

(Short Answer Type Questions)

2. Describe different 2PL protocol in brief.

See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 1.

3. Consider the following database with primary keys underlined

Project (P – No, P – Name, P – incharge)

Employee (E – no, E – Name)

Assigned–To (E – no, P – no)

Write relational algebra expression for the following:

(a) List detail of employee working on all projects.

(b) List E – no, E – name of employee who do not work on project with P – no = DB2003.

See Topic: RELATIONAL MODEL, Short Answer Type Question No. 7.

4. What is partial functional dependency? Explain BCNF with a suitable example.

See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Short Answer Type Question No. 14.

5. Consider the following two transactions:

T1 : read (A); read (B); if A = 0, then B = B + 1; write (B);

T2 : read (B); read (A); if B = 0, then A = A + 1; write (A);

Add lock and unlock instructions to transaction T1 and T2. Is there any dead lock observed?

See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 10.

6. Explain Armstrong axiom in brief.

See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Short Answer Type Question No. 15.

GROUP – C

(Long Answer Type Questions)

7. a) Why do we need normalization?

b) What are the advantages of database over traditional file system?

c) Explain the role of database administrator?

d) Relation assignment (worker-id, building-id, start-date, name, skill).

FD's are {worker-id → name, {worker-id, building-id} → start date, name → skill}. Is the relation in 3NF, if not then make it in 3NF.

a) & d) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 20.

b) See Topic: STORAGE STRATEGIES, Short Answer Type Question No. 1.

c) See Topic: DATABASE SYSTEM ARCHITECTURE, Short Answer Type Question No. 2.

8. a) What is the utility of Query Tree?

b) Describe different phase of a Query Tree.

c) Consider the employee query "Select work-on, e-name, from employee, proj-assign project where p-name = "ASSEMBLY" and p-no = "p1" and Join-Date = "21-12-17" and employee.eno = proj-assign. eno and proj-assign, pno=project. p-no.

d) "Primary keys are candidate keys but reverse is not true." – explain.

a) & b) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 21.

c) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 22.

d) See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 14.

9. a) What is deadlock? What are necessary and sufficient condition for deadlock?

b) What are the necessary measurements for deadlock avoidance?

c) Let T_1, T_2 and T_3 be transactions that operate on the same data items A, B and C. Let $r_1(A)$ mean that T_1 read A, $W_1(A)$ means that T_1 writes A and so on for T_2 and T_3. Consider the following schedule:

$S_1 : r_2(C), r_2(B), W_2(B), r_3(B), r_3(C), r_1(A) W_1 : W_3(B), W_3(C), r_2(A), r_1(B), W_1(B), W_2(A)$ Is the schedule serializable?

See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 12.

10. Answer the following queries in SQL using given database schema:
EMP (E no, E name, E add, B date, super-no)
DEPT (D no, D name, mgrno)
PROJECT (P no, P name, D no, P location)
Work-on (E no, D no, Hours)
a) List E no, E name, E add, mgrno of all employee who work in "Research" department.
b) For all project in "Kolkata" list p-no, controlling department name, manager name, address and birth date.
c) List all project number, project name, manager name which belongs to "Product" department.
d) List all employee name who works more than 40 hours in "Research" department.

See Topic: SQL, Long Answer Type Question No. 8.

11. Answer *any three* questions:
a) Time stamp based protocol for concurrency control
b) B+ tree file organization
c) Primary Index and Secondary Index
d) Anomalies in concurrent execution of transaction
e) BCNF

a) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 13(d).
b) See Topic: STORAGE STRATEGIES, Long Answer Type Question No. 7(c).
c) See Topic: STORAGE STRATEGIES, Long Answer Type Question No. 7(e).
d) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 13(f).
e) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 24(c).

QUESTION 2019

GROUP – A
(Multiple Choice Type Questions)

1. Choose the correct alternatives for the following:
i) In the relational modes, cardinality is termed as
✓a) number of tuples
b) number of attributes
c) number of tables
d) number of constraints

ii) Which of the following operations is used if we are interested in only certain columns of a table?
✓a) PROJECTION b) SELECTION c) UNION d) JOIN

iii) The strategy for processing a query is improved by
a) query evaluation
b) decomposition
✓c) query optimization
d) none of these

iv) Cartesian product in relational algebra is

a) a unary operator
✓b) a binary operator
c) a ternary operator
d) not defined.

v)) Transitive dependency is removed in

a) 1 NF
b) 2NF
✓c) 3NF
d) 4NF

vi) Relational algebra is

✓a) procedural
b) non-procedural
c) object oriented
d) none of these.

vii) In 2-plase locking a transaction must

a) release all its locks at the same time
b) not obtain any new locks once it has started releasing locks
✓c) obtain locks on items not used by any other transaction
d) ensure that deadlocks will never occur

viii) DML is provided for

a) description of logical structure of database
b) addition of new structures in the database system
✓c) manipulation & processing of database
d) definition of physical structure of database system.

ix) Which of the following protocols ensures conflict serializability and safety from deadlocks?

a) Two-phase locking protocol
✓b) Time stamp ordering protocol
c) Graph based protocol
d) None of these.

x) Which one of the following is not an indexing technique?

a) Primary index
b) Secondary index
c) Multilevel index
✓d) Sequential index

GROUP – B

(Short Answer Type Questions)

2. a) What is referential integrity?

b) Explain with example the difference between strong and weak entity sets.

a) See Topic: RELATIONAL MODEL, Short Answer Type Question No. 8.

b) See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 15.

3. Define schedule and conflict serializable schedule.

See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 11.

4. Explain with example super key, candidate key and primary key.

See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 2.

5. Define BCNF. How does it differ from 3NF? Why is it considered a stronger than 3 NF?

See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Short Answer Type Question No. 2.

6. What is the main difference between two-phase locking and the time stamping technique concurrency control? Explain briefly.

See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 5.

GROUP – C

(Long Answer Type Questions)

7. a) What is mapping constraint ? Describe three-layer architecture of DBMS.

b) Draw the ER diagram for the system given as follows:

An organization has number of faculties who are expert in one or more subjects. For each subject, numbers of such experts are there, system will store faculty and subject information and must support query on finding expertise on subjects. Students get enrolled to have training on one or more subjects. System will keep student's information also. One faculty is allotted to teach one or more subjects. For one subject only one faculty is assigned. System must keep the information regarding such assignment.

a) See Topic: ENTITY-RELATIONSHIP MODEL, Long Answer Type Question No. 7.

b) See Topic: ENTITY-RELATIONSHIP MODEL, Long Answer Type Question No. 4(a).

8. a) What is transaction?

b) What is ACID property?

c) Explain with example serial and serializable schedule.

d) What are the problems of concurrent execution of transaction?

e) Explain with the help of precedence graph the conflict and non-conflict serializability.

a) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 8(a).

b) See Topic: TRANSACTION PROCESSING, Short Answer Type Question No. 2.

c) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 8(b).

d) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 8(c).

e) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 8(d).

9. a) Explain the terms 'full functional dependency' and 'multivalued dependency' with example.

b) Differentiate between 2NF and 3NF. What is lossless decomposition?

c) What is closure? Explain with example.

d) What do you mean by integrity constraint?

a) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 23(a).
b) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 23(b).
c) See Topic: FUNCTIONAL DEPENDENCIES AND NORMALIZATION, Long Answer Type Question No. 23(c).
d) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 7.

10. a) State two-phase commit protocol and discuss the implications of a failure on the part of the coordinator, a participant, during each of the two phases.
b) Describe the wait-die and wound-wait protocols for deadlock prevention.
c) Define three concurrency problems:
dirty read, non-repeatable read, phantoms.
d) Let T_1, T_2 and T_3 be transactions that operate on the same data items *A*, *B* and *C*. Let r_1 (*A*) means that T_1 reads *A*, w_1 (*A*) means that T_1 writes *A* and so on for T_2 and T_3.
Consider the following scheduled:

$$S_1 : r_2(c), r_2(B), w_2(b), r_3(B), r_3(C), r_1(A), w_1(A),$$
$$w_3(B), w_3(C), r_2(A), r_1(B), w_1(B), w_2(A)$$

Is the schedule serializable?
e) What are the roles of Analysis, Redo and Undo phases in the recovery algorithm 'ARIES'?
a) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 6(a).
b) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 2(b).
c) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 4(b).
d) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 6(b).
e) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 5.

11. Write short notes on any *three* of the following:
a) Two-phase Locking protocol
b) Dead lock
c) Transaction state diagram
d) *B*-tree
e) Data Dictionary

a) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 13(a).
b) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 13(b).
c) See Topic: TRANSACTION PROCESSING, Long Answer Type Question No. 13(c).
d) See Topic: STORAGE STRATEGIES, Long Answer Type Question No. 7(a).
e) See Topic: ENTITY-RELATIONSHIP MODEL, Short Answer Type Question No. 4(b).