TYPICAL QUESTIONS & ANSWERS

PART -I

OBJECTIVE TYPE QUESTIONS

Each Question carries 2 marks.

Choosethe correct or the best alternative in the following:

Q.1 Which of the following relational algebra operations do not require the participating tables to be union-compatible?

(A) Union **(B)** Intersection
(C) Difference **(D)** Join

Ans: (D)

Q.2 Which of the following is not a property of transactions?

(A) Atomicity **(B)** Concurrency
(C) Isolation **(D)** Durability

Ans: (B)

Q.3 Relational Algebra does not have

(A) Selection operator. **(B)** Projection operator.
(C) Aggregation operators. **(D)** Division operator.

Ans: (C)

Q.4 Checkpoints are a part of

(A) Recovery measures. **(B)** Security measures.
(C) Concurrency measures. **(D)** Authorization measures.

Ans: (A)

Q.5 Tree structures are used to store data in

(A) Network model. **(B)** Relational model.
(C) Hierarchical model. **(D)** File based system.

Ans: (C)

Q.6 The language that requires a user to specify the data to be retrieved without specifying exactly how to get it is

(A) Procedural DML. **(B)** Non-Procedural DML.
(C) Procedural DDL. **(D)** Non-Procedural DDL.

Ans: (B)

Q.7 Precedence graphs help to find a

(A) Serializable schedule. **(B)** Recoverable schedule.
(C) Deadlock free schedule. **(D)** Cascadeless schedule.

Ans: (A)

Q.8 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.

Ans: (A) The rule that a value of a foreign key must appear as a value of some specific table is called a referential constraint. (Referential integrity constraint is concerned with foreign key)

Q.9 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 columns is
(A) View **(B)** Order by
(C) Group by **(D)** Having

Ans: (B) 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 columns is ORDER BY. (ORDER BY clause is used to arrange the result of the SELECT statement)

Q.10 What is a disjoint less constraint?
(A) It requires that an entity belongs to no more than one level entity set.
(B) The same entity may belong to more than one level.
(C) The database must contain an unmatched foreign key value.
(D) An entity can be joined with another entity in the same level entity set.

Ans: (A) Disjoint less constraint requires that an entity belongs to no more than one level entity set. (Disjoint less constraint means that an entity can be a member of at most one of the subclasses of the specialization.)

Q.11 According to the levels of abstraction, the schema at the intermediate level is called
(A) Logical schema. **(B)** Physical schema.
(C) Subschema. **(D)** Super schema.

Ans: According to the levels of abstraction, the schema at the intermediate level is called ***conceptual schema***.
(**Note:** All the options given in the question are wrong.)

Q.12 It is an abstraction through which relationships are treated as higher level entities
(A) Generalization. **(B)** Specialization.
(C) Aggregation. **(D)** Inheritance.

Ans: (C) It is an abstraction through which relationships are treated as higher level entities Aggregation. (In ER diagram, aggregation is used to represent a relationship as an entity set.)

Q.13 A relation is in _____ if an attribute of a composite key is dependent on an attribute of other composite key.

(A) 2NF **(B)** 3NF
(C) BCNF **(D)** 1NF

Ans: (B) A relation is in 3 NF if an attribute of a composite key is dependent on an attribute of other composite key. (If an attribute of a composite key is dependent on an attribute of other composite key then the relation is not in BCNF, hence it has to be decomposed.)

Q.14 What is data integrity?

(A) It is the data contained in database that is non redundant.
(B) It is the data contained in database that is accurate and consistent.
(C) It is the data contained in database that is secured.
(D) It is the data contained in database that is shared.

Ans: (B) (Data integrity means that the data must be valid according to the given constraints. Therefore, the data is accurate and consistent.)

Q.15 What are the desirable properties of a decomposition

(A) Partition constraint. **(B)** Dependency preservation.
(C) Redundancy. **(D)** Security.

Ans: (B) What are the desirable properties of a decomposition – dependency preserving. (Lossless join and dependency preserving are the two goals of the decomposition.)

Q.16 In an E-R diagram double lines indicate

(A) Total participation. **(B)** Multiple participation.
(C) Cardinality N. **(D)** None of the above.

Ans: (A)

Q.17 The operation which is not considered a basic operation of relational algebra is

(A) Join. **(B)** Selection.
(C) Union. **(D)** Cross product.

Ans: (A)

Q.18 Fifth Normal form is concerned with

(A) Functional dependency. **(B)** Multivalued dependency.
(C) Join dependency. **(D)** Domain-key.

Ans: (C)

Q.19 Block-interleaved distributed parity is RAID level

(A) 2. **(B)** 3
(C) 4. **(D)** 5.

Ans: (D)

Q.20 Immediate database modification technique uses

(A) Both undo and redo. **(B)** Undo but no redo.
(C) Redo but no undo. **(D)** Neither undo nor redo.

Ans: (A)

Q.21 In SQL the statement **select * from R, S** is equivalent to

(A) Select * from R natural join S. **(B)** Select * from R cross join S.
(C) Select * from R union join S. **(D)** Select * from R inner join S.

Ans: (B)

Q.22 Which of the following is not a consequence of concurrent operations?

(A) Lost update problem. **(B)** Update anomaly.
(C) Unrepeatable read. **(D)** Dirty read.

Ans: (B)

Q.23 As per equivalence rules for query transformation, selection operation distributes over

(A) Union. **(B)** Intersection.
(C) Set difference. **(D)** All of the above.

Ans: (D)

Q.24 The metadata is created by the

(A) DML compiler **(B)** DML pre-processor
(C) DDL interpreter **(D)** Query interpreter

Ans: (C)

Q.25 When an E-R diagram is mapped to tables, the representation is redundant for

(A) weak entity sets **(B)** weak relationship sets
(C) strong entity sets **(D)** strong relationship sets

Ans: (B)

Q.26 When $R \cap S = \phi$, then the cost of computing $R \bowtie S$ is

(A) the same as $R \times S$ **(B)** greater the $R \times S$
(C) less than $R \times S$ **(D)** cannot say anything

Ans: (A)

Q.27 In SQL the word 'natural' can be used with

(A) inner join **(B)** full outer join
(C) right outer join **(D)** all of the above

Ans: (A)

Q.28 The default level of consistency in SQL is

(A) repeatable read **(B)** read committed
(C) read uncommitted **(D)** serializable

Ans: **(D)**

Q.29 If a transaction T has obtained an exclusive lock on item Q, then T can

(A) read Q **(B)** write Q
(C) both read and write Q **(D)** write Q but not read Q

Ans: (C)

Q.30 Shadow paging has

(A) no redo **(B)** no undo
(C) redo but no undo **(D)** neither redo nor undo

Ans: (A)

Q.31 If the closure of an attribute set is the entire relation then the attribute set is a

(A) superkey **(B)** candidate key
(C) primary key **(D)** not a key

Ans: (A)

Q.32 DROP is a _____ statement in SQL.

(A) Query **(B)** Embedded SQL
(C) DDL **(D)** DCL

Ans: (C)

Q.33 If two relations R and S are joined, then the non matching tuples of both R and S are ignored in

(A) left outer join **(B)** right outer join
(C) full outer join **(D)** inner join

Ans: (D)

Q.34 The keyword to eliminate duplicate rows from the query result in SQL is

(A) DISTINCT **(B)** NO DUPLICATE
(C) UNIQUE **(D)** None of the above

` **Ans: (C)**

Q.35 In 2NF

(A) No functional dependencies (FDs) exist.
(B) No multivalued dependencies (MVDs) exist.
(C) No partial FDs exist.
(D) No partial MVDs exist.

Ans: (C)

Q.36 Which one is correct statement?
Logical data independence provides following without changing application programs:
(i) Changes in access methods.
(ii) Adding new entities in database
(iii) Splitting an existing record into two or more records
(iv) Changing storage medium
(A) (i) and (ii) **(B)** (iv) only, **(C)** (i) and (iv) **(D)** (ii) and (iii)

Ans: (D)

Q.37 In an E-R, Y is the dominant entity and X is a subordinate entity. Then which of the following is incorrect :
(A) Operationally, if Y is deleted, so is X
(B) existence is dependent on Y.
(C) Operationally, if X is deleted, so is Y.
(D) Operationally, if X is deleted, & remains the same.

Ans: (C)

Q.38 Relational Algebra is
(A) Data Definition Language .
(B) Meta Language
(C) Procedural query Language
(D) None of the above

Ans: (C)

Q.39 Which of the following aggregate functions does not ignore nulls in its results?.
(A) COUNT . **(B)** COUNT (*)
(C) MAX **(D)** MIN

Ans: (B)

Q.40 R (A,B,C,D) is a relation. Which of the following does not have a lossless join dependency preserving BCNF decomposition
(A) A→B, B→CD **(B)** A→B, B→C, C→D .
(C) AB→C, C→AD **(D)** A→BCD

Ans: (D)

Q.41 Consider the join of relation R with a relation S. If R has m tuples and S has n tuples, then the maximum and minimum size of the join respectively are
(A) m+n and 0 **(B)** m+n and |m-n|
(C) mn and 0 **(D)** mn and m+n

Ans: (C)

Q.42 Maximum height of a B+ tree of order m with n key values is

(A) $Log_m(n)$ **(B)** (m+n)/2

(C) $Log_{m/2}(m+n)$ **(D)** None of these

Ans: (D)

Q.43 Which one is true statement :

(A) With finer degree of granularity of locking a high degree of concurrency is possible.

(B) Locking prevents non – serializable schedules.

(C) Locking cannot take place at field level.

(D) An exclusive lock on data item X is granted even if a shared lock is already held on X.

Ans: (A)

Q.44 Which of the following statement on the view concept in SQL is invalid?

(A) All views are not updateable

(B) The views may be referenced in an SQL statement whenever tables are referenced.

(C) The views are instantiated at the time they are referenced and not when they are defined.

(D) The definition of a view should not have GROUP BY clause in it.

Ans: (D)

Q.45 Which of the following concurrency control schemes is not based on the serializability property?

(A) Two – phase locking **(B)** Graph-based locking

(C) Time-stamp based locking **(D)** None of these .

Ans: (D)

Q.46 Which of the following is a reason to model data?

(A) Understand each user's perspective of data

(B) Understand the data itself irrespective of the physical representation

(C) Understand the use of data across application areas

(D) All of the above

Ans: (D)

Q.47 If an entity can belong to only one lower level entity then the constraint is

(A) disjoint **(B)** partial

(C) overlapping **(D)** single

Ans: (B)

Q.48 The common column is eliminated in

(A) theta join **(B)** outer join

(C) natural join **(D)** composed join

Ans: (C)

Q.49 In SQL, testing whether a subquery is empty is done using
(A) DISTINCT **(B)** UNIQUE
(C) NULL **(D)** EXISTS

Ans: (D)

Q.50 Use of UNIQUE while defining an attribute of a table in SQL means that the attribute values are
(A) distinct values **(B)** cannot have NULL
(C) both **(A)** & **(B)** **(D)** same as primary key

Ans: (C)

Q.51 The cost of reading and writing temporary files while evaluating a query can be reduced by
(A) building indices **(B)** pipelining
(C) join ordering **(D)** none of the above

Ans: (B)

Q.52 A transaction is in _____ state after the final statement has been executed.
(A) partially committed **(B)** active
(C) committed **(D)** none of the above

Ans: (C)

Q.53 In multiple granularity of locks SIX lock is compatible with
(A) IX **(B)** IS
(C) S **(D)** SIX

Ans: (B)

Q.54 The statement that is executed automatically by the system as a side effect of the modification of the database is
(A) backup **(B)** assertion
(C) recovery **(D)** trigger

Ans: (D)

Q.55 The normal form that is not necessarily dependency preserving is
(A) 2NF **(B)** 3NF
(C) BCNF **(D)** 4NF

Ans: (A)

Q.56 A functional dependency of the form $x \rightarrow y$ is trivial if

(A) $y \subseteq x$ **(B)** $y \subset x$
(C) $x \subseteq y$ **(D)** $x \subset y$

Ans: (A)

Q.57 The normalization was first proposed by _____.
(A) Code **(B)** Codd
(C) Boyce Codd **(D)** Boyce

Ans: (B)

Q.58 The division operator divides a dividend A of degree m+n by a divisor relation B of degree n and produces a result of degree
(A) m – 1 **(B)** m + 1
(C) m * m **(D)** m

Ans: (D)

Q.59 Which of the following is not a characteristic of a relational database model?
(A) Table **(B)** Tree like structure
(C) Complex logical relationship **(D)** Records

Ans: (B)

Q.60 Assume transaction A holds a shared lock R. If transaction B also requests for a shared lock on R.

(A) It will result in a deadlock situation.
(B) It will immediately be rejected.
(C) It will immediately be granted.
(D) It will be granted as soon as it is released by A .

Ans: (C)

Q.61 In E-R Diagram total participation is represented by

(A) double lines **(B)** Dashed lines
(C) single line **(D)** Triangle

Ans: (A)

Q.62 The FD $A \rightarrow B$, $DB \rightarrow C$ implies
(A) $DA \rightarrow C$ **(B)** $A \rightarrow C$
(C) $B \rightarrow A$ **(D)** $DB \rightarrow A$

Ans: (A)

Q.63 The graphical representation of a query is _____.
(A) B-Tree **(B)** graph
(C) Query Tree **(D)** directed graph

Ans: (C)

Q.64 Union operator is a :

(A) Unary Operator **(B)** Ternary Operator
(C) Binary Operator **(D)** Not an operator

Ans: (C)

Q.65 Relations produced from an E-R model will always be

(A) First normal form. **(B)** Second normal form.
(C) Third normal form. **(D)** Fourth normal form.

Ans: (A)

Q.66 Manager salary details are hidden from the employee .This is

(A) Conceptual level data hiding.
(B) External level data hiding.
(C) Physical level data hiding.
(D) None of these.

Ans: (A)

Q.67 Which of the following is true for network structure?

(A) It is a physical representation of the data.
(B) It allows many to many relationship.
(C) It is conceptually simple.
(D) It will be the dominant database of the future.

Ans: **(A)**

Q.68 Which two files are used during operation of the DBMS?

(A) Query languages and utilities
(B) DML and query language
(C) Data dictionary and transaction log
(D) Data dictionary and query language

Ans: (C)

Q.69 A list consists of last names, first names, addresses and pin codes. If all people in the list have the same last name and same pin code a useful key would be

(A) the pin code
(B) the last name
(C) the compound key first name and last name
(D) Tr from next page

Ans: (C)

Q.70 In b-tree the number of keys in each node is ____ than the number of its children.

(A) one less **(B)** same
(C) one more **(D)** half

Ans: (A)

Q.71 The drawback of shadow paging technique are

(A) Commit overhead **(B)** Data fragmentation
(C) Garbage collection **(D)** All of these

Ans: (D)

Q.72 Which normal form is considered adequate for normal relational database design?

(A) 2NF **(B)** 5NF
(C) 4NF **(D)** 3NF

Ans: (D)

Q.73 Which of the following addressing modes permits relocation without any change over in the code?

(A) Indirect addressing **(B)** Indexed addressing
(C) PC relative addressing **(D)** Base register addressing

Ans: (B)

Q.74 In a multi-user database, if two users wish to update the same record at the same time, they are prevented from doing so by

(A) jamming **(B)** password
(C) documentation **(D)** record lock

Ans: (D)

Q.75 The values of the attribute describes a particular_____

(A) Entity set **(B)** File
(C) Entity instance **(D)** Organization

Ans: (C)

Q.76 Which of the following relational algebraic operations is not from set theory?

(A) Union **(B)** Intersection
(C) Cartesian Product **(D)** Select

Ans: (D)

Q.77 Which of the following ensures the atomicity of the transaction?

(A) Transaction management component of DBMS
(B) Application Programmer
(C) Concurrency control component of DBMS
(D) Recovery management component of DBMS

Ans: (A)

Q.78 If both the functional dependencies : $X \rightarrow Y$ and $Y \rightarrow X$ hold for two attributes X and Y then the relationship between X and Y is

(A) M:N **(B)** M:1
(C) 1:1 **(D)** 1:M

Ans: (C)

Q.79 What will be the number of columns and rows respectively obtained for the operation, A-B, if A B are Base union compatible and all the rows of a are common to B? Assume A has 4 columns and 10 rows; and B has 4 columns and 15 rows
(A) 4,0 **(B)** 0,0
(C) 4,5 **(D)** 8,5

Ans: (A)

Q.80 For correct behaviour during recovery, undo and redo operation must be
(A) Commutative **(B)** Associative
(C) idempotent **(D)** distributive

Ans: (C)

Q.81 Which of the following is not a consequence of non-normalized database?
(A) Update Anomaly **(B)** Insertion Anomaly
(C) Redundancy **(D)** Lost update problem

Ans: (D)

Q.82 Which of the following is true for relational calculus?
(A) $\forall x(P(x)) \equiv \neg(\exists x)(\neg P(x))$ **(B)** $\forall x(P(x)) \equiv \neg(\exists x)(P(x))$
(C) $\forall x(P(x)) \equiv (\exists x)(\neg P(x))$ **(D)** $\forall x(P(x)) \equiv (\exists x)(P(x))$

Ans: (A)

Q.83 The part of a database management system which ensures that the data remains in a consistent state is
(A) authorization and integrity manager
(B) buffer manager
(C) transaction manager
(D) file manager

Ans: (C)

Q.84 Relationships among relationships can be represented in an-E-R model using
(A) Aggregation **(B)** Association
(C) Weak entity sets **(D)** Weak relationship sets

Ans: (A)

Q.85 In tuple relational calculus P1 AND P2 is equivalent to
(A) $(\neg P1 OR \neg P2)$. **(B)** $\neg(P1 OR \neg P2)$.
(C) $\neg(\neg P1 OR\ P2)$. **(D)** $\neg(\neg P1 OR\ \neg P2)$.

Ans: (D)

Q.86 If $\alpha \rightarrow \beta$ holds then so does

(A) $\gamma\alpha \rightarrow \gamma\beta$ **(B)** $\alpha \rightarrow \rightarrow \gamma\beta$

(C) both (A) and (B) **(D)** None of the above

Ans: (A)

Q.87 Cascading rollback is avoided in all protocol except

(A) strict two-phase locking protocol.

(B) tree locking protocol

(C) two-phase locking protocol

(D) validation based protocol.

Ans: (D)

Q. 88 Wait-for graph is used for

(A) detecting view serializability. **(B)** detecting conflict serializability.

(C) deadlock prevention **(D)** deadlock detection

Ans: (D)

Q.89 The expression $\sigma_{\theta 1}(E1 \bowtie_{\theta 2} E2)$ is the same as

(A) $E1 \bowtie_{\theta 1 \wedge \theta 2} E2$ **(B)** $\sigma_{\theta 1}\, E1 \wedge \sigma_{\theta 2}\, E2$

(C) $E1 \bowtie_{\theta 1 \vee \theta 2} E2$ **(D)** None of the above

Ans: (A)

Q.90 The clause **alter table** in SQL can be used to

(A) add an attribute

(B) delete an attribute

(C) alter the default values of an attribute

(D) all of the above

Ans: (D)

Q. 91 The data models defined by ANSI/SPARC architecture are

(A) Conceptual, physical and internal

(B) Conceptual, view and external

(C) Logical, physical and internal

(D) Logical, physical and view

Ans: (D)

Q.92 Whenever two independent one-to-many relationships are mixed in the same relation, a _____ arises.

(A) Functional dependency **(B)** Multi-valued dependency

(C) Transitive dependency **(D)** Partial dependency

Ans:(B)

Q.93 A table can have only one

(A) Secondary key **(B)** Alternate key
(C) Unique key **(D)** Primary key

Ans: (D)

Q.94 Dependency preservation is not guaranteed in

(A) BCNF **(B)** 3NF
(C) PJNF **(D)** DKNF

Ans: (A)

Q.95 Which is the best file organization when data is frequently added or deleted from a file?

(A) Sequential **(B)** Direct
(C) Index sequential **(D)** None of the above

Ans: (B)

Q.96 Which of the following constitutes a basic set of operations for manipulating relational data?

(A) Predicate calculus **(B)** Relational calculus
(C) Relational algebra **(D)** SQL
Ans: (C)

Q.97 An advantage of views is

(A) Data security **(B)** Derived columns
(C) Hiding of complex queries **(D)** All of the above

Ans: (A)

Q.98 Which of the following is not a recovery technique?

(A) Deferred update **(B)** Immediate update
(C) Two-phase commit **(D)** Shadow paging

Ans: (C)

Q.99 Isolation of the transactions is ensured by

(A) Transaction management **(B)** Application programmer
(C) Concurrency control **(D)** Recovery management

Ans: (C)

Q.100 _____ operator is used to compare a value to a list of literals values that have been specified.

(A) Like **(B)** COMPARE
(C) BETWEEN **(D)** IN
Ans: (A)

PART- II

DESCRIPTIVES

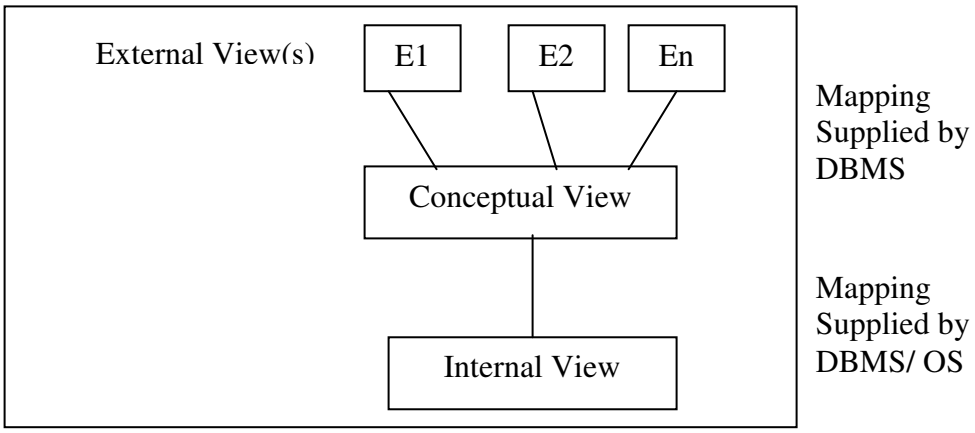
Q.1 Briefly discuss the different layers of ANSI SPARC architecture. Define physical and logical data independence. How does this architecture help in achieving these? **(7)**

Ans: The three layers of ANSI SPARC architecture are as follows:

- **Internal view** is at the lowest level of abstraction, closest to the physical storage method used. It indicates how the data will be stored and describes the data structures and access methods to be used by the database. There is one internal view for the entire database.
- **Global or Conceptual View :** At this level of abstraction all the database entities and the relationships among them are included. There is one conceptual view for the entire database.
- **External or User View:** The external or user view is at the highest level of database abstraction where only those portions of the database concern to a user or application programme are included. Any number if external or user views may exists for a given global or conceptual view.

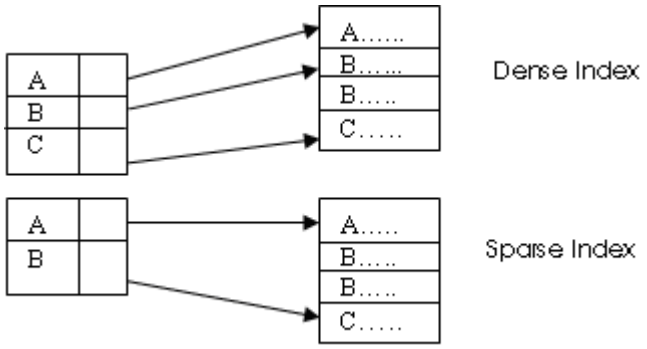
Data independence implies that change in one view must not require a change in the view(s) above. There are two types of data independence : logical and physical. **Logical data independence** means that the conceptual view can be changed without effecting the existing external view, i.e., a given record may be spilt or combined with other records but the external views need not be changed to reflect this. Physical data independence means that the physical storage structures or devices used to store the data can be changed without effecting the existing conceptual view or external view, i.e., if earlier indexed sequential files are used to store data and then the B- trees are used, even then the upper layers should not be effected.

There are two different mappings between the layers as shown below in the diagram. The mapping between external and conceptual levels is responsible to provide logical data independence and the mapping between internal and conceptual levels is responsible to provide physical data independence.



Q.2 Describe the storage structure of indexed sequential files and their access method. **(7)**

Ans: Storage structure of Indexed Sequential files and their access : To gain fast random access to records in a file, we can use an index structure. An index record consists of a search key value and pointers to data records, which is associated with a particular search key. An ordered index stores the values of the search keys in sorted order. A file may have several indices on different search keys. If the files containing the records is sequentially ordered, a primary index is an index whose search key also defines the sequential order of the file. Such files are known as index sequential files. There are two types of ordered indices : dense and sparse. In dense index, and index record appears only for some of the search-key in the files as shown below.



To access a particular record with search key value, K, using dense index we search index record with search key value, K, from which reach the first entry of data record with search key value K. Then the data records are searched linearly to obtain the required record. If either the index record is not there or the linear search reaches the data record with different search key value, then the record is not there. Now for sparse index, we search index record with search key value, K or the index record with highest search key value less than K, from which reach the first entry of data record with search key value K or highest value less than K. Then the data records are searched linearly to obtain the required record. If the linear searches the data record with search key value greater than K, then the record is not there.

Q.3 Define the terms entity, attribute, role and relationship between the entities, giving examples for each of them. **(4)**

Ans: Entity: An entity is a "thing" or "object" in the real world that is distinguishable from other objects. For example, a person and bank account can be considered as entities.
Attribute: Entities are described in a database by a set of attributes, i.e., the characteristics of an entity are known as attributes. For example, name, age, date of birth, etc are attributes of the entity person. Similarly, account number, balance, nature of account, etc are attributes of the entity bank account.
Relationship: A relationship is an association among the several entities. For example, a depositor relationship associates the entity person with a bank account.

Role: The function that an entity plays in a relationship is called that entity's role. For example, in the relationship depositor mentioned above, the entity person plays the role of a customer in the relationship.

Q.4 What are the three data anomalies that are likely to occur as a result of data redundancy? Can data redundancy be completely eliminated in database approach? Why or why not? **(5)**

Ans: The three type of anomalies that can arise in the database because of redundancy are insertion, deletion and modification/updation anomalies. Consider a relation emp_dept with attributes: E#, Ename, Address, D#, Dname, Dmgr# with the primary key as E#.

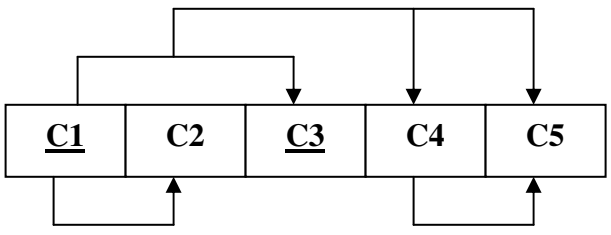
Insertion anomaly: Let us assume that a new department has been started by the organization but initially there is no employee appointed for that department, then the tuple for this department cannot be inserted into this table as the E# will have NULL, which is not allowed as E# is primary key. This kind of a problem in the relation where some tuple cannot be inserted is known as insertion anomaly.

Deletion anomaly: Now consider there is only one employee in some department and that employee leaves the organization, then the tuple of that employee has to be deleted from the table, but in addition to that the information about the department also will get deleted. This kind of a problem in the relation where deletion of some tuples can lead to loss of some other data not intended to be removed is known as deletion anomaly.

Modification /update anomaly: Suppose the manager of a department has changed, this requires that the Dmgr# in all the tuples corresponding to that department must be changed to reflect the new status. If we fail to update all the tuples of the given department, then two different records of employee working in the same department might show different Dmgr# leading to inconsistency in the database. This is known as modification/update anomaly.

The data redundancy. Cannot be totally removed from the database, but there should be controlled redundancy, for example, consider a relation student_report(S#, Sname, Course#, SubjectName, marks) to store the marks of a student for a course having some optional subjects, but all the students might not select the same optional papers. Now the student name appears in every tuple, which is redundant and we can have two tables as students(S#, Sname, CourseName) and Report(S#, SubjectName, Marks). However, if we want to print the mark-sheet for every student using these tables then a join operation, which is a costly operation, in terms of resources required to carry out, has to be performed in order to get the name of the student. So to save on the resource utilization, we might opt to store a single relation, students_report only.

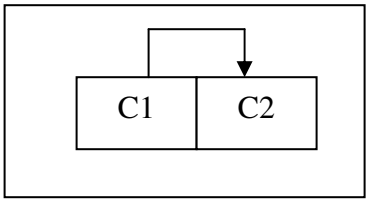
Q.5 Given the dependency diagram shown in the following figure, (the primary key attributes are underlined)



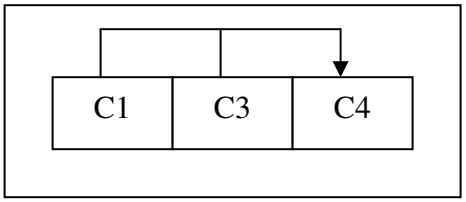
(i) Identify and discuss each of the indicated dependencies?
(ii) Create a database whose tables are atleast in 3NF, showing dependency Diagram for each table? **(4+5)**

Ans:
(i) The *FDS* for the given relation are:
1. C1→ C2
2. C1, C3 → C2, C4, C5
3. C4 → C5

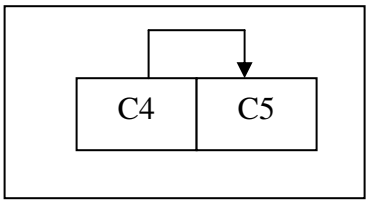
The Primary key of the given relation is (C1, C3), so C1 → C2 is a partial *FD* and there is a transitive *FD* also : C1, C3 → C4 →C5 in the relation.
(ii) According to the algorithm to decompose the relation schema into 3*NF* as follows:
R1 (C1, C2) with *FD* {C1 → C2} and functional dependency diagram.



R2(C1, C3, C4) with *FD* {C1, C3 → C4} and functional dependency diagram



R3(C4, C5) with *FD* {C4 → C5} and functional dependency diagram.



We cannot have a single relation for the *FD* C1, C3 → C2, C4, C5 as then there will be partial and transitive *FD*s in the relation.

Q.6 Consider the following relations with underlined primary keys.
Product(P_code, Description, Stocking_date, QtyOnHand, MinQty, Price, Discount, V_code)
Vendor(V_code, Name, Address, Phone)
Here a vendor can supply more than one product but a product is supplied by only one vendor. Write SQL queries for the following :
(i) List the names of all the vendors who supply more than one product.
(ii) List the details of the products whose prices exceed the average product price.

(iii) List the Name, Address and Phone of the vendors who are currently not supplying any product. **(3 x 3)**

Ans:

(i) Select Name from Vendor
Where V_code in (Select V_code from Product
group by V_code having count (V_code) > 1)

(ii) Select * from Product
Where Price > (Select avg (Price)from Product)

(iii) Select Name, Address, Phone From Vendor
Where V_code not in (Select V_code from Product)

Q.7 Define the domain relational calculus. **(5)**

Ans: An expression in the domain relational calculus is of the form
$\{< x_1, x_2, \ldots\ldots x_n > 1\ P(x_1, x_2 \ldots.., x_n)\} \ldots\ldots\ldots\ldots 1$ Where $x_1, x_2 \ldots\ldots\ldots x_n$ represent the domain variables. *P* represents a formula composed of atoms. An atom in the domain relational calculas has one of the following forms:

- $\{< x_1, x_2), \ldots.. x_n > \in r$, where r is a relation on *n* attributes and $x_1, x_2, \ldots.. x_n$ are domain variables or domain constants.
- x Θ y, where a and y where r domain variables and Θ is a comparison operator (<, ≤, =, ≠, ≥, >). We require that attributes x and y have domains that can be compared by Θ.
- x Θ c, where x is a domain variable, Θ is a comparison operator, and c is a constant in the domain of the attribute for which x is a domain variable.

We build up formulae from atoms by using the following rules :

- An atom is a formula.
- If P_1 is a formula, then so are $\neg p_1$ and (P_1)

If P_1 and P_2 are formulae, then so are $p_1 \Lambda\ p_2$ and $p_1 \vee p_2$ and $p_1 \Rightarrow P_2$

- If p_1 (x) is a formula in x, where x is a domain variable, then so are $\exists x(p_1(x))$ and $\forall x(p_1(x))$.

Q.8 Given $R(A, B, C, D, E)$ with the set of FDs, $F\{AB \rightarrow CD, A \rightarrow E, C \rightarrow D\}$. Is the decomposition of R into $R_1(A, B, C), R_2(B, C, D)$ and $R_3(C, D, E)$ lossless? Prove. **(5)**

Ans: To find whether the decomposition of R(A, B,C,D,E) with the set of functional dependencies, F={AB → CD, A → E, C → D} into R_1(A,B,C) R_2(B,C, D) and R_3(C,D,E) is lossless or not, we have the following initial table.

	A	B	C	D	E
R1	α_A	α_B	α_C	α_{1D}	β_{1E}
R2	β_{2A}	α_B	α_C	α_D	β_{2E}
R3	β_{3A}	β_{3B}	α_C	α_D	α_E

There is no change in table above for the functional dependencies AB → CD and A →E, but for the functional dependency C → D, all the rows of C have α_C so and the corresponding rows of D can be updated to α_D as there is an α_D in two of such rows. Here the new table is as follows:

	A	B	C	D	E
R1	α_A	α_B	α_C	α_D	β_{1E}
R2	β_{2A}	α_B	α_C	α_D	β_{2E}
R3	β_{3A}	β_{3B}	α_C	α_D	α_E

No change can be made in the table further and from this table we can see that there is no row with all the α's, hence the decomposition is a lossy one.

Q.9 Given R(A,B,C,D,E) with the set of FDs,
F{AB→CD, ABC → E, C → A}
(i) Find any two candidate keys of R
(ii) What is the normal form of R? Justify. **(9)**

Ans:

(i) To find two candidate keys of R, we have to find the closure of the set of attributes under consideration and if all the attributes of R are in the closure then that set is a candidate key. Now from the set of FD's we can make out that B is not occurring on the RHS of any FD, therefore, it must be a part of the candidate keys being considered otherwise it will not be in the closure of any attribute set. So let us consider the following sets AB and BC.

Now $(AB)^+$= ABCDE, CD are included in closure because of the FD AB →CD, and E is included in closure because of the FD ABC → E.

Now $(BC)^+$= BCAED, A is included in closure because of the FD C → A, and then E is included in closure because of the FD ABC → E and lastly D is included in closure because of the FD AB → CD.

Therefore two candidate keys are : AB and BC.

(ii) The prime attributes are A, B and C and non-prime attributes are D and E.

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 (D,E) 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^+ of the form X → A, either X is a superkey or A is prime. From the set of FDs we see that for all the FDs, this is satisfied, therefore, the relation is in 3NF.

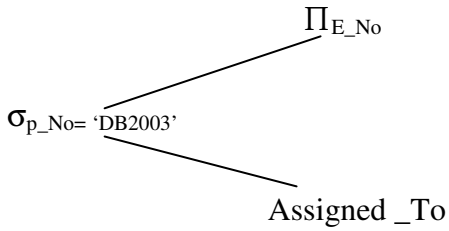
A relation scheme is in BCNF, if for all the non-trivial FDs in F^+ of the form X→ A, X is a superkey. From the set of FDs we can see that for the FD C → A, this is not satisfied as LHS is not a superkey, therefore, the relation is not in BCNF.

Hence, the given relation scheme is in 3NF.

Q.10 How does a query tree represent a relational algebra expression? Discuss any three rules for query optimisation, giving example as to when should each rule be applied. **(8)**

Ans: A query tree is a tree structure that corresponds to a relational algebra expression. It represents the input relations 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. The execution terminates when the root node is executed and produces the result relation for the query. For example, the query tree for the relational algebra expression

$\prod_{E\_No} (\sigma_{p\_No = \text{'DB2003'}} (\text{Assigned\_To}))$:



The three rules for query optımızatıon are as follows:

- In a cascade (sequence) of $\prod$ operations, all but the last one can be ignored :
  $\prod_1(\prod_2(\ldots\ldots\prod_n (R))) \equiv \prod_1(R)$
- If a selection condition c involves only those attributes A1, A2, …. An in the projection list, the two operations can be commuted :
  $\prod_{A1, A2, \ldots\ldots An} (\sigma_C (R) \equiv \sigma_C (\prod_{A1, A2, \ldots\ldots An} (R)) \ldots\ldots))$
- A conjunctive selection condition can be broken up into a cascade of individual σ operations :
  $\sigma_{C1 \text{ AND } C2 \text{ AND } \ldots\ldots \text{AND } Cn} (R) \equiv \sigma_{C1} (\sigma_{C2}(\ldots.(\sigma_{Cn} (R))\ldots.))$

Q. 11 Consider the following database with primary keys underlined

Project(<u>P_No</u>, P_Name, P_Incharge)

Employee(<u>E_No</u>, E_Name)

Assigned_To(<u>P_No</u>, <u>E_No</u>)

Write the relational algebra for the following :

(i) List details of the employees working on all the projects.
(ii) List E_No of employees who do not work on project number DB2003. **(6)**

Ans:

(i) Employee $\bowtie$(Assigned_To) $\div \prod_{p\_no}$ (Project))
(ii) $\prod_{E\_No}$(Assigned_To) - $\prod_{E\_No}(\sigma_{p\_No = \text{'DB2003'}}$ (Assigned _To))

Q.12 What is a hashing function? What are the properties of a good hashing function? Describe the folding technique for hashing functions. **(7)**

Ans: Hashing function is a technique to store a file on the disk. A hash function is a function which when applied to a value of a record (hash field) gives a hash value, which is the address of the disk block in which the record is stored.

The properties of a good hash function are as follows:

- It should be easy to evaluate.
- The hash value obtained should be within the range of valid addresses for a given file.
- The hash values should be uniformly distributed over the range of addresses so that the collisions are minimum.

Folding technique for hashing functions: It involves applying an arithmetic function such as addition or a logical function like exclusive OR to different parts of a hash field to calculate the hash addresses. For example, the numeric hash field can be split into start, middle and end regions, such that the sum of the lengths of the start and end regions equals the length of the middle region. The start, middle and end regions' digits are added to get a new value, x. Then x mod s, where s is the upper limit for the hash function, gives the hash value.

Q.13 Describe the problems of lost update, inconsistent read and phantom phenomenon which arise as a result of concurrency. **(7)**

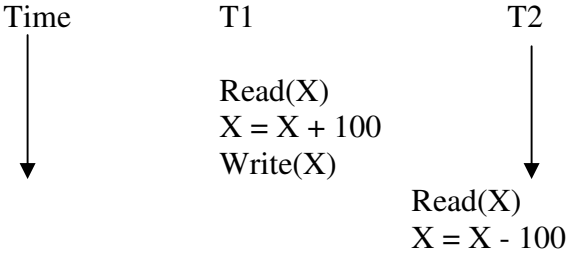
Ans: **Lost update problem:** The problem occurs when two transactions that access the same database items have their operations interleaved in a way that makes the value of some database item incorrect. For example, consider the following interleaved schedule of two transactions:

Time	T1	T2
	Read(X)	
	X = X+ 100	
		Read(X)
		X = X – 100
	Write(X)	
		Write(X)

From the schedule, it can be seen that the updation of x by T1 will be overwritten by T2. If these transactions execute in serial order then the value of x should not change, but with this schedule the value of x will be reduced by 100, which is incorrect.

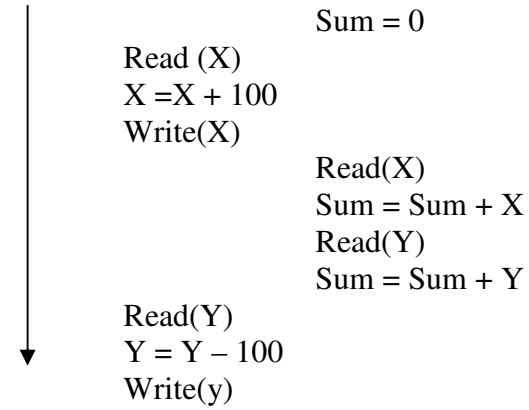
Inconsistent Read or Temporary Update or Dirty Read problem: problem occurs when one transaction updates a database item and then the transaction fails for some reason. The updated value is accessed by another transaction before it is changed back to

the original value. For example, consider the following interleaved schedule of two transactions :



From the schedule, it can be seen that T2 will read the value written by T1, however if T1 fails for some reason before commit then T2 will use the value which is not valid.

Phantom Phenomenon : If a transaction is calculated an aggregate summary function on a number of records while other transactions are updating some of records, the aggregate function may calculate some values before they are updated and others after they are updated. For example, consider the following interleaved schedule of two transactions:



From the schedule, it can be seen that T2 will read the new value of X written by T1 but not that of Y, so the value of Sum will be off by 100 from the actual valid value.

Q.14 Define

(i) Shared locks.
(ii) Serializable schedule.
(iii) Thomas write rule.
(iv) Two phase commit. **(4)**

Ans:

(i) **Shared lock:** During concurrent execution of transactions, before a transaction can access a data item, it has to acquire a lock on it. Now if the transactions only want to read the data item then every transaction can lock the data item in shared mode, such a lock is known as shared lock.

(ii) **Serializable schedule:** An interleaved schedule of more than one transactions is called a serializable schedule, if it is equivalent to some serial schedule of those transactions.

(iii) **Thomas' write rule:** The Thomas' write rule is a modification of timestamp-ordering protocol for concurrency control. Suppose that transaction T_i issues write (Q) and $TS(T_i)$ is its timestamp then the following actions are taken depending on $TS(T_i)$ and the read-write timestamps of the data item Q:

- If $TS(T_i)$<R-timestamp (Q), then the value of Q that T_i is producing was previously needed, and it had been assumed that the value would never be produced. Hence, the system rejects the write operation and rolls T_i back.
- If $TS(T_i)$ < W-timestamp (Q), then T_i is attempting to write an obsolete value of Q. Hence, this write operation can be ignored otherwise, the system executes the write operation and sets W-timestamp(Q) to $Ts(T_i)$

(iv) Two phase commit: To ensure atomicity, all the sites in which a transaction is being executed must agree on the final outcome of the execution. The transaction must either commit at all sites or it must abort at all sites. For this simplest protocol is two-phase commit, which has the voting phase and the decision phase, In the voting phase the sub-transactions are requested to vote on their readiness to commit or abort. In the decision phase the decision as to whether all sub-transactions should commit or abort is made and carried out. The transactions at a site interact with the transaction manager of the site, cooperating in the exchange of messages through which the two –phase commit is executed.

Q.15 Let transactions T1, T2 and T3 be defined to perform the following operations :

T1 : Add one to A
T2 : Double A
T3 : Display A on the screen and then set A to one.
(where A is some item in the database)

Suppose transactions T1, T2 and T3 are allowed to execute concurrently. If A has initial value zero, how many possible correct results are there? Enumerate them. **(10)**

Ans: The following three transactions, manipulating the contents of A, are to execute concurrently:
T1 : Add one to A ie. A = A + 1
T2 : Double A ie. A + 2 * A
T3 : Display A on screen and then set it to 1 ie. A = 1
Assuming A has an initial value zero. Now three transactions can execute concurrently in the following different ways. The tables are showing the different transaction schedules along with the changes they make in the value of A and the value of A displayed by T3.

Transactions Schedule	Current value of A	Display
T1	1	
T2	2	
T3	1	2

Transactions Schedule	Current value of A	Display
T1	1	
T3	1	1
T2	2	

Transactions Schedule	Current value of A	Display
T2	0	
T1	1	
T3	1	1

Transactions Schedule	Current value of A	Display
T2	0	
T3	1	0
T1	2	

Transactions Schedule	Current value of A	Display
T3	1	0
T1	2	
T2	4	

Transactions Schedule	Current value of A	Display
T3	1	0
T2	2	
T1	3	

Q.16 Define and differentiate between the following :-

(i) Deadlock prevention.

(ii) Deadlock detection.

(iii) Deadlock avoidance. **(6)**

Ans:

(i) Deadlock prevention: These protocols ensure that the system will never enter a deadlock state. There are two approaches to deadlock prevention. One approach ensures that no cyclic waits can occur by ordering the requests for locks, or requiring all locks to be acquired together. The other approach is closer to deadlock recovery, and performs transaction rollback instead of waiting for a lock, whenever the wait could potentially result in a deadlock. In this approach two timestamp based techniques are there : wait – die and wound – wait . In wait –die, the older transaction is allowed to wait if it needs data from older transaction. In wound - wait, the younger transaction is rolled back if it needs data from older transaction if older transaction needs data currently held by a younger transaction, however younger transaction is allowed wait if it needs data from older transaction.

(ii) Deadlock detection: If a system does not employ some protocol that ensures deadlock freedom, then a detection and recovery scheme must be used. An algorithm that examines the state of the system is invoked periodically to determine whether a deadlock has occurred. If one has, then the system must attempt to recover from the deadlock.

(iii) Deadlock avoidance : These protocols also ensure that the system will never enter a deadlock state but the way it is done is different from deadlock prevention. In this whenever a transaction requests for some data, the system consider the resources currently allocated to each process and the future requests and releases of each process currently allocated to each process and the future requests and releases of each process, to decide whether the current request can be satisfied or must wait to avoid a possible future deadlock. In this the transactions have to give additional information about how the data will be requested in future.

Q.17 Write short notes on any FOUR of the following:

(i) Two phase locking protocol.
(ii) Audit Trails.
(iii) Query Processing.
(iv) Disadvantages of file based systems.
(v) Query-by-Example.
(vi) B-tree. **(3.5×4=14)**

Ans:

(i) Two Phase Locking Protocol : This is a protocol which is used to ensure serializability of transactions. This protocol requires that each transaction issue lock and unlock requests in two phases :

Growing Phase : A transaction may obtain locks, but may not release any lock.

Shrinking Phase: A transaction may release locks, but may not obtain any new locks. Initially, a transaction is in growing phase. The transaction acquires locks as needed. Once the transaction releases a lock, it enters the shrinking phase, and it can issue no more lock requests. Two-phase locking ensures conflict serializability, but does not ensure freedom from deadlock.

(ii) Audits trails : Audit trail is maintained by the databases for security. An audit trail is a log of all changes (inserts /deletes /updates) to the database, along with information such as which user performed the change and when the change was performed. The audit trails help in security in several ways. For example, if the balance on an account is found to be incorrect, the bank may wish to trace all the updates performed on the account, to find out incorrect (or fraudulent) updates, as well as the persons who carried out the updates. The bank could then also use the audit trails to trace all the updates performed by these persons, in order to find other incorrect or fraudulent updates.

An audit trail is a series of records of computer events, about an operating system, an application, or user activities. It is generated by an auditing system that monitors system activity. Audit trails have many uses in the realm of computer security :

Individual Accountability : An individual's actions are tracked in an audit trail allowing users to be personally accountable for their actions. This deters the users from circumventing security policies. Even if they do, they can be held accountable.

Reconstructing Events : Audit trails can also be used to reconstruct events after a problem has occurred. The amount of damage that occurred with an incident can be assessed by reviewing audit trails of system activity to pinpoint how, when, and why the incident occurred.

Problem Monitoring : Audit trails may also be used as on-line tools to help monitor problems as they occur. Such real time monitoring helps in detection of frauds etc.

Intrusion Detection : Intrusion detection refers to the process of identifying attempts to penetrate a system and gain unauthorized access. Audit trails can help in intrusion detection if they record appropriate events. Determining what events to audit so that audit trails can be used in an effective manner to aid intrusion detection is one of the present research issues being looked into by the research community.

(iii) QueryProcessing: query processing is the procedure of selecting the best plan or strategy to be used in responding to a database request. The plan is then executed to generate a response. The component of the DBMS responsible for generating this strategy is called a query processor. It is a stepwise process. The first step is to transform the query into a standard from. For example, a query in QBE looked into by the research community. **Query** may be transformed into a relational algebraic expression, the

optimization is performed by substituting equivalent expressions for those in the query. In the next step a number of strategies called access pans are generated for evaluating the transformed query. The physical characteristics of the data and any supporting access method are taken into account in generating alternate access pans. The cost of each access plan is estimated and the optimal one is chosen and executed.

(iv) Disadvantages of file based systems : Some of the disadvantages of file based systems are as follows:

- **Data redundancy and inconsistency :** the different application programs may require the same data but may store it in separate files to be used by them only. For example, in a college environment, the address, contact no. etc., may be maintained both by the office staff and the library staff to sent appropriate reminders to the fee defaulters or to the students who have not returned the books on due dates, respectively. If a student changes the residence and that is reflected in only one of the system then it leads to inconsistency.
- **Difficulty in accessing data :** Even if the data is there in the file but if a new type of details are required from the data, they cannot be retrieved unless and until a new application is written for it.
- **Data isolation :** Because data are scattered in various files, and files may be in different formats, writing new application programs to retrieve the appropriate data is difficult.
- **Integrity problems:** The integrity constraint can be applied through the application program, but not directly to the data file. For example, there is a constraint to insure that the balance of an account must not be below Rs. 1000, then through application program this can be done but someone may directly make the changes to the data file and violate this constraint.

(v) Query-By-Example (QBE) : QBE is a query language based on domain calculus and has two dimensional syntax. The queries are written in the horizontal and vertical dimensions of table. Queries are formed by entering an example of a possible answer in a skeleton table as shown in the following diagram:

Relation on Name	**Attribute_name**	**Atrtribute_name 2**		**Attribute_name**
Tuple Operations	Domain var., Constants and Predicates	Domain Var., Constants and predicates	…… ……..	Domain Var., Constants and predicates

Conditions specified in a single row are ANDed and conditions specified in separate rows are ORed. The tuple operations are P., I ., etc., for PRINT, insert commands, resp. The variable names are specified by preceding their name with an underscore. QBE also provides a "conditions" box to specify additional constraints so that the conditions that cannot be added in a Skelton table can be entered in the conditions table. The aggregation operations like max., min, sum. etc. are also provided by QBE.

(vi) B-tree : To gain fast random access to record in a file, we can use an index structure. B-trees take the form of a balanced tree in which every path from the root of the tree to a leaf of the tree is of same length. Each nonleaf node in the tree has between *n/2* and *n* children, where *n* is fixed for a particular tree. A generalized leaf (a) and nonleaf (b) nodes of a B-tree are shown below:

P1	K1	P2	…….	P_{n-1}	K_{n-1}	P_n

(a)

P_1	B_1	K_1	P_2	B_2	K_2	…..	P_{m-1}	B_{m-1}	K_{m-1}	P_m

(b)

In the leaf nodes, for i =1,2,….n-1, pointer P_i points to either a file record with the search key value K_i or to a bucket of pointers, each of which points to a file record with search key value K_i . The bucket key structure is used only if the search key does not from a primary key, and if the file is not sorted on the search key value order. The pointer P_i is used to chain together the leaf nodes in the search key order. In nonleaf nodes, the pointer P_i are the tree pointers that are similar to leaf node pointers, P_i. The pointers *B*-tree, they are used to store the indices. How ever, a variation of them, B^+-trees, are nowadays more frequently used by the databases as they are more easier to implement.

Q.18 Differentiate between

(i) Single value and multiple valued attribute.
(ii) Derived and non-derived attributes.
(iii) Candidate and super key.
(iv) Partial key and primary key.

(8)

Ans:

(i) Single Value and Multivalued attribute – Single value attribute has a single atomic value for an entity and multivalued attribute may have more than one value for an entity. For example, PreviousDegrees of a STUDENT.

(ii) Derived and Non-Derived Attribute – In some cases, two or more attribute values are related, for example, Age and BirthDate attributes of a person. For particular person entity, the value of Age can be determined from the current date and the value of that person's BirthDate. Hence, the attribute Age is called as derived attribute and the attribute BirthDate is called as non-derived or stored attribute.

(iii) Candidate Key and Super Key – A super key is a set of one or more attributes that, taken collectively, allows us to identify uniquely a tuple in the relation. While a candidate key is a minimal superkey, which can be used to uniquely identify a tuple in the relation. In superkey, there may some extra attributes.

(iv) Partial Key and Primary Key – A partial key, also called as discriminator, is the set of attributes that can uniquely identify weak entities that are related to the same owner entity. Primary key is a unique key but not have ***null values***. An entity set can have ***at most one primary key***.

Q.19 Define the basic operations of the relational algebra? **(4)**

Ans: Basic operators of relational algebra are:

1. **Union (∪) -** Selects tuples that are in either P or Q or in both of them. *The duplicate tuples are eliminated.*

$$R = P \cup Q$$

2. **Difference or Minus (-)** - Removes common tuples from the first relation.

$$R = P - Q$$

3. **Cartesian Product or Cross Product (×) -** The cartesian product of two relations is the concatenation of tuples belonging to the two relations and consisting of all possible combination of the tuples.

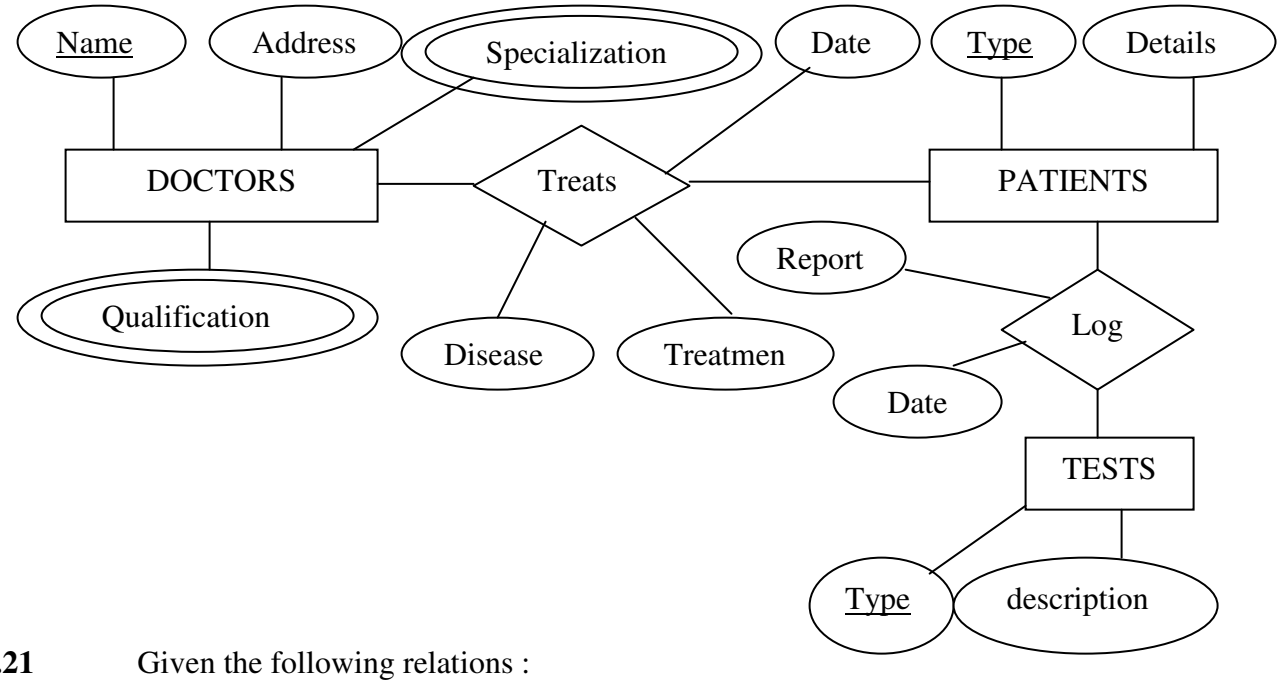
$$R = P \times Q$$

4. **Projection (π) -** The projection of a relation is defined as a projection of all its tuples over some set of attributes, i.e., it yields a ***vertical subset*** of the relation. The projection of a relation T on the attribute A is denoted by T[A] or $\pi_A(T)$,.
5. **Selection (σ) -** Selects only some of the tuples, those satisfy given criteria, from the relation. It yields a ***horizontal subset*** of a given relation.

$$R = \sigma_B(P)$$

Q.20 Construct an ER diagram for a hospital with a set of patients and a set of medical doctors. Associate with each patient a log of the various tests and examinations conducted. **(6)**

Ans:



Q.21 Given the following relations :

vehicle (reg_no, make, colour)
Person (eno, name, address)
Owner (eno, reg_no)

Write expressions in relational algebra to answer the following queries :

(i) List the names of persons who do not own any car.
(ii) List the names of persons who own only Maruti Cars. **(6)**

Ans:

(i) $\pi_{name}(PERSON \bowtie \pi_{eno}(PERSON) - \pi_{eno}(OWNER))$

(ii) $\pi_{name}(((\pi_{eno}(OWNER \bowtie \sigma_{make='maruti'}(VEHICLE))$

$- \pi_{eno}(OWNER \bowtie \sigma_{make\neq'maruti'}(VEHICLE))) \bowtie PERSON)$

Q.22 Define Join and Outer Join and differentiate between them. **(4)**

Ans: Join – It produces all the combinations of tuples from two relations that satisfy a join condition. **Outer Join** - If there are any values in one table that do not have corresponding value(s) in the other, in an equi-join that will not be selected. Such rows

can be forcefully selected by using the *outer join. The corresponding columns for that row will have NULLs.* There are actually three forms of the outer-join operation: left outer join (⟕), right outer join (⟖) and full outer join (⟗).

Q.23 Consider the following relations
Physician (rgno, phyname, addr, phno)
Patient (ptname, ptaddr)
Visits(rgno, ptname, dateofvisit, fees-charged)
Answer the following in SQL :

(i) Define the tables. Identify the keys and foreign keys.

(ii) Create an assertion that the total fees charged for a patient can not be more than Rs.1000/- assuming that patients can visit the same doctor more than once.

(iii) Create a view Patient_visits(name, times) where name is the name of the patient and times is the number of visits of a patient.

(iv) Display the ptname, ptaddr of the patient(s) who have visited more than one physician in the month of May 2000 in ascending order of ptname. **(3.5 x 4 = 14)**

Ans:

(i)
```
CREATE TABLE PHYSICIAN
( RGNO          VARCHAR2(5) PRIMARY KEY,
  PHYNAME       VARCHAR2(15),
  ADDR          VARCHAR2(25),
  PHNO          VARCHAR2(10));
CREATE TABLE PATIENT
(       PTNAME      VARCHAR2(15) PRIMARY KEY,
        PTADDR      VARCHAR2(25));
CREATE TABLE VISITS
( RGNO          VARCHAR2(5) REFERENCES PHYSICIAN(RGNO),
  PTNAME        VARCHAR2(15) REFERENCES, PATIENT(PTNAME),
  DATEOFVISIT   DATE,
  FEE-CHARGED   NUMBER(8,2),
  CONSTRAINT VISITS_PK PRIMARY KEY(RGNO, PTNAME));
```

(ii)
```
CREATE ASSERTION
CHECK ((SELECT SUM(FEES_CHARGED) FROM VISITS
WHERE PTNAME = :NEW.PTNAME AND RGNO = :NEW.RGNO) <= 1000)
```

(iii)
```
CREATE VIEW PATIENT_VISITS (NAME, TIMES) ASSELECT PTNAME,
  COUNT(PTNAME) FROM VISITS GROUP BY PTNAME;
```

(iv)
```
SELECT * FROM PATIENT
    WHERE PTNAME IN (SELECT PTNAME FROM VISITS A, VISITS B
    WHERE A.PTNAME = B.PTNAME AND A.RGNO <> B.RGNO
    AND A.DATEOFVISIT BETWEEN TO_DATE('01-MAY-2000') AND
    TO_DATE('31-MAY-2000')) ORDER BY PTNAME
```

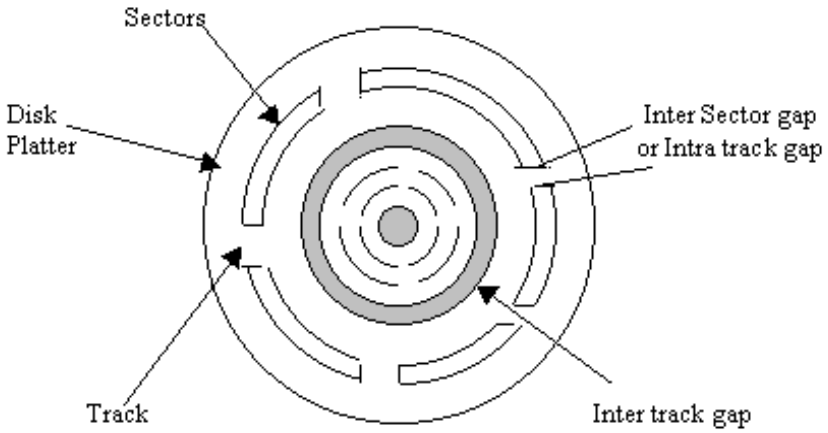
Q.24 What are the advantages of having an index structure? **(2)**

Ans: The index structures typically provide secondary access paths, which provide alternative ways of accessing the records without affecting the physical placement of records on disk. They enable efficient access to records based on the indexing fields that

are used to construct the index. Indexes are used to improve the efficiency of retrieval of records from a data file.

Q.25 What are the physical characteristics of magnetic disks? **(4)**

Ans: A magnetic disk is a circular plate or platter of plastic or metal which is coated with magnetizable material such as iron-oxide on both sides. Data are recorded on the disk surface in the form of tiny invisible magnetized spots (representing 1s) or non-magnetized spots (representing 0s). A conducting coil, named as Head, performs the job of reading and writing on the surface of magnetic disk. Generally, the head remains stationary while the disk rotates below it for reading or writing operation. The surface of the disk is divided into a number of concentric circles called as tracks. Number of tracks varies from disk to disk. Each track is further subdivided into sectors. Each sector of a disk is assigned a unique number. Each sector having the same capacity.

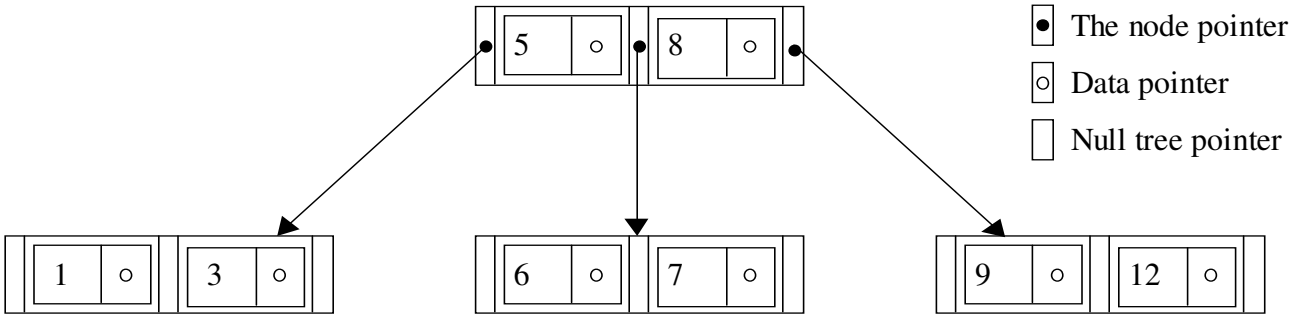


Q.26 Differentiate between B-tree and B^+ tree **(4)**

Ans: A B-tree (or Balanced Tree) is a search tree with additional constraints that ensures that the tree is always balanced and that the space wasted by deletion, if any, never becomes excessive. In a B-tree, every value of the search field appears once at some level in the tree, along with a data pointer. In B^+-tree, data pointers are stored only at the leaf nodes of the tree; hence, the structure of leaf nodes differs from the structure of internal nodes. The leaf nodes of the B^+-tree are usually linked together to provide ordered access on the search field to the records. Some search field values from the leaf nodes are repeated in he internal nodes of the B^+-tree to guide the search.

Q.27 Draw an index structure for B-tree. **(4)**

Ans:

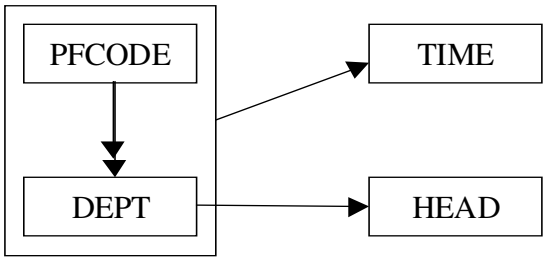


An index structure for B-tree

Q.28 Consider the following relation
Professor (Pfcode, dept, head, time)
It is assumed that
(i) A professor can work in more than one dept.
(ii) The time he spends in each dept is given.
(iii) Each dept has only one head

Draw the dependency diagram for the above relation by identifying the dependencies. **(2)**

Ans:



Dependency Diagram

Q.29 Normalize the relation given Q.28. Justify each step. **(6)**

Ans:
The key of the relation is {PFCODE, DEPT}. Partial functional dependency because DEPT → HEAD. So,
R_1(PFCODE, DEPT, TIME)
R_2(DEPT, HEAD)
R_1 is not in 4NF because of multivalued dependency PFCODE →→ DEPT.
So, final decompositions are:
R_1(PFCODE, DEPT) F_1={PFCODE →→ DEPT}
R_2(PFCODE, TIME)
R_3(DEPT, HEAD) F_2={DEPT → HEAD}

Q.30 Is the decomposition dependency preserving? **(2)**

Ans: We can see that PFCODE, DEPT → TIME is neither in F_1 nor F_2 in the given Q.28 and $(F_1 \cup F_2)^+ \neq F^+$
So, it is not dependency preserving.

Q.31 Define multivalued dependency and 4NF. **(4)**

Ans: **Multivalued Dependency** – Let R be a relation schema and let $\alpha \subseteq R$ and $\beta \subseteq R$. The multivalued dependency

$$\alpha \rightarrow\rightarrow \beta$$

holds on R if, in any legal relation r(R), for all pairs of tuples t_1 and t_2 in r such that $t_1[\alpha] = t_2[\alpha]$, there exist tuples t_3 and t_4 in r such that

$$t_1[\alpha] = t_2[\alpha] = t_3[\alpha] = t_4[\alpha]$$
$$t_3[\beta] = t_1[\beta]$$
$$t_3[R - \beta] = t_2[R - \beta]$$
$$t_4[\beta] = t_2[\beta]$$
$$t_4[R - \beta] = t_1[R - \beta]$$

Fourth Normal Form (4 NF) – A relation schema R is in 4NF with respect to a set D of functional and multivalued dependencies if, for all multivalued dependencies in D^+ of the form $\alpha \rightarrow\rightarrow \beta$, where $\alpha \subseteq R$ and $\beta \subseteq R$, at least one of the following holds:

- $\alpha \rightarrow\rightarrow \beta$ is a trivial multivalued dependency,
- α is a superkey for schema R.

Q.32 What do you understand by transitive dependencies? Explain with an example any two problems that can arise in the database if transitive dependencies are present in the database. **(7)**

Ans: Suppose X, Y, and Z are the set of attributes. If X → Y and Y → Z then the functional dependency X → Z is a transitive functional dependency. In other words, if a nonkey attribute is also functionally dependent on the other nonkey attribute(s), then there is a transitive functional dependency. Transitive functional dependencies are also cause the data redundancies and the other consequences. For example, consider the following relation:

```
CREATE TABLE CONTACTS
( CONTACT_ID          NUMBER(5) PRIMARY KEY,
  L_NAME              VARCHAR(20),
  F_NAME              VARCHAR(20),
  COMPANY_NAME        VARCHAR(20),
  COMPANY_LOCATION    VARCHAR(50))
```

In the above relation, all the nonkey attributes are functionally dependent on the primary key, i.e., CONTACT_ID. But, there is also a transitive functional dependency exists in the relation – CONTACT_ID → COMPANY_LOCATION because CONTACT_ID → COMPANY_NAME and COMPANY_NAME → COMPANY_LOCATION. As a result of transitive dependency, there are anomalies in the above relation as follows:

- **Insertion Anomaly** – A new company cannot be inserted until a contact person has been assigned to that company.

- **Deletion Anomaly** – If a company has only one contact person and is deleted from the table, we will loose the information about that company, as the company information is associated with that person.
- **Update Anomaly** – If a company changes its location we will have to make the changes in all the records where the company name appears.

Q.33 Given R {ABCD} and a set F of functional dependencies on R given as $F = \{AB \rightarrow C, AB \rightarrow D, C \rightarrow A, D \rightarrow B\}$. Find any two candidate keys of R. Show each step. In what normal form is R? Justify. **(7)**

Ans: To find out the candidate keys of a relation schema, R, based on given set of FDs, F, we can compute the closure of X (X^+), where X is the set of attributes. If the X^+ have all the attributes of the relation schema then X is the candidate key of R. With the given set of FDs, it is not possible to derive or access all the other attributes based on single attribute (e.g., $\{A\}^+ = \{A\}$, $\{B\}^+ = \{B\}$, $\{C\}^+ = \{CA\}$, $\{D\}^+ = \{DB\}$). Therefore, we have to chose the composite keys:

1. If we assume X = {AB} then X^+ = {ABCD} because we can determine the attributes C and D from AB as per F.
2. If we assume X = {CD} then X^+ = {CDAB} because we can determine the attributes A and B from C and D respectively as per F.

The other possible candidate keys are: {CB} and {AD}.
If we choose, {CD} as the primary key of the relation R, then FDs C→A and D→B are partial functional dependencies as CD is the key of the relation. Therefore, the given relation R is not in 2NF and therefore first normal form (1 NF).

Q.34 What is a query tree? **(2)**

Ans: A query tree, also called as operator graph, is 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 or root node.

Q.35 What is meant by heuristic optimisation? Discuss the main heuristics that are applied during query optimisation. **(6)**

Ans: In heuristic optimization, heuristics are used to reduce the cost of optimization instead of analyzing the number of different plans to find out the optimal plan. For example, A heuristic optimizer would use the rule 'Perform selection operation as early as possible' without finding out whether the cost is reduced by this transformation. Heuristics approach usually helps to reduce the cost but not always. The main heuristics that are applied during query optimization are:

- Pushes the selection and projection operations down the query tree
- Left-deep join trees – convenient for pipelined evaluation
- Non-left-deep join trees

Q.36 Consider the relations of Q.21. For the query

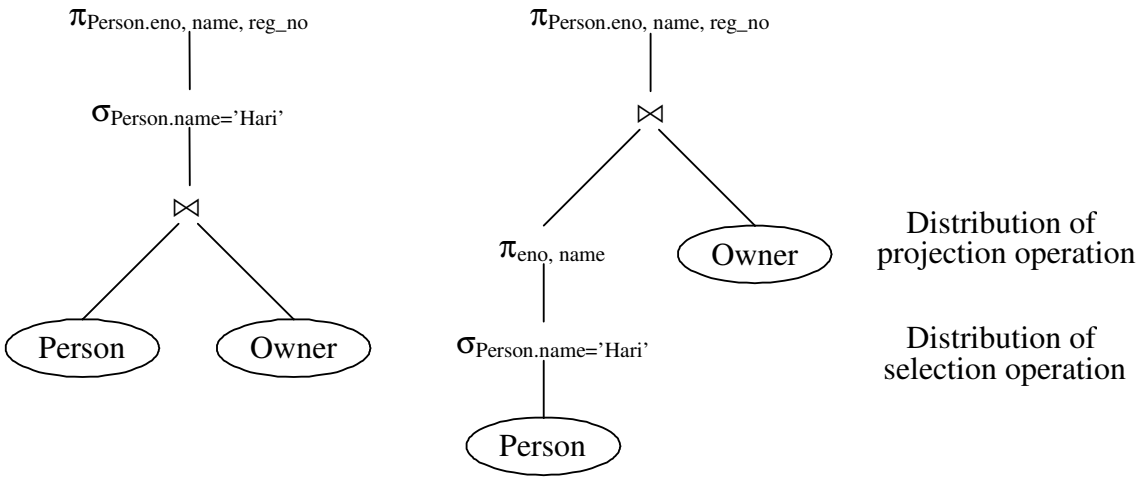
```
Select eno, name, reg_no
From Person, Owner
Where Person.eno = Owner.eno and Person.name = 'Hari'
```

Draw the initial query tree.
Optimise the query and draw the optimised query tree. **(6)**

Ans:

Assuming eno in the select statement refers to persons, the SQL statement will be:
Select Person.eno, name, reg_no From Person, Owner
Where Person.eno = Owner.eno and Person.name = 'Hari'
To draw the query tree the SQL expression is to transformed into relational algebraic expression:

$\pi_{Person.eno,\ name,\ reg\_no}(\sigma_{Person.name='Hari'}(Person \bowtie Owner))$



The initial query tree **The optimised query tree**

Q.37 How does the two phase protocol ensure serializability in database schedules? **(4)**

Ans: A transaction is said to follow the two-phase locking protocol if all locking operations precede the first unlock operation in the transaction. Such transaction can be divided into two phases: and expanding or growing (first) phase and a shrinking (second) phase. The two-phase locking protocol ensures serializability in database schedules. Consider any transaction. The point in the schedule where the transaction has obtained its final lock (the end of its growing phase) is called the lock point of the transaction. Now, transactions can be ordered according to their lock points – this ordering is, in fact, a serializability ordering for the transactions

Q.38 Give a schedule for three transactions T1, T2 and T3 such that the schedule
(i) observes two phase and

(ii) is not deadlock free.

Ans: **(4)**

Transaction T1	Transaction T2	Transaction T3
Write_lock(A)		
Read_item(A)		
	Write_lock(B)	
	Read_item(B)	
Read_Lock(B)		
Read_item(B)		
A := A * B		
Write_item(A)		
Unlock(A)		
Unlock(B)		
	Read_lock(A)	
	Read_item(A)	
	B := A / B	
	Write_item(B)	
	Unlock(B)	
	Unlock(A)	
		Sum := 0
		Read_lock(A)
		Read_lock(B)
		Read_item(A)
		Read_item(B)
		Sum := A + B
		Unlock(A)
		Unlock(B)

Time ↓

Q.39 Describe the algorithm to draw the dependency graph? **(4)**

Ans: **Algorithm to draw Precedence Graph is as follows:**

1. For each transaction T_i participating in schedule S, create a node labeled T_i in the precedence graph.
2. For each case in S where T_j executes a read_item(X) after T_i executes a write_item(X), create an edge ($T_i \rightarrow T_j$) in the precedence graph.
3. For each case in S where T_j executes a write_item(X) after T_i executes a read_item(X), create an edge ($T_i \rightarrow T_j$) in the precedence graph.
4. For each case in S where T_j executes a write_item(X) after T_i executes a write_item(X), create an edge ($T_i \rightarrow T_j$) in the precedence graph.
5. The schedule S is serializable if and only if the precedence graph has no cycles.

Q.40 What is system log? What are the entries? **(3)**

Ans: The system log, which is usually written on stable storage, contains the redundant data required to recover from volatile storage failures and also from errors discovered by the transaction or the database system. System log is also called as log and

has sometimes been called the DBMS journal. It contains the following entries (also called as log records):

- [start_transaction, T]: Indicates that transaction T has started execution.
- [write_item, T, X, old_value, new_value]: Indicates that transaction T has changed the value of database item X from old_value to new_value.
- [read_item, T, X]: Indicates that transaction T has read the value of database item X.
- [commit, T]: Indicates that transaction T has completed successfully, and affirms that its effect can be committed (recorded permanently) to the database.
- [abort, T]: Indicates that transaction T has been aborted.

Q.41 Discuss deferred update technique of recovery. What are the advantages? **(6)**

Ans: The deferred update techniques do not physically update the database on disk until after a transaction reaches its commit point; then the updates are recorded in the database. In other words, deferred update techniques postpone any actual updating of the database on disk until a transaction reaches its commit point. The transaction force-writes the log to disk before recording the updates in the database.

Advantages:

• If a transaction fails before reaching its commit point, it will not have changed the database in any way, so UNDO is not needed. Deferred update can lead to a recovery algorithm known as NO-UNDO/REDO.

• This approach, when used with certain concurrency control methods, is designed never to require transaction rollback, and recovery simply consists of redoing the operations of transactions committed after the last checkpoint from the log.

Q.42 How does the system cope up with a record crash when recovery is going on after the first crash. **(2)**

Ans: In a system the undo and redo operations are required to be idempotent to cope up with a record crash when recovery is going on after the crash. Due to the idempotent property, recovery process, while in the process of undoing or redoing the actions of a transaction, may fail without a trace, and this type of failure can occur any number of times before the recovery is completed successfully.

Q.43 Define check point and its impact on data base recovery. **(3)**

Ans: In a large on-line database system there could be hundreds of transactions handled per minute. The log for this type of database contains a very large volume of information. A scheme called checkpoint is used to limit the volume of log information that has to be handled and processed in the event of a system failure involving the loss of volatile information. The checkpoint scheme is an additional component of the logging scheme. A checkpoint operation, performed periodically, copies log information onto stable storage. For all transactions active at checkpoint, their identifiers and their database modification actions, which at that time are reflected only in the database buffers, will be propagated to the appropriate storage.

Q.44 Write short notes on

a Wait for graph.

b Direct file organization.

c Canonical cover.
d. Timestamp ordering. **(3.5 x 4)**

Ans:

a The wait-for-graph is a directed graph and contains nodes and directed arcs; the nodes of the graph are active transactions. An arc of the graph is inserted between two nodes if there is a data-item is required by the node at the tail of the arc, which is being held by the node at the head of the arc. If there is a transaction T_i, waiting for a data-item that is currently allocated and held by transaction T_j, then there is a directed arc from the node for transaction T_i, to the node for transaction T_j. The wait-for-graph is used for detection of deadlock in the system. If there is cycle exits in the graph, all the transaction (represented as nodes) are in deadlock, which are in the cycle.

b. The direct file organization is used to improve the performance in accessing the records as compared to sequential file organization. In direct file organization, the key value is mapped directly to the storage location. The usual method of direct mapping is by performing some arithmetic manipulation of the key value. This process is called hashing. When using this organization, an application such as on-line transaction processing system can be designed so that centralized data are not only instantly accessible but are always up-to-date.

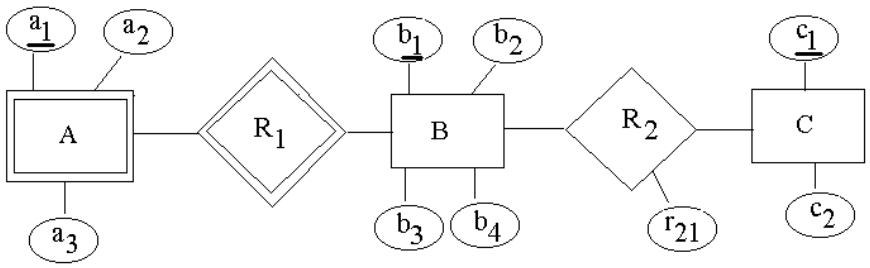
c. A canonical cover F_c for F is a set of dependencies such that F logically implies all dependencies in F_c, and F_c logically implies all dependencies in F. F_c must have the following properties:

- No functional dependency in Fc contains extraneous attribute
- Each left side of a functional dependency in F_c is unique. That is, there are no two dependencies $\alpha_1 \rightarrow \beta_1$ and $\alpha_2 \rightarrow \beta_2$ in F_c such that $\alpha_1 = \alpha_2$.

F_c is minimal in certain sense – it does not contain extraneous attributes, and it combines functional dependencies with the same left side. It is cheaper to test Fc than it is to test F itself.

d. In timestamp-based method, a serial order is created among the concurrent transaction by assigning to each transaction a unique nondecreasing number. The usual value assigned to each transaction is the system clock value at the start of the transaction, hence the name timestamp ordering. This value then can be used in deciding the order in which the conflict between two transactions is resolved. A transaction with a smaller timestamp value is considered to be an "older" transaction than another transaction with a larger timestamp value.

Q.45 Convert the following ER – diagram into a relational database (the primary keys are underlined): **(8)**



Ans: The relational database schema for the given ER diagram is as follows:
$A(\underline{a_1, b_1}, a_2, a_3)$
$B(\underline{b_1}, b_2, b_3, b_4)$
$C(\underline{c_1}, c_2)$
$D(\underline{b_1, c_1}, r_{21})$

Q.46 List two restrictions that are applied on the modification (updation, insertion or deletion) of base tables through view. With the help of examples, explain why those restrictions are required. **(6)**

Ans: The two restrictions are:

- If a view does not contain primary key and/or at least one of the attributes with the NOT NULL constraint that do not have default values specified, is not updatable. This restriction is required because if we are inserting any row in the view than all attributes not included in view of the base relation are will be represented by the NULLs if any default value is not specified. In that case, if any attribute is set as NOT NULL or primary key, then it must not be NULL, and the insertion will be violating the constraints.
- If a view defined using grouping and/or aggregate functions, is not updatable. This restriction is also required because, if we want to modify the base relation through view then we are not in the position to identify which tuple(s) will be modified in the group in what way.

For Example:
Employee:

EmpNo	Name	Job	Salary
1	Ramesh	Salesman	5000
2	John	Clerk	3000
3	Sanjeet	Manager	10000
4	Praveen	Salesman	4500
5	Rakesh	Manager	12000

View1: Create View Name_Job as Select Name, Job from Employee
View2: Create View Job, Sum(salary) from Employee Group By Job
In the view1 if we want to insert a new record, then the primary key constraint violates because primary key does not accept NULL. In the view2 if we want to make any modification (insertion, updation, or deletion) then it is difficult to identify the corresponding tuples of base relation from the view for the modification.

Q.47 Consider the following relations with key underlined
Customer (C#, Cname, Address)
Item (I#, Iname, Price, Weight)
Order (O#, C#, I#, Quantity)
Write SQL queries for the following:

a. List the names of customers who have ordered items weighing more than 1000 and only those.
b. List the names of customers who have ordered atleast one item priced over Rs.500.
c. Create a view called "orders" that has the total cost of every order. **(9)**

Ans:

(a)
```
SELECT CNAME FROM CUSTOMER WHERE C# IN
        (SELECT C# FROM ORDER
         MINUS
         SELECT C# FROM ORDER, ITEM
           WHERE ORDER.I# = ITEM.I# AND WEIGHT <= 1000)
```

(b)
```
SELECT CNAME FROM CUSTOMER WHERE C# IN
        (SELECT C# FROM ORDER WHERE I# IN
          (SELECT I# FROM ITEM WHERE PRICE > 500))
                    OR
SELECT DISTINCT CNAME FROM CUSTOMER, ORDER, ITEM
WHERE CUSTOMER.C# = ORDER.C# AND ORDER.I# = ITEM.I#
      AND PRICE > 500
```

(c)
```
CREATE VIEW ORDERS (O#, TOTALCOST) AS
SELECT O#, SUM(QUANTITY * PRICE)
FROM ORDER, ITEM WHERE ORDER.I# = ITEM.I#
GROUP BY O#
```

Q.48 List the Armstrong's axioms for functional dependencies. What do you understand by soundness and completeness of these axioms? **(5)**

Ans: **The Armstrong's axioms are:**

- F1: Relexivity: If X is a set of attributes and $Y \subseteq X$ then $X \rightarrow Y$ holds.
- F2: Augmentation: If $X \rightarrow Y$ holds and Z is a set of attributes then $XZ \rightarrow YZ$.
- F3: Transitivity: $\{X \rightarrow Y, Y \rightarrow Z\} \models \{X \rightarrow Z\}$

Soundness: By sound, we mean that a given set of functional dependencies F specified on a relation schema R, any dependency that we can infer from F by using F1 through F3 holds in every relation state r of R that satisfies the dependencies in F.

Completeness: By complete, we mean that using F1 through F3 repeatedly to infer dependencies until no more dependencies can be inferred results in the complete set of all possible dependencies that can be inferred from F.

Q.49 Consider the following relations

Employee (E#, Ename, salary, Bdate, D#)
Department (D#, Dname, mgremp#, Location)
Dependent (E#, DependentName)

a. Write tuple calculus queries for the following:

(i) List the names of mangers who have at least one dependent.

(ii) List the names of employees working for 'research' department. **(6)**

b. Write QBE queries for the following:

(i) List all the employees who earn more than the average salary of all employees.

(ii) Increase the salary of managers by 10%.

(iii) List the names of all employees working in Delhi.

(iv) Change the location of all departments to "Mumbai" which have location as "Bombay" **(8)**

Ans:

(a) (i) List names of managers who have at least one dependent.

{m[Ename] | m ∈ EMPLOYEE ∧ ∃u, t (t ∈ DEPENDENT
∧ u ∈ DEPARTMENT ∧ t[E#] = u[mgremp#] ∧ m[E#] = u[mgremp#])}

(ii) List the names of employees working for 'research' department.

{e[Ename] | e ∈ EMPLOYEE ∧ ∃d (d ∈ DEPARTMENT
∧ d[Dname] = 'research' ∧ e[D#] = d[D#])}

(b) (i) List all the employees who earn more than the average salary of all employees.

	E#	Ename	Salary	Bdate	D#
P.	EX	NX	SX AVG.ALL.SY	BX	DX

Conditions
SX > AVG.ALL.SY

(ii) Increase the salary of managers by 10%.

DEPARTMENT	D#	Dname	mgremp#	Location
			MX	

EMPLOYEE	E#	Ename	Salary	Bdate	D#
U.	MX		SX SX * 1.1		

(iii) List the names of all employees working in Delhi

DEPARTMENT	D#	Dname	mgremp#	Location
	DX			Delhi

EMPLOYEE	E#	Ename	Salary	Bdate	D#
		P.NX			DX

(iv) Change the location of all departments to "Mumbai" which have locations as "Bombay."

DEPARTMENT	D#	Dname	mgremp#	Location
U.				Bombay Mumbai

Q.50 Consider the following relation:

A	B	C
10	b1	c1
10	b2	c2
11	b4	c1
12	b3	c4
13	b1	c1
14	b3	c4

Given the above state, which of the following dependencies hold in the above relation at this point of time? If the dependency does not hold, explain why, by specifying the tuples that cause the violation

(i) $A \rightarrow B$ (ii) $B \rightarrow C$
(iii) $C \rightarrow B$ (iv) $B \rightarrow A$
(v) $C \rightarrow A$.

Does the above relation have a potential candidate key? If it does, what is it? If it does not, why not? **(7)**

Ans:

S. No.	FD	Valid / Invalid	Tuples that cause the violation
(i)	A → B	Invalid	1st and 2nd
(ii)	B → C	Valid	
(iii)	C → B	Invalid	3rd and 1st
(iv)	B → A	Invalid	1st and 5th, 4th and 6th
(v)	C → A	Invalid	1st, 3rd, and 5th

Yes, the given relation has the potential candidate keys. The candidate keys are: {AB}, and {BC} because $\{AB\}^+ = \{ABC\}$ and $\{BC\}^+ = \{BCA\}$.

Q.51 Find 3NF decomposition of the relation scheme,
R = {Faculty, Dean, Dept, Chairperson, Professor, Rank, Student} with the set of functional dependencies,

F = {Faculty → Dean
Professor → Rank, Chairperson
Student → Department, Faculty,Dean
Professor, Rank → Department, Faculty}
Department → Chairperson
Department → Faculty
Dean → Faculty

(7)

Ans: To find out 3NF decomposition of a relation we use canonical cover (F_c).

Requirements for canonical cover (F_c):

1. Each FD in Fc must be simple.

F' = {Faculty → Dean, Department → Chairperson, Professor → Rank, Professor → Chairperson, Department → Faculty, Student → Department, Student → Faculty, Student → Dean, Dean → Faculty, ProfessorRank → Department, ProfessorRank → Faculty}

2. Each FD in Fc must be left reduced.

As given in F, the attribute Rank is functionally dependent on the Professor. Therefore the FDs ProfessorRank → Department and ProfessorRank → Faculty can be left reduced to Professor → Department and Professor → Faculty respectively based on the Augmentation axiom.

F'' = {Faculty → Dean, Department → Chairperson, Professor → Rank, Professor → Chairperson, Department → Faculty, Student → Department, Student → Faculty, Student → Dean, Dean → Faculty, Professor → Department, Professor → Faculty}

3. Each FD in Fc must be non-redundant.

The FDs Professor → Chairperson, Student → Faculty, Student → Dean and Professor → Faculty are redundant and have to be removed based on the Transitivity axiom.

F_c = {Faculty → Dean, Department → Chairperson, Professor → Rank, Department → Faculty, Student → Department, Dean → Faculty, Professor → Department}

To make lossless join decompositions, it is required to preserve the key of the original relation. The key of the relation R is {Professor, Student}. The 3NF decompositions of the given relation are:

R_1 = {Faculty, Dean} F_1= {Faculty → Dean, Dean → Faculty}

R_2 = {Department, Chairperson, Faculty} F_2 = {Department → Chairperson,Faculty}

R_3 = {Professor, Rank, Department} F_3 = {Professor → Rank,Department}

R_4 = {Student, Department} F_4 = {Student → Department }

R_5 = {Professor, Student} F_5 = {ProfessorStudent → ϕ}

Q.52 Set R(A,B,C) and S(B,C,D) be the relations:

R:

A	**B**	**C**
a	c	c
a	e	c
a	c	d
b	d	d

S:

B	**C**	**D**
c	c	a
d	c	a
e	d	b

Compute the following for the relations above:

(i) $R \div \pi_c(s)$ **(ii)** $\pi_{R:B,\,S:C}(\sigma_{A=D}(R \times S))$

(iii) $R \cup S$. **(8)**

Ans: (i) $R \div \pi_c(S)$

$\pi_c(S)$

C
c
d

$R \div \pi_c(S)$

A	**B**
a	c

(ii) $\pi_{R.B,\,S.C}(\sigma_{A=D}(R \times S))$

R.B	**S.C**
c	c
e	c
d	d

(iii) $R \cup S$

A	**B**	**C**
a	c	c
a	e	c
a	c	d
b	d	d
c	c	a
d	c	a
e	d	b

Q.53 Consider the decomposition of relation scheme, shipping = (Ship, Capacity, Date, Cargo, Value) with the set of functional dependencies, F = {Ship →Capacity

Ship, Date $\rightarrow$Cargo, Cargo, Capacity $\rightarrow$Value} into R_1={Ship, Capacity} with F_1 = {Ship $\rightarrow$Capacity} and R_2={Ship, Date, Cargo, Value} with F_2={Ship, Date $\rightarrow$Cargo, Value}.
Is this decomposition in BCNF? Is this decomposition lossless and dependency preserving? Justify your answers. **(6)**

Ans: The given decomposition of the given relation is in BCNF because there is no overlapped composite keys in the relation. The decomposition is lossless because the key of the original relation {Ship,Date} is preserved in the decomposition R2. But the decomposition is not dependency preserving because the FD Cargo,Capacity → Value is not implied by the set of FDs {Ship → Capacity, Ship,Date → Cargo,Value}

Q.54 What are deferred modification and immediate modification technique for recovery? How does recovery takes place in case of a failure in these techniques? **(7)**

Ans: **Deferred Update** – The deferred update techniques do not physically update the database on disk until after a transaction reaches its commit point; then the updates are recorded in the database. In other words, deferred update techniques postpone any actual updating of the database on disk until a transaction reaches its commit point. The transaction force-writes the log to disk before recording the updates in the database.
Immediate Update – The immediate update techniques may apply changes to the database on disk before the transaction reaches a successful conclusion. However, these changes are typically recorded in the log on disk by force writing before they are applied to the database, making recovery still possible.
Recovery – In deferred update technique, if a transaction fails before reaching its commit point, it will not have changed the database in any way, so UNDO is not needed. Deferred update can lead to a recovery algorithm known as NO-UNDO/REDO. This approach, when used with certain concurrency control methods, is designed never to require transaction rollback, and recovery simply consists of redoing the operations of transactions committed after the last checkpoint from the log. In immediate update technique, if a transaction fails after recording some changes in the database but before reaching its commit point, the effect of its operations on the database must be undone; that is, the transaction must be rolled back. This technique requires both operations and is used most often in practice. Therefore, it is also known as UNDO/REDO.

Q.55 What are the two integrity rules? Explain with examples how these rules are important to enforce consistent database states. **(7)**

Ans: The two integrity rules are: Entity Integrity Rule and Referential Integrity Rule.
Entity Integrity Rule – If the attribute A of relation R is a prime attribute of R then A cannot accept null values.
Referential Integrity Rule – In referential integrity, it is ensured that a value that appears in one relation for a given set of attributes also appears for a certain set of attributes in another relation.

For example:

STUDENT

Enrl No	Roll No	Name	City	Mobile
11	17	Ankit Vats	Delhi	9891663808
15	16	Vivek Rajput	Meerut	9891468487
6	6	Vanita	Punjab	
33	75	Bhavya	Delhi	9810618396

GRADE

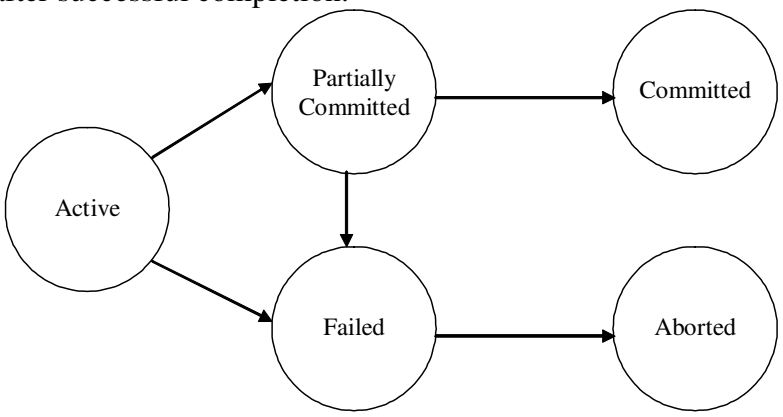
Roll No	Course	Grade
6	C	A
17	VB	C
75	VB	A
6	DBMS	B
16	C	B

- **Roll No** is the primary key in the relation STUDENT and **Roll No + Course** is the primary key of the relation GRADE - (Entity Integrity). Primary keys ensure that any prime attribute must not be NULL.
- **Roll No** in the relation GRADE (child table) is a foreign key, which is referenced from the relation STUDENT (parent table) - (Referential Integrity). Referential integrity here ensures that the values in the foreign key must belongs to it parent key or it may be entirely NULL.

Q.56 What are the various states through which a transaction passes through in its lifetime? Briefly discuss all the events that causes transition from one state to another. **(7)**

Ans: A transaction can be considered to be an atomic operation by the user, but actually it goes through a number of states during its lifetime as given follows:

- **Active**, the initial state; the transaction stays in this state while it is executing
- **Partially Committed**, after the initial statement has been executed
- **Failed**, after the discovery that 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.



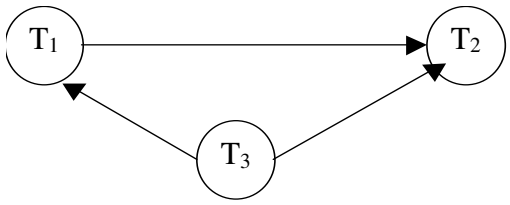
State Diagram of a Transaction

It is assumed that before a transaction starts, the database is in a consistent state. A transaction starts when the first statement of the transaction is executed; it becomes active. After any outcome, the database will be considered in consistent state.

Q.57 Consider three transactions : T_1, T_2 and T_3. Draw the precedence graph for the following schedule consisting of these three transactions and determine whether it is serializable. If so, give its serial order(s). **(7)**

Time	T_1	T_2	T_3
t_1 :			read(Y)
t_2 :			read(Z)
t_3 :	read(X)		
t_4 :	write(X)		
t_5 :			write(Y)
t_6 :			write(Z)
t_7 :		read(Z)	
t_8 :	read(Y)		
t_9 :	write(Y)		
t_{10} :		read(Y)	
t_{11} :		write(Y)	
t_{12} :		read(X)	
t_{13} :		write(X)	

Ans:



Precedence Graph

The given schedule is serializable. The serial orders of the transactions are: T_3, T_1, and T_2.

Q.58 Which sorting technique is used to sort databases, whose sizes are very big? Give one such algorithm. Why do sorting techniques like quicksort, insertion sort, etc. not applied on very big databases? **(7)**

Ans: To sort a large database or file, the sort-merge (or k-way merge) algorithm is widely used. This is one of the external sorting algorithms. External sorting algorithm refers to sorting algorithms that are suitable for large file of records stored on disk that do not fit entirely in main memory, such as most database files. The sort-merge algorithm starts by sorting small subfiles -–called runs – of the main file and then merges the sorted runs, creating larger sorted subfiles that are merged in turn. Like other database algorithms, it requires buffer space in main memory, where the actual sorting and

merging of the runs is performed. This algorithm consists of two phases: (1) the sorting phase, and (2) the merging phase.

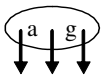
The sorting techniques like quick sort, insertion sort, etc. are not applied on large databases because these are internal sorting algorithms, which requires buffer space in main memory. The space complexity of these algorithms are O(n) and require a large buffer space in main memory to sort the large database. Therefore, internal sorting algorithms are not applied on large databases.

Q.59 Insert the keys below in the order stated into an initially empty B-tree of order 3. Show all intermediate steps

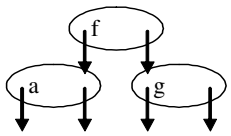
a g f b k d h m j e s i r **(7)**

Ans:

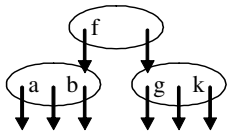
After inserting a and g



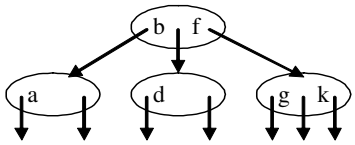
After inserting f



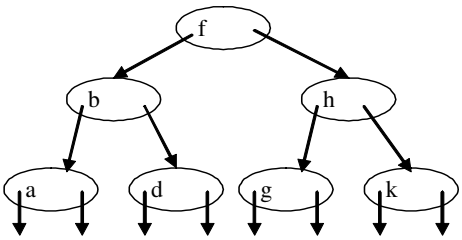
After inserting b and k



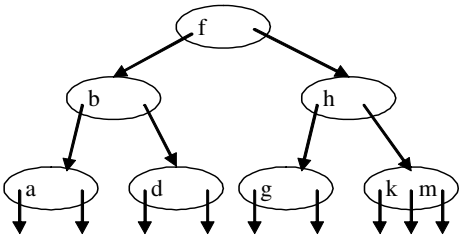
After inserting d



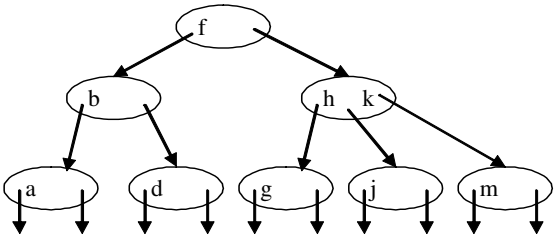
After inserting h



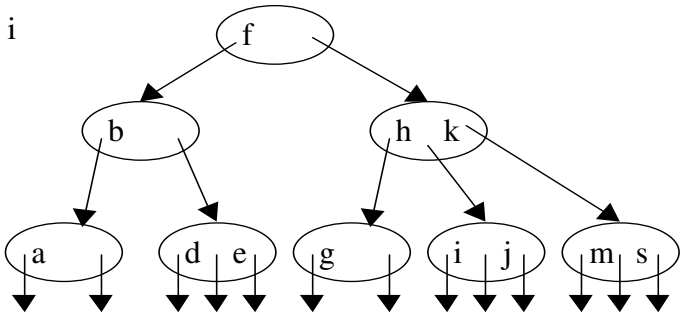
After inserting m



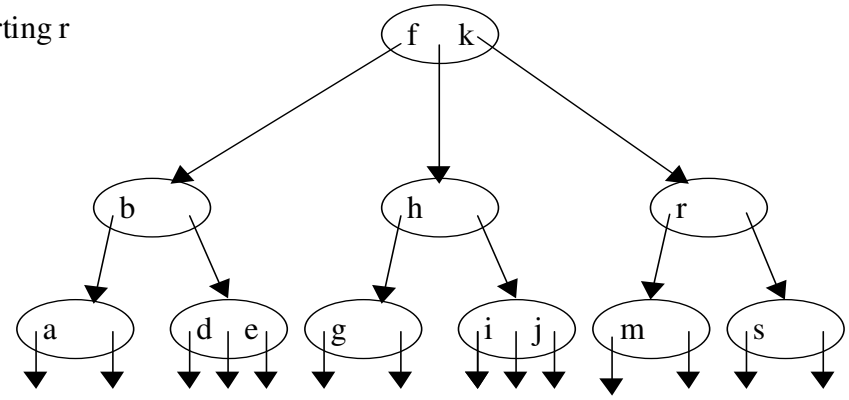
After inserting j



After inserting e, s, and i



After inserting r



Q.60 What are wait-for-graphs? Give the algorithm to construct a wait-for-graph from a given schedule of transactions? How can deadlocks be detected from wait-for-graphs? **(7)**

Ans: The wait-for-graph is a directed graph and contains nodes and directed arcs; the nodes of the graph are active transactions. An arc of the graph is inserted between two nodes if there is a data-item is required by the node at the tail of the arc, which is being held by the node at the head of the arc.

Algorithm to construct wait-for-graph is as follows:

1. For each transaction T_i active at the time of deadlock detection, create a node labeled T_i in the wait-for-graph.
2. For each case, if there is a transaction T_i, waiting for a data-item that is currently allocated and held by transaction T_j, then there is a directed arc from the node for transaction T_i, to the node for transaction T_j.
3. For each case, if there is a directed arc from the node for transaction T_i, to the node for transaction T_j and transaction Tj released the lock, then remove the arc between them.
4. There no deadlock if the wait-for-graph is acyclic.

The wait-for-graph is used for detection of deadlock in the system. If there is any cycle exits in the wait-for-graph, all the active transaction (represented as nodes) are in deadlock, which are in the cycle.

Q.61 What is the need of a log in a DBMS? Briefly describe the various types of records that are normally present in a log. **(7)**

Ans: The system log, which is usually written on stable storage, contains the redundant data required to recover from volatile storage failures and also from errors discovered by the transaction or the database system. System log is also called as log and

has sometimes been called he DBMS journal. It contains the following entries (also called as log records):

- [start_transaction, T]: Indicates that transaction T has started execution.
- [write_item, T, X, old_value, new_value]: Indicates that transaction T has changed the value of database item X from old_value to new_value.
- [read_item, T, X]: Indicates that transaction T has read the value of database item X.
- [commit, T]: Indicates that transaction T has completed successfully, and affirms that its effect can be committed (recorded permanently) to the database.
- [abort, T]: Indicates that transaction T has been aborted.

Q.62 Write short notes on (any **FOUR**) of the following:

(i) Deadlock recovery measures.
(ii) Query optimisation.
(iii) Hashing techniques.
(iv) Two phase locking protocol.
(v) ANSI SPARC architecture.
(vi) Domain calculus. **(3.5 x 4=14)**

Ans:

(i) To recover from deadlock, the cycle in the wait-for-graph must be broken. The common method of doing this is to rollback one or more transactions in the cycles until the system exhibits no further deadlock situation. The selection of the transaction to be rolled back is based on the following deadlock recovery measures:

- The progress of the transaction and the number of data-items it has used and modified. It is preferable to rollback a transaction that has just started or has not modified any data-item.
- The amount of computing remaining for the transaction and the number of data items that have yet to be accessed by the transaction. It is preferable not to rollback a transaction it has almost run to completion and/or it needs very few additional data-items before its termination.
- The relative cost of rolling back a transaction. It is preferable to roll back a less important or non-critical transaction.

(ii) The query parser typically generate a standard initial tree to correspond to an SQL query, without doing any optimization. Such a canonical query tree represents a relational algebra expression that is very inefficient if executed directly. A query typically has many possible execution strategies, and the process of choosing a suitable one for processing a query is known as query optimization. The main guidelines that are applied during query optimization are:

- Combine a cascade of selections
- Combine a cascade of projections
- Commute selection and projection
- Commuting selection with join or cartesian product Commuting projection with join or cartesian product
- Commute projection with set operations
- Commute selection with set operations

(iii) The hashing techniques can be classified as:

Static Hashing Techniques – In this technique, the data can be viewed a collection of buckets, with one primary page and possibly additional overflow pages per bucket. A file consists of buckets 0 through N-1, with one primary page per bucket initially and additional overflow pages chained with bucket, if required later. Buckets contain data entries (or data records). A major drawback of the static hashing is that the hash address space is fixed. Hence, it is difficult to expand or shrink the file dynamically.

Dynamic Hashing Techniques – There are two schemes in it: extendible hashing – stores an access structure in addition to the file , and hence is somewhat similar to indexing. The main difference is that the access structure is based on the values that the result after application of the hash function to the search field. The second one is, linear hashing – does not require additional access structures.

(iv) A transaction is said to follow the two-phase locking protocol if all locking operations precede the first unlock operation in the transaction. Such transaction can be divided into two phases: and expanding or growing (first) phase, during which new locks on items can be acquired but none can be released; and a shrinking (second) phase, during which existing locks can be released but no new locks can be acquired. This two-phase protocol can ensure the serializability because all the data items required by a transaction will be locked at the beginning of the transaction and if other transaction wants to use those data items then it has to wait until they become unlocked.

(v) The three-schema architecture is also known as ANSI SPARC architecture. The goal of the three-schema architecture is to separate the user applications and the physical database. The view at each of these levels is described by a schema. The processes of transforming requests and results between levels are called mappings. In this architecture, schemas can be defined at the following three levels:

- **External Level or Subschema –** It is the highest level of database abstraction where only those portions of the database of concern to a user or application program are included. Any number of user views may exist.
- **Conceptual Level or Conceptual Schema -** At this level of database abstraction all the database entities and the relationships among them are included. There is only one conceptual schema per database.
- **Internal Level or Physical Schema –** It is closest to the physical storage method used.

(**vi)** Domain calculus is one of the type of the relational calculus. The formal specification of the domain calculus was proposed after the development of the QBE system. Domain calculus has the variables range over single value from domains of attributes. To form a relation degree *n* for a query result, we must have *n* domain variables – one for each attribute. An expression of the domain calculus is of the form

$$\{X \mid F(X)\}$$

Where F is a formula on X and X represents a set of domain variables. The expression characterizes X such that F(X) is true.

Q.63 Describe five main functions of a database administrator. **(5)**

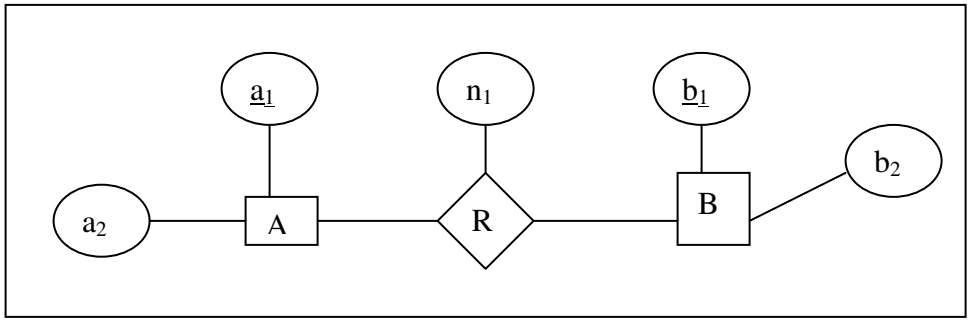
Ans: A **database administrator** (**DBA**) is a person who is responsible for the environmental aspects of a database. In general, these include:

- **Recoverability** - Creating and testing Backups
- **Integrity** - Verifying or helping to verify data integrity
- **Security** - Defining and/or implementing access controls to the data
- **Availability** - Ensuring maximum uptime

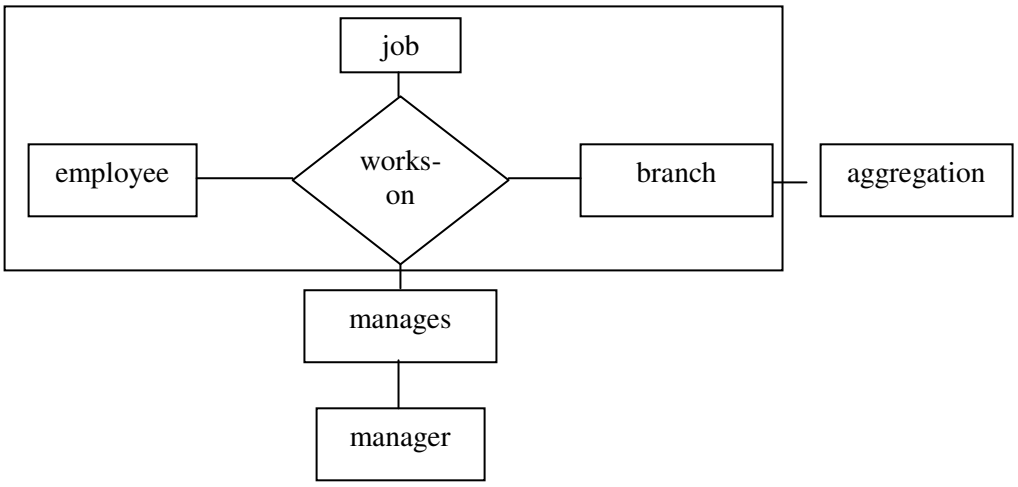
- **Performance** - Ensuring maximum performance
- **Development and testing support** - Helping programmers and engineers to efficiently utilize the database.

Q.64 Define the following with respect to an E-R diagram. Explain the manner in which each is mapped to a table. Illustrate with an example.
(i) Relationship set. **(ii)** Aggregation.
(iii) Multivalued attribute. **(9)**

Ans:
(i) Relationship set: It is a set of relationships of same type. Formally, it is a mathematical relation on n >=2 entity sets. If E_1, E_2-----,E_n are entity sets, then a relationship set R is a subset of { (e1, e2,-----, en) | e1, $e_1 \varepsilon E_1$, e2 εE_2 , ----, $e_n \varepsilon E_n$ } where (e_1,e_2,----,e_n) is a relationship
Mapping to a table: Let R be a relationship set, let a_1, a_2,-----, a_m be set of attributes formed by the union primary keys of each of entity sets participating in R, and let the descriptive attributes (if any) of R be b_1,b_2,------b_n. Then the table corresponding to R will have one column for each attribute of the set: { a_1, a_2, ----, a_m) U { b_1, b_2,----b_n}
Eg, given an ER diagram as



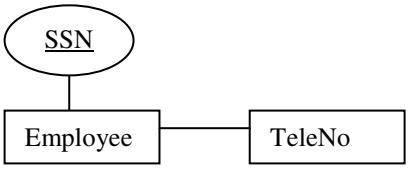
The table corresponding to R will be R(a_1, b_1, r_1)
ii) Aggregation: It is an abstraction through which relationships are treated as higher level entities.
Mapping to a table: This is explained through following example. Consider the E-R diagram given below:



The table for relationship set manages between aggregation onworks-on and entity set manager includes a column for each attribute in primary keys of entity set manages and relationship set works. It would also include a column for any descriptive attribute if they exist, of relationship manages.

iii) Multivalued attribute: An attribute that is having more than one values for a given attribute. Eg, Telephone no. of an employee

Mapping to a table: For a multivalued attribute M, a table is created with a column C that corresponds to M and columns corresponding to the primary key of the entity set or relationship set of which M is an attribute. Eg, Given ER diagram as below:



The corresponding table will be Telephone (SSN, TeleNo)

Q.65 Consider the following relations with primary keys underlined.
Salesperson (SNo, Sname, Designation)
Area (ANo, Aname, ManagerNo)
Product (PNo, Pname, Cost)
SAP (SNo, ANo, PNo)

(a) Define the schema in SQL specify the attributes, and keys assuming that ManagerNo is a foreign key. Specify the constraint that the cost of a product cannot be greater than Rs.10000/-. **(5)**

(b) Answer using SQL

(i) Get the names of all the products that are sold.

(ii) Get the product numbers which are marketed by alteast two sales persons.

(iii) Get the names of all salespersons who are not Managers. **(9)**

Ans:

(a) Create table Salesperson (SNo number(3) primary key, Sname char(30), Destination char(15));
Create table Area (Ao number(3) primary key, Aname char(30), ManagerNo number(3) references Salesperson(SNo));
Create table Product (PNo number(3) primary key, Pname char(30), Cost number(5) check cost < 100000);
Create table SAP (SNo number(3), ANo number(3), PNo number(3), primary key (SNo, ANo, PNo), SNo references Salesperson(SNo), PNo references Product(PNo), ANo references Area (ANo));

(b)

(i) Select PNo from Product, SAP where Product. PNo = SAP.PNo;

(ii) Select T1.PNo from SAP T1, SAP T2 where T1.PNo= T2.PNo and T1.SNo <> T2.SNo;

(iii) Select Sname from Salesperson where designation <> 'Managers'.

Q.66 What is the basic difference between relational algebra and relational calculus? Define the atoms in tuple relational calculus. Use these to define the formulae. **(5)**

Ans: An algebra consists of a set of *objects* and a set of *operators* that together satisfy certain *axioms* or *laws* (such as closure, commutativity, associativity, and so on). The word "algebra" itself ultimately derives from Arabic *al-jebr,* meaning a *resetting* (of something broken) or a *combination.*

Relational calculus is an applied form of a fundamental branch of logic called predicate calculus. In general, the term "calculus" signifies merely a system of computation (the Latin word calculus means a pebble, perhaps used in counting or some other form of reckoning). Thus, relational calculus can be thought of as a system for computing with relations

Atoms

For the construction of the formulas we will assume an infinite set V of tuple variables. The formulas are defined given a database schema S = (D, R, h) and a partial function *type* : $V \rightarrow 2^C$ that defines a *type assignment* that assigns headers to some tuple variables. We then define the *set of atomic formulas A*[*S*,*type*] with the following rules

1. if *v* and *w* in *V*, *a* in *type*(*v*) and *b* in *type*(*w*) then the formula " *v.a* = *w.b* " is in *A*[*S*,*type*],
2. if *v* in *V*, *a* in *type*(*v*) and *k* denotes a value in *D* then the formula " *v.a* = *k* " is in A[S,type], and
3. if *v* in *V, r* in *R* and *type(v) = h(r)* then the formula " *r(v)* " is in *A[S,type].*

   Examples of atoms are:
   - (*t*.age = *s*.age) — *t* has an age attribute and s has an age attribute with the same value
   - (*t*.name = "Codd") — tuple t has a name attribute and its value is "Codd"
   - Book*(t)* — tuple *t* is present in relation Book.

The formal semantics of such atoms is defined given a database *db* over *S* and a tuple variable binding *val* : $V \rightarrow T_D$ that maps tuple variables to tuples over the domain in *S*:

1. " *v.a* = *w.b* " is true if and only if *val*(*v*)(*a*) = *val*(*w*)(*b*)
2. " *v.a* = *k* " is true if and only if *val*(*v*)(*a*) = *k*
3. " *r*(*v*) " is true if and only if *val*(*v*) is in *db*(*r*)

Formulas

The atoms can be combined into formulas, as is usual in first-order logic, with the logical operators Λ (and), V (or) and ¬ (not), and we can use the existential quantifier (∃) and the universal quantifier (∀) to bind the variables. We define the *set of formulas F*[*S*,*type*] inductively with the following rules:

1. every atom in *A*[*S*,*type*] is also in *F*[*S*,*type*]
2. if f_1 and f_2 are in *F*[*S*,*type*] then the formula " f_1 V f_2 " is also in *F*[*S*,*type*]
3. if f_1 and f_2 are in *F*[*S*,*type*] then the formula " f_1 Λ f_2 " is also in *F*[*S*,*type*]
4. if *f* is in *F*[*S*,*type*] then the formula " ¬ *f* " is also in *F*[*S*,*type*]
5. if *v* in *V*, *H* a header and *f* a formula in $F[S,type_{[v->H]}]$ then the formula " ∃ *v* : *H* (*f*) " is also in *F*[*S*,*type*], where $type_{[v->H]}$ denotes the function that is equal to *type* except that it maps *v* to *H*,
6. if *v* in *V*, *H* a header and *f* a formula in $F[S,type_{[v->H]}]$ then the formula" ∀*v* : *H* (*f*) " is also in *F*[*S*,*type*]

Example of formula:

t.name = "C. J. Date" ∀ *t*.name = "H. Darwen"

Q.67 Consider the following relations

Person (name, street, city)

Owns (name, reg_no, model, year)
Accident (date, reg_no)
Answer the following using tuple relational calculus
(i) Find the names of persons who are not involved in any accident.
(ii) Find the names and street of persons who own a maruti car.
(iii) Find the registration numbers of the cars manufactured in the year 2004. **(9)**

Ans:
(i) $\{t[name] \mid t \in owns \wedge \neg \exists a \in Accident(t[reg\text{-}no] = a[reg\text{-}no])\}$
(ii) $\{t[name], t[street] \mid t \in person \wedge \exists O \in Owns(t[name] = O[name] \wedge O[model] = \text{'Maruti'})\}$
(iii) $\{t[reg\text{-}no] \mid t \in owns \wedge t(year] = 2004\}$

Q.68 Define functional and multivalued dependencies. **(2)**

Ans: A **functional dependency** is a property of the semantics of the attributes in a relation. The semantics indicate how attributes relate to one another, and specify the functional dependencies between attributes. When a functional dependency is present, the dependency is specified as a constraint between the attributes. Consider a relation with attributes A and B, where attribute B is functionally dependent on attribute A. If we know the value of A and we examine the relation that holds this dependency, we will find only one value of B in all of the tuples that have a given value of A, at any moment in time. Note however, that for a given value of B there may be several different values of A.
Multivalued Dependencies: A multivalued dependency is an outcome of two independent 1: N relationships A: B and A: C being mixed in the same relation R(A,B,C)

Q.69 Consider the relation Student (stid, name, course, year)
Given that
A student may take more than one course but has unique name and the year of joining.
(i) Identify the functional and multivalued dependencies for Student. **(4)**
(ii) Identify a candidate key using the functional and multivalued dependencies arrived at in step (b). **(4)**
(iii) Normalize the relation so that every decomposed relation is in 4NF. **(4)**

Ans:
(i) **F.D are:** stid → name, year
M.V.D. are: stid →→ Course
course →→ stid
(ii) **Candidate Keys:** (stid, course)
(iii) Since in both the MVDs, LHS is not a superkey . Therefore decomposition is as follows:

R1(stid, name, year)	Primary Key = stid
R2(stid, course)	Primary Key = (stid, course) Foreign Key = stid

Q.70 Explain the following
(i) ISA relationship.
(ii) NULL value.
(iii) Trigger.

(iv) EXEC statement in SQL. **(2 x 4)**

Ans:

(i) **ISA** notation is used in generalization/ specialization to show relationship between lower and higer level entity set.

(ii) **NULL value** NULL means something is unknown. It does NOT mean null (the digit 0). Null is also used as attribute value for a particular entity where attribute is not applicable eg, name of child for unmarried.

(iii) **Trigger:** A **database trigger** is procedural code that is automatically executed in response to certain events on a particular table in a database. Triggers can restrict access to specific data, perform logging, or audit data modifications.

(iv) **EXEC statement in SQL**

All statements that begin with EXEC SQL are embedded SQL database statements. High level languages like C can be used to write and SQL statements may be embedded in them using EXEC statement

Q.71 Define a view ProductArea in relational algebra and SQL, using the relations of question (65) above, which contains the area name and the names of products sold in that area.**(6)**

Ans: $\Pi_{aname,\ pname}$ (Area ⋈ SAP ⋈ Product)

Create view ProductArea as select Aname, Pname from Area, Product, SAP
where SAP.PNo = Product.PNo and SAP.ANo= Area. ANo

Q.72 Compare the two method for storing variable length records – byte string representation and fixed length representation. Discuss the merits and demerits of the two. **(6)**

Ans: **Variable-Length Records**

Variable-length records arise in a database in several ways:

- Storage of multiple items in a file
- Record types allowing variable field size
- Record types allowing repeating fields

Byte string representation uses the technique of attaching a special end-of record symbol(⊥)to the end of each record. Then we can store each record as a string of successive bytes.

Byte string representation has several disadvantages:

- It is not easy to re-use space left by a deleted record
- In general, there is no space for records to grow longer. (Must move to expand, and record may be pinned.)

So this method is not usually used.

Fixed-length representation uses one or more fixed-length records to represent one variable-length record. Two techniques:

- **Reserved space** - uses fixed-length records large enough to accommodate the largest variable-length record. (Unused space filled with end-of-record symbol.)
- **Pointers** - represent by a list of fixed-length records, chained together.

The reserved space method requires the selection of some maximum record length. If most records are of near-maximum length this method is useful.

- Otherwise, space is wasted.

- Then the pointer method may be used .
- Disadvantage is that space is wasted in successive records in a chain as non-repeating fields are still present.
- To overcome this last disadvantage we can split records into two blocks
  - **Anchor block** - contains first records of a chain
  - **Overflow block** - contains records other than first in the chain.
- Now all records in a block have the same length, and there is no wasted space.

Q.73 Describe the different RAID levels. Discuss the choices of the different RAID levels for different applications. **(8)**

Ans: There are various combinations of these approaches giving different trade offs of protection against data loss, capacity, and speed. RAID levels 0, 1, and 5 are the most commonly found, and cover most requirements.

- RAID 0 (striped disks) distributes data across several disks in a way that gives improved speed and full capacity, but all data on all disks will be lost if any one disk fails.
- RAID 1 (mirrored disks) could be described as a backup solution, using two (possibly more) disks that each store the same data so that data is not lost as long as one disk survives. Total capacity of the array is just the capacity of a single disk. The failure of one drive, in the event of a hardware or software malfunction, does not increase the chance of a failure nor decrease the reliability of the remaining drives (second, third, etc).
- RAID 2 It uses memory style redundancy by using Hamming codes, which contain parity bits for distinct overlapping subsets of components.
- RAID 3 It uses a single parity disk relying on the disk controller to figure out which disk has failed.
- RAID 4 It uses block-level data striping.
- RAID 5 (striped disks with parity) combines three or more disks in a way that protects data against loss of any one disk; the storage capacity of the array is reduced by one disk.
- RAID 6 (less common) can recover from the loss of two disks through P + Q redundancy scheme using Reed- Adoman codes

Q.74 Define two-phase locking protocol. **(2)**

Ans: 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.

The two-phase locking protocol

1. Before operating on any row, a transaction must acquire a lock on that row.
2. After releasing a lock, a transaction must never acquire any more locks.

Q75 Differentiate between strict two-phase and rigorous two-phase with conversion protocol **(4)**

Ans: **Strict two-phase** locking holds all its exclusive (write) locks until commit time, once granted. While **Rigorous two-phase locking** is more restrictive than standard 2PL

in that it enforces the rule that no locks can be released by a transaction until after it commits or aborts. The former locks all its items before it starts so once transaction starts, it is in its shrinking phase whereas latter does not unlock any of its items until after it terminates to the transaction is in its expanding phase until it ends.

Q.76 Describe the nested-loop join and block-nested loop join. Compare them. **(8)**

Ans: The block nested- loop join algorithm is as given below:

for each block B_r of relation R do begin
    for each block B_s of relation S do begin
        for each tuple t_r in B_r do begin
            for each tuple t_s in B_s do begin
                test pair (t_r,t_s) to see if they satisfy the join condition
                if they do, add $t_r.t_s$ to the result
            end
        end
    end
end

The nested-loop from algorithm is as below:

for each tuple t_r in relation R do begin
  for each tuple t_s in relation S do begin
    test pair (t_r,t_s) to see if they satisfy the join condition Θ
      if they do, add $t_r.t_s$ to the result
        end
          end

Difference between them:

Nested loop join	Block-nested-loop join
It is expensive as worst case cost, no. of block accesses required is $n_r * b_s + b_r$ where bs and br represent no. of blocks containing n_s and n_r tuples of relations S and R, respective and in best case there will be $b_r + b_s$ block accesses	In worst case no. of block accesses is $b_r * b_s + b_r$. In best case, $b_r + b_s$ block accesses will be required.

Q.77 Two relations R with 60000 tuples and occupying of 300 blocks is to be joined with a relation S with 40000 tuples and occupying 400 blocks. What is the total cost using the two algorithms of (Q 76) above in terms of block transfers. Give both the best case and the worst case figures. **(6)**

Ans:
Nested-loop-join: Worst Case: If R as the outer relation 60000 X 400 + 300 = 24000300 Disk access, if S as the outer relation we need 40000 X 300 + 400 = 12000400 disk accesses
Best Case: 300 + 400 = 700 disk accesses will be required.
Block-nested loop join:
Worst Case: If R is the outer relation, we need 300 * 400 + 300 = 120300 disk access, if S is the outer relation we need 400 * 300 + 400 = 120400 disk access
Best Case: 300+ 400 = 700 disk accesses will be required.

Q.78 Explain the ACID properties of a transaction. **(6)**

Ans: ACID properties are an important concept for databases. The acronym stands for Atomicity, Consistency, Isolation, and Durability.
Atomicity: Either all operations of transaction are reflected properly in the database or none are.
Consistency: Execution of a transaction in isolation (i.e., with no other transaction executing concurrently) preserves the consistency of the database
Isolation: Even though multiple transactions may execute concurrently the system guarantees that for every pair of transaction Ti and Tj, it appears to Ti that either Tj that either Tj finished execution before Ti started or Tj started execution after Ti finished. Thus each transaction is unaware of other transactions executing concurrently in the system.
Durability: After a transaction completes successfully, the changes it has made to the database persist, even if there are system failures.

Q.79 Compare wait-die deadlock prevention scheme with wait-wound scheme. **(4)**

Ans: Wait-Die Scheme
- Based on a nonpreemptive technique.

- If Pi requests a resource currently held by Pj, Pi is allowed to wait only if it has a smaller timestamp than does Pj (Pi is older than Pj). Otherwise, Pi is rolled back (dies).
- Example: Suppose that processes P1, P2, and P3 have timestamps 5, 10, and 15, respectively.
  - If P1 requests a resource held by P2, then P1 will wait.
  - If P3 requests a resource held by P2, then P3 will be rolled back.

Wound-Wait Scheme

- Based on a preemptive technique; counterpart to the wait-die system.
- If Pi requests a resource currently held by Pj, Pi is allowed to wait only if it has a larger timestamp than does Pj (Pi is younger than Pj). Otherwise, Pj is rolled back (Pj is wounded by Pi).
- Example: Suppose that processes P1, P2, and P3 have timestamps 5, 10, and 15, respectively.
  - If P1 requests a resource held by P2, then the resource will be preempted from P2 and P2 will be rolled back.
  - If P3 requests a resource held by P2, then P3 will wait.

Q.80 What are the costs to be considered when a transaction has to be rolled back when recovering from deadlock? **(4)**

Ans: Some transaction will have to rolled back (made a victim) to break deadlock. Select that transaction as victim that will incur minimum cost.

- Rollback -- determine how far to roll back transaction.
  - **Total rollback**: Abort the transaction and then restart it.
  - More effective to roll back transaction only as far as necessary to break deadlock.
- Starvation happens if same transaction is always chosen as victim. Include the number of rollbacks in the cost factor to avoid starvation.

Q.81 Write short notes on

(i) Views in relational algebra.

(ii) Data dictionary.

(iii) Assertions in SQL.

(iv) B^+ tree. **(3.5 x 4=14)**

Ans:

(i) Views in relational algebra:

1. basic expression consists of either
   - A relation in the database.
   - A constant relation.
2. General expressions are formed out of smaller sub expressions using
   - $\sigma_p(E_1)$ select (p a predicate)
   - $\Pi s(E_1)$ project (s a list of attributes)
   - $P_x(E_1)$ rename (x a relation name)
   - $E_1 \cup E_2$ union
   - $E_1 - E_2$ set difference
   - $E_{1x} \times E_2$ cartesian product

(ii) Data dictionary

A data dictionary is a reserved space within a database which is used to store information about the database itself.

A data dictionary may contain information such as:

- Database design information
- Stored SQL procedures
- User permissions
- User statistics
- Database process information
- Database growth statistics
- Database performance statistics

(ii) Assertions in SQL

1. An **assertion** is a predicate expressing a condition we wish the database to always satisfy.
2. Domain constraints, functional dependency and referential integrity are special forms of assertion.
3. Where a constraint cannot be expressed in these forms, we use an assertion, e.g.
   - Ensuring the sum of loan amounts for each branch is less than the sum of all account balances at the branch.
   - Ensuring every loan customer keeps a minimum of $1000 in an account.

(iv) B+ tree

B+ tree is a type of tree which represents sorted data in a way that allows for efficient insertion, retrieval and removal of records, each of which is identified by a *key*. It is a dynamic, multilevel index, with maximum and minimum bounds on the number of keys in each index segment (usually called a 'block' or 'node'). In a B+ tree, in contrast to a B-tree, all records are stored at the lowest level of the tree; only keys are stored in interior blocks.

The primary value of a B+ tree is in storing data for efficient retrieval in a block-oriented storage context. Given a storage system with a block size of *b*, a B+ tree which stores a number of keys equal to a multiple of *b* will be very efficient when compared to a binary search tree (the corresponding data structure for non-block-oriented storage contexts).

Q.82 What is completeness constraint on generalization? Explain the difference between total and partial design constraint. Give an example each. **(6)**

Ans: Completeness Constraints

- total specialization constraint
  - every entity in the superclass must be a member of some subclass in the specialization

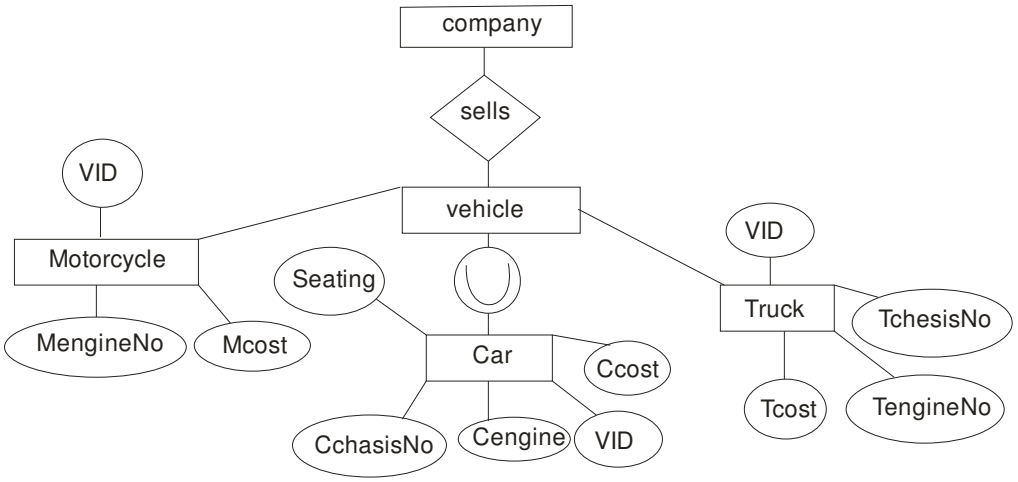
  e.g., {HOURLY_EMPLOYEE, SALARIED_EMPLOYEE}
  - notation: superclass
- partial specialization constraint

an entity may not belong to any of the subclasses

e.g., {SECRETARY, ENGINEER, TECHNICIAN}

Q.83 Design a generalization-specialization hierarchy for a motor-vehicle sales company. The company sells motorcycles which have an engine number and cost; cars which have a chassis number, an engine number, seating capacity and cost; trucks which have chassis number, an engine number and cost. **(10)**

Ans:



Q.84 Define the following operations of relational algebra and give an example each

(i) Division.

(ii) Cartesian product. **(7)**

Ans:

(i) Division:

Let R be a relation having attributes (A1,...Ap,Ap+1,...An) and S having attributes (Ap+1,...An)

DEF: Division

The division of R by S, denoted R ÷ S, gives a new relation Q, the quotient of the division, such that Q has attributes (A1,...Ap) and every tuple of Q must appear in R in combination with every tuple in S.

The division can be obtained from the difference, the Cartesian product and the projection as follows:

R ÷ S = T - Y, where T = $\pi_{A1,...Ap}$ (R) and Y = $\pi_{A1,...Ap}$ ((T X S) - R)

ii) Cartesian product:

The Cartesian product operation does not require relations to union-compatible. It means that the involved relations may have different schemas. Let R & S be relations that may have different schemas.

DEF: **Cartesian Product**

The Cartesian product of relations R and S, denoted R **X** S, is the set of all possible combinations of tuples of the two operation relations. Each resultant tuple consists of all the attributes of R and S.

The resultant relation has

cardinality = (cardinality of R) * (cardinality of S); and

arity = (arity of R) + (arity of S).

For example, in given table, STUDENT **X** MODULE is equal to

STUDENT #	STUDENT-NAME	COURSE	MODULE #	MODULE-NAME
A101	Astor	APPCOM	CS301	IS
A101	Astor	APPCOM	CS302	DBS
B334	Bernstein	INFTEC	CS301	IS
B334	Bernstein	INFTEC	CS302	DBS
J326	Jones	APPCOM	CS301	IS
J326	Jones	APPCOM	CS302	DBS

Q.85 Let R(A, B) and S(A, C) be two relations. Give relational algebra expressions for the following domain calculus expressions.

(i) $\{< a > | \exists b (< a, b > \in r \wedge b = 17)\}$

(ii) $\{< a, b, c > | < a, b > \in r \wedge < a, c > \in s\}$

(iii) $\{< a > | \exists b (\exists c \{< a, b > \in r) \wedge < a, c > \in s)\}\}$ **(9)**

Ans:

(i) $\Pi_A (\sigma_{B=17} (r))$

(ii) $r \bowtie s$

(iii) $\Pi_{r.A} ((r \bowtie s))$

Q.86 Consider the following relations with key underlined

lives (person_name, street, city)
works (person_name, company_name, salary)
located (company_name, city)
manages (person_name, manager_name)

Answer the following using SQL:

(i) Find the names and city of persons who work for manager John.

(ii) Find the names of persons who live in the same city as the company they work for.

(iii) John's manager has changed. The new manager is Anna.

(iv) Susan doesn't work anymore.

(v) Create a view BangWork (person_name, company_name, manager_name) of all people who work in Bangalore in ascending order of person name **(4x3x4)**

Ans:

(i) Select person_name, city from lives, manages
where manager_name = 'John' and lives.person_name = manages.person_name;

(ii) Select person_name from lives works, located where lives.person_name = works.person_name and works.company_name = located.company_name and lives.city = located. city;

(iii) Update manages
set manager_name = 'Anna'
where manager_name = 'John';

(iv) delete from works where person_name = 'Susan';

(v) create view BangWork as select person_name, company_name, manager_name from work where city = 'Bangalore' order by person_name;

Q.87 What are the problems if one were not to normalize? When do these problems surface? **(2)**

Ans: **Database normalization**, sometimes referred to as *canonical synthesis*, is a technique for designing relational database tables to minimize duplication of information and, in so doing, to safeguard the database against certain types of logical or structural problems, namely data anomalies. For example, when multiple instances of a given piece of information occur in a table, the possibility exists that these instances will not be kept consistent when the data within the table is updated, leading to a loss of data integrity. A table that is sufficiently normalized is less vulnerable to problems of this kind, because its structure reflects the basic assumptions for when multiple instances of the same information should be represented by a single instance only.

Q.88 Consider the relation
Book (accno, author, author_address, title, borrower_no, borrower_name, pubyear)
with the following functional dependencies
accno → title accno → pubyear
author → accno
accno → author author → author_address
accno → borrower_no borrower_no → borrower_name
(i) Normalize the relation. Clearly show the steps. **(6)**
(ii) For each decomposed relation identify the functional dependencies that apply and identify the candidate key. **(8)**

Ans:
(i) **Book1** (accno, title, author, borrow_no, pubyear, author_address)
Book2 (borrower_no, borrower_name)
Book3 (author, account)
ii) a) Functional Dependencies for "Book1" relation are:
accno → title
accno → author
accno → borrow_no
accno → pubyear
accno → author_address
b) F.Ds for "Book2" relation are:
borrow_no → borrower_name
c) F.Ds for "Book3" relation are:
author → account_no

Q.89 Describe sequential file organization. Explain the rules for
(i) inserting a new record.
(ii) Deleting an existing record. **(8)**

Ans: A sequential file contains records organized by the order in which they were entered. The order of the records is fixed.
Records in sequential files can be read or written only sequentially.

After you have placed a record into a sequential file, you cannot shorten, lengthen, or delete the record. However, you can update (`REWRITE`) a record if the length does not change. New records are added at the end of the file.
If the order in which you keep records in a file is not important, sequential organization is a good choice whether there are many records or only a few. Sequential output is also useful for printing reports.

(i) **Inserting a new record:** Insertion poses problems if no space where new record should go. If space, use it, else put new record in an **overflow block.**

(ii) **Deleting an existing record:** Deletion can be managed with the pointer chains

Q.90 Define and differentiate between ordered indexing and hashing. **(8)**

Ans: **Ordered indexing:** To gain fast random access to records in a file, we can use an index structure. Each index structure is associated with a particular search key. An ordered index stores the values of the search keys in sorted order and associates with each search key records that contain it. The records in the indexed file may themselves be stored in some sorted order. A file may have several indices on different search keys.
Hashing: It also provides a way of constructing indices. In hashing, the address of the disk lock containing a desired record directly is computed by a function on the search-key value of the record.
Differentiate between ordered indexing and hashing: Ordered indexing is stored in sorted order while in **Hashing** search keys are distributed uniformly across "buckets" using 'hash function'

Q.91 How do you compute the query cost for the following?

(i) Selection with linear search.

(ii) Negation. **(9)**

Ans:

(i) Scan each file block and test all records to see whether they satisfy the selection condition

- Cost estimate (number of disk blocks scanned) = b_r
  (b_r denotes number of blocks containing records from relation r)
- Selections on key attributes have an average cost $b_r/2$,
  but still have a worst-case cost of b_r
- Linear search algorithm can be applied to any file, regardless of
  - Ordering of records in the file
  - Availability of indices
  - Nature of the selection operation

(ii) Negation: In the absence of nulls the result of a selection $\sigma_{7\Theta}$(r) is simply the tuples of r that are not in σ_{Θ}(r). The number of tuples in $\sigma_{7\Theta}$(r) therefore estimated to be n (r) minus the estimated number of tuples in σ_{Θ} (r). To account for nulls, the number of tuples for which the condition $_{\Theta}$ would evaluate to unknown will be estimated and subtracting that number from above estimate ignoring nulls.

Q.92 Explain the statement 'Projection operation distributes over the union operation'. Give an example. **(4)**

Ans: $\Pi_L(E_1 \cup E_2) = (\Pi_L(E_1)) \cup (\Pi_L(E_2))$
This says that result obtained by projecting on an attribute set A, after taking union of two relations is same as union of two relations each of which is a projection of original relations on A.

Q.93 Explain pipelining. **(3)**

Ans: In order to explain pipelining in simple terms, think of it as breaking down processor functions into smaller and smaller parts. For example, the act of drinking milk can be broken down to opening the refrigerator, getting the milk carton, getting a glass from the cupboard, opening and pouring the milk, lifting the glass to your mouth, and pouring the milk inside. Each one of these subprocesses is much shorter than the entire process of drinking milk.A processor functions similarly. Instructions progress through, and as they do, they accomplish smaller parts of the overall execution. The more pipeline stages you have, the smaller each of these sub-processes are. Designers can design these sub-processes to be very fast, and efficient. The frequency of a processor advances each of these sub-processes to the next stage, so smaller sub-processes mean less time to complete them, and thus higher frequencies

Q.94 Explain the rules for creating a labelled precedence graph for testing view serializability. **(8)**

Ans: A schedule S is view serializable if it is view equivalent to a serial schedule

- Every conflict serializable schedule is also view serializable
- A schedule which is view serializable but not conflict serializable:

Testing for Serializability

- Consider some schedule of a set of transactions T1, T2, ..., Tn
- Precedence graph is a direct graph where the vertices represent the transactions
- We draw an arc from T_i to T_j if the two transaction conflict and T_i accessed the data item earlier on which the conflict arose
- We may label the arc by the item on which the conflict arose

A schedule is conflict serializable if and only if its precedence graph is acyclic

- If precedence graph is acyclic, the serializability order can be obtained by a topological sorting of the graph.
- The problem of checking if a schedule is view serializable falls in the class of *NP*-complete problems
- However practical algorithms that just check some sufficient conditions for view serializability can still be used

Q.95 Explain the difference between the three storage types – volatile, non volatile and stable. **(8)**

Ans: **Volatile storage:** if storage media loses data when power goes off, it is known as volatile storage media eg, RAM. It is the fastest between the three in terms of data access time.

Non-volatile storage: If storage media retains data even when power goes off, it is known as non-volatile storage media. Eg: hard disk. It is faster than stable storage but slower than volatile storage.

Stable Storage: Information residing in stable storage is never lost (theoretically). A natural catastrophe may result in a loss otherwise the probability of data loss is negligible. Eg, using multiple hard disks as in the case of RAID technology. This is the slowest of all storage media mentioned above.

Q.96 How does the system recover from a crash? **(8)**

Ans: Not all problems that cause computer crashes are severe. If you carefully analyze and try to find out probable causes, you usually have a good chance to quickly and completely recover your system.

Most of the time, it is quite simple to point out the main reason behind a problem. Suppose that your system starts freezing when you restart after installing a new hardware or software or if you have made some changes to system configuration, then you can safely say that the new device, driver, or software is the reason behind the problem.

Windows XP comes equipped with a number of options that you can use to troubleshoot and repair your PC problems. Here, we are going to discuss a few common options that can help you recover your Windows XP system after a crash.

Safe Mode

If your system frequently hangs during startup, then you must try to start your system in Safe Mode. To do this, restart your PC and press F8 as soon as your system starts booting up. Now, scroll down the displayed options using arrow keys and select one of the three Safe Mode options displayed. You can perform a number of activities to fix your computer in this mode. You can uninstall the suspect driver or software, and change configuration settings.

Last Known Good Configuration

The Last Known Good Configuration option also appears when you press the F8 key during system boot up. This option helps you in reversing any driver and hardware changes that you would have done since your last successful startup.

Recovery Console

In case your system is unable to start up even in the Safe Mode, then you may have to use the Recovery Console. You can start the Recovery Console by booting up from the Windows XP installation CD, and selecting the Repair option. This utility helps you delete corrupted software and driver files to stop services that might be causing frequent system crashes.

System Restore

System Restore is a boon for all computer users. The tool monitors all changes on your system, periodically takes snapshots of system files and settings, and stores original files in compressed form in a protected location on the hard drive. To perform system restore, log on as an Administrator, and open the Start menu. Next, select All Programs-Accessories-System Restore command to display the System Restore dialog box. Follow the simple instructions in the dialog box to rollback your computer to the most current system checkpoint.

Reinstallation

If none of the options above work, then you may have to reinstall the operating system. If you are careful with this step, you can easily reinstall the operating system without effecting your system preferences and settings.

Q.97 Describe the storage structure of indexed sequential files and their access method. **(7)**

Ans: Index provides a lookup capability to quickly reach the vicinity of the desired record.

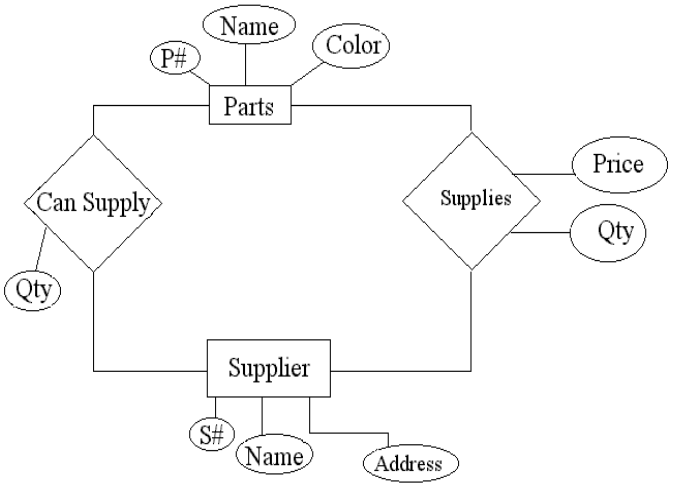
- Contains key field and a pointer to the main file
- Indexed is searched to find highest key value that is equal to or precedes the desired key value
- Search continues in the main file at the location indicated by the pointer.

If an index contains 1000 entries, it will take on average 500 accesses to find the key, followed by 500 accesses in the main file. Now on average it is 1000 accesses.

- New records are added to an overflow file
- Record in main file that precedes it is updated to contain a pointer to the new record
- The overflow is merged with the main file during a batch update
- Multiple indexes for the same key field can be set up to increase efficiency.

ISAM (Indexed Sequential Access Method) is a file management system developed at IBM that allows records to be accessed either sequentially (in the order they were entered) or randomly (with an index). Each index defines a different ordering of the records. An employee database may have several indexes, based on the information being sought. For example, a name index may order employees alphabetically by last name, while a department index may order employees by their department. A key is specified in each index. For an alphabetical index of employee names, the last name field would be the key.

Q.98 Map the following ER diagram to a relational database. Give the relation names and attributes in them. Also mention the primary key and foreign keys if any for each table. **(10)**



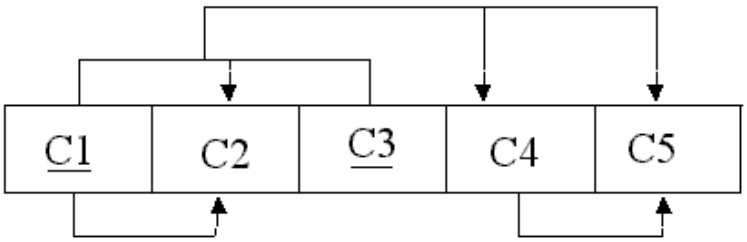
Ans:

Relation Name	**Attributes**	**Primary key**	**Foreign Key (s)**
Parts	P#, Name, color	P#	NIL
Supplier	S#, Name, Address	S#	NIL
Can_supply	P#, S#, QTY	P#,S#	P# references Parts.P# S# references Supplier. S#
Supplies	P#, S#, QTY, Price	P#, S#	P# references Parts. P# S# references Supplier.S#

Q.99 What are the three data anomalies that are likely to occur as a result of data redundancy? Can data redundancy be completely eliminated in database approach? Why or why not? **(5)**

Ans: The most common anomalies considered when data redundancy exists are: update anomalies, addition anomalies, and deletion anomalies. All these can easily be avoided through data normalization. Data redundancy produces data integrity problems, caused by the fact that data entry failed to conform to the rule that all copies of redundant data must be identical. The data redundancy cannot be totally removed from the database, but there should be controlled redundancy. Sometimes redundancy is allowed to overcome the problem of excessive join operations for computing queries.

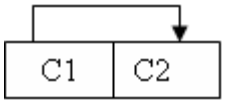
Q.100 Given the dependency diagram shown in the following figure, (the primary key attributes are underlined)



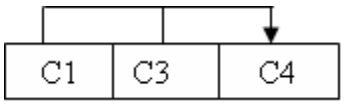
(i) Identify and discuss each of the indicated dependencies.
(ii) Create a database whose tables are atleast in 3NF, showing dependency diagram for each table **(4+5)**

Ans:
(i) C1 → C2 represents a *partial dependency*, because C2 depends only on C1, rather than on the entire primary key composed of C1 and C3.
C4 → C5 represents a *transitive dependency*, because C5 depends on an attribute (C4) that is not part of a primary key.
C1, C3 → C2, C4, C5 represents a functional dependency, because C2, C4, and C5 depend on the primary key composed of C1 and C3.
ii) After decomposition into 3NF, relations are

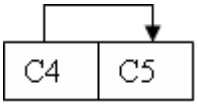
1) R1 (C1, C2) with FD {C1→ C2)} and functional dependency diagram:



2) R2 (C1,C3, C4) with FD {C1,C3 →C4) and functional dependency diagram:



3) R3 (C4, C5) with FD{C4 → C5} and functional dependency diagram:



Q.101 Consider the following relations with underlined primary keys

Product(P_code, Description, Stocking_date, QtyOnHand, MinQty, Price, Discount, V_code)

Vendor(V_code, Name, Address, Phone)

Here a vendor can supply more than one product but a product is supplied by only one vendor. Write SQL queries for the following :

(i) List the names of all the vendors who supply more than one product.

(ii) List the details of the products whose prices exceed the average product price.

(iii) List the Name, Address and Phone of the vendors who are currently not supplying any product. **(3 x 3)**

Ans: **(i)** Select name from Vendor where V_code in (Select V_code from Product group by V_code having count (V_code) >1)

(**ii**) Select * from Product where price > (select avg(Price) from product)

(iii) Select Name, Address, Phone from Vendor v where not exists (select * from Product p where p.V_code = v.V_code)

Q.102 Define the domain relational calculus. **(5)**

Ans: In DRC, *queries* have the form: { <x1,x2,--------xn> | P(x1,x2,--------xn)}

where each X_i is either a domain variable or constant, and p(<X1, X2, ..., Xn>) denotes a DRC *formula*. The result of the query is the set of tuples X_i to X_n which makes the DRC formula true.

This language uses the same operators as tuple calculus, the logical connectives (and), (or) and ¬ (not). The existential quantifier and the universal quantifier can be used to bind the variables.

Q.103 Consider the transactions t1, t2 and t3 and a schedule S given below.
$S: \mathbf{read}_1(A); \mathbf{read}_2(B); \mathbf{write}_1(C); \mathbf{read}_3(B); \mathbf{read}_3(C); \mathbf{write}_2(B); \mathbf{write}_3(A)$ Where the subscript denotes the transaction number. Assume that the time stamp of t1<t2<t3. Using time-stamp ordering scheme for concurrency control find out if the schedule will go through. If there is to be a rollback, which transaction(s) will be rolled back? **(8)**

Ans: **Schedule:**

t_1	t_2	t_3
read(A)		
	read(B)	
write (C)		
		read (B)
		read (C)
	*** write(B)**	
		write (A)

Since this write in t_2 is after a younger transaction (t_3) has read the value of B, therefore t_2 will be rolled back.

Q.104 Consider the following database with primary keys underlined
Project(P_No, P_Name, P_Incharge)
Employee(E_No, E_Name)
Assigned_To(P_No, E_No)
Write the relational algebra for the following :
(i) List details of the employees working on all the projects.
(ii) List E_No of employees who do not work on project number DB2003. **(6)**

Ans:

(i) $\Pi_{E\_No,\ E\_Name}$ (Project ⋈Employee ⋈Assigned) ÷ Π_{E_Name} (Employee)

(ii) Π_{E_No} (Assigned _To) - Π_{E_No} ($\sigma_{P\_No\ =\ \text{“DB2003”}}$ (Assigned_To)

Q.105 The following represents the sequence of events in a schedule involving transactions T1, T2, T3, T4 and T5. A,B, C, D, E, F are items in the database.

T2	:	Read	B
T4	:	Read	D
T2	:	Read	E
T2	:	Write	E
T3	:	Read	F
T2	:	Read	F
T1	:	Read	C
T5	:	Read	A
T5	:	Write	A
T1	:	Read	E
T5	:	Read	C
T3	:	Read	A
T5	:	Write	C
T2	:	Write	F
T4	:	Read	A

Draw a wait-for-graph for the data above and find whether the transactions are in a deadlock or not? **(8)**

1. T2 : Read B
The data item B is locked by T2 and no change in the wait-for-graph

2. T4 : Read D
The data item D is locked by T4 and no change in the wait-for-graph

3. T2 : Read E
The data item E is locked by T2 and no change in the wait-for-graph

4. T3 : Write E
The data item E is unlocked by T2 and no change in the wait-for-graph

5. T3 : Read F
The data item F is locked by T3 and no change in the wait-for-graph

6. T2 : Read F
Now T2 wants to lock F, which is already locked by T3 (in step 5) so insert an edge from T2 to T3 in the wait-for-graph.

7. T1 : Read C
The data item C is locked by T1 and no change in the wait-for-graph.

8. T5 : Read A
The data item A is locked by T5 and no change in the wait-for-graph.

9. T5 : Write A
The data item B is locked by T5 and no change in the wait-for-graph.

10. T1 : Read E
The data item E is locked by T1 and no change in the wait-for-graph.

11. T5 : Read C
Now T5 wants to lock C, which is already locked by T1 (in step 7) so insert an edge from T5 to T1 in the wait-for-graph.

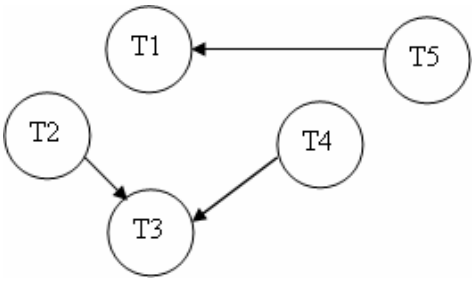
12 T3 : Read A
The data item A is locked by T3 and no change in the wait-for-graph.

13. T5 : Write C
The transaction cannot proceed further as it was not able to lock data item C (in step 11) so the transaction has to wait, hence there is no change in the wait-for-graph.

14. T2 : Write F

The transaction cannot proceed further as it was not able to lock data item F (in step 6) so the transaction has to wait, hence there is no change in the wait-for-graph.

15. T4 : Read A
Now T4 wants to lock A, which is already locked by T3 (in step 13) so insert an edge from T4 to T3 in the wait-for-graph.
Thus finally the wait-for graph will be as follows:



Since there is no cycle in the wait-for-graph, the system is not in a deadlock.

Q.106 Write short notes on any

(i) Disadvantages of file based systems.
(ii) Query-by-Example.
(iii) B-tree. **(3.5×3=14)**

Ans:

(i) Disadvantages of file based systems:

1. duplication of data
2. data integrity problem
3. limited data sharing
4. lengthy processing time

(ii) Query-by-Example:

Query by Example (QBE) is a method of query creation that allows the user to search for documents based on an example in the form of a selected text string or in the form of a document name or a list of documents. Because the QBE system formulates the actual query, QBE is easier to learn than formal query languages, such as the standard Structured Query Language (SQL), while still enabling powerful searches.

(iii) B-tree:

B-tree is a tree data structure that keeps data sorted and allows searches, insertions, and deletions in logarithmic amortized time. It is most commonly used in databases and filesystems.

In B-trees, internal (non-leaf) nodes can have a variable number of child nodes within some pre-defined range. When data is inserted or removed from a node, its number of child nodes changes. In order to maintain the pre-defined range, internal nodes may be joined or split. Because a range of child nodes is permitted, B-trees do not need re-balancing as frequently as other self-balancing search trees, but may waste some space, since nodes are not entirely full. The lower and upper bounds on the number of child nodes are typically fixed for a particular implementation. For example, in a 2-3 B-tree (often simply referred to as a **2-3 tree**), each internal node may have only 2 or 3 child nodes.

Q.107 Supreme Products manufactures products like pressure cookers, cookwares, water purifiers, food processors etc. The company markets its products to wholesalers all over the country and dealers sell them to customer. The company has five regional offices and many sales persons are attached to regional offices. Salespersons contact dealers and explain about products, incentives offered, training programs for wholesalers and demo for customers etc. Dealers place orders with the salespersons 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 sales of different products, performance of salespersons and orders from wholesalers.

Do the following :

(i) Identify entities, attribute and relationships giving functionalities and draw E-R diagram for the system.

(ii) Convert this to relational tables explaining logic involved. **(5+5)**

Ans: The ER Diagram for the given problem is : The various entities for the given problem will be

WHOLESALER
REGIONALOFFICE
SALESPERSON
PAYMENT
DEALER
ORDER
ITEM

CUSTOMER

The various assumptions for the given system are :

- A salespersons is attached to only one regional office; however a regional office may have several salespersons under it.
- A dealer can place several orders but an order will be placed by a single dealer.
- A dealer will make payment to only one salesperson of his/her area but a salesperson can collect payments from several dealers of his/her area.
- An order may contain several items but an item can be placed only in one order.
- A wholesaler can appoint several dealers but a dealer will be only under one wholesaler.
- Many dealers can sales their products to several customers and several customers can purchase the products from many dealers.

The relational schema of the above ER Diagram is given here :

WHOLESALER:

WS_ID	WS_NAME	WS_ADD

REGIONAL OFFICE:

RO_ID	RO_NAME	RO_ADD

SALESPERSON:

SALES_ID	SALES_NAME	SALES_ADD	CATEGORY

PAYMENT

AMIT_PAYABLE	AMT_PAID	DATE

DEALER:

DEAL_ID	DEAL_NAME	DEAL_ADD

ORDER:

ORDER_ID	ITEM_ID	DATE	QTY

ITEM:

ITEM_ID	ITEM_NAME	STOCK	PRICE

CUSTOMER:

CUST_ID	CUST_NAME	CUST_ADD

For relationship, some of the relations will be as follows:

APPOINTS:

WS_ID	DEAL_ID

SALES:

CUST_ID	DEAL_ID
WS_ID	DEAL_ID
SALES:CUST_ID	DEAL_ID

PLACE:

DEAL_ID	ORDER_ID

ATTACHED:

SALES_ID	RO_ID

Some reports which can be proposed from the above system are :

- List the names of all salespersons with maximum sales from all regional offices.
- Name of the dealer who have maximum customers under him/her.
- All the dealers who have payments due against them.
- Lists the highest selling items.
- List of all regional offices along with their dealers and salespersons

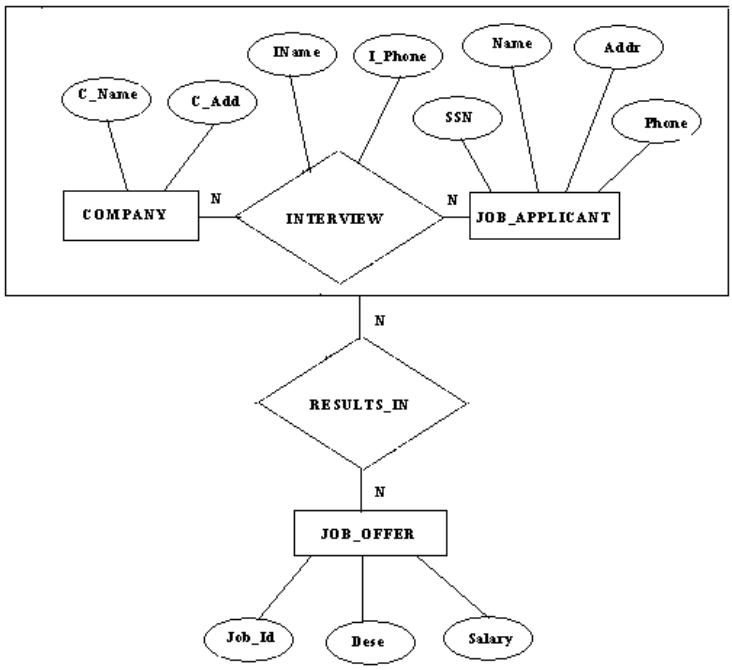
Q.108 Give examples of :

(i) A many – to – many relationship in which one of the participants is another relationship;

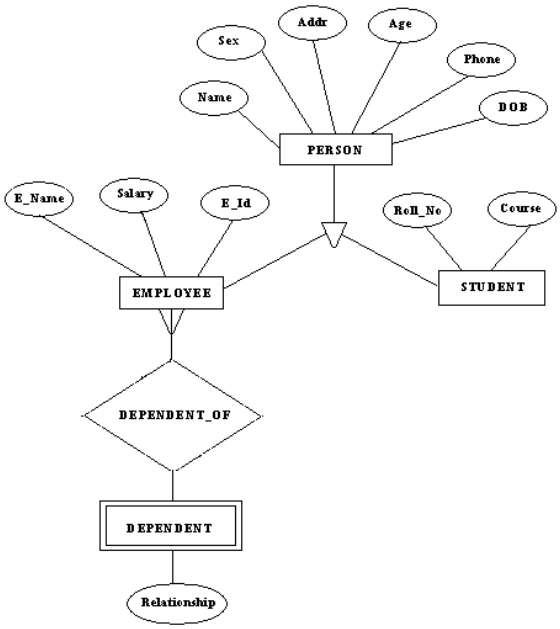
(ii) A subtype that has an associated weak entity that dose not apply to the super type

Ans:

(i) A many-to-many relationship in which one of the participants is another relationship;



(ii) A subtype that has an associated weak entity that does not apply to the supertype.



PERSON can b*e* EMPLOYEE or STUDENT. EMPLOYEE has DEPENDENT that works under him. DEPENDENT is a weak entity set because many dependents can have same Name, Age, Sex, etc.

Q.109 What is meant by heuristic optimisation? Discuss the main heuristics that are applied during query optimisation. **(6)**

Ans:

In heuristic opimization, heuristics are used to reduce the cost of optimization instead of analyzing the number of different plans to find out the optimal plan. For example. A analyzing optimizer would use the rule 'perform selection operation as early as possible' without finding out whether the cost is reduced by this transformation. Heuristics approach usually helps to reduce the cost but not always. The heuristics that are applied during query optimization are:

- Pushes the selection and projection operations down the query tree
- Left-deep join trees- convenient for pipelined evaluation
- Non- left-deep join trees

Q.110 Consider the following tables :
customer (c_id, c_name, c_address)
branch (br_name, br_city, assets)
account (c_id, act_no, br_name, balance)

(i) Customers who have accounts in all branches of Bhopal.

(ii) Customers who have accounts in branches with assets more than 50 crores.**(3X2)**

Ans:

The customers who have accounts in branches of Bhopal.

$$\prod_{c\_name}(\sigma_{br\_name='BHOPAL'}(\text{BRANCH } br\_name \bowtie \textit{ACCOUNT} \bowtie \textit{CUSTOMER } c\_id))$$

$$\prod_{c\_name}(\sigma_{assets>500000000}(\text{BRANCH } br\_name \bowtie \text{ACCOUNT} \bowtie \text{CUSTOMER } c\_id))$$

Q.111 Consider the following tables.

EMP (emp_no, name, salary, supervisor_no, sex_code, dept_code)

DEPT (dept_cd, dept_name)

Write down queries in SQL for getting following information:

(i) Employees getting more salary than their supervisor.

(ii) Department name and total number of employees in each department who earn more than average salary for their department.

(iii) Department(s) having maximum employees earning more than 25000.

(iv) Name of employee(s) who earn maximum salary in their organization. **(4x3)**

Ans:

(i) Employees getting more salary than their supervisor.

Again it will be solved by using self join concept.

```
SELECT E1.name, E2.name Supervisor
                    FROM EMPAS E1, EMP AS E2
                    WHERE E1.supervisor_no = E2.emp_no
                            AND E1.salary>E2.salary;
```

(ii) Department name and total number of employees in each department who earn more than average salary for their department.

This will use the correlated query concept.

```
SELECT dept_name, COUNT(*)
        FROM EMP E, DEPT
                WHERE E.dept_code=DEPT.dept_cd
                        AND salary>(SELECT AVG(salary)FROM EMP
                                WHERE EMP.dept-code=E.dept_code)
                                        GROUP BY dept_name;
```

(iii) Departments having maximum employees earning more than 25000.

```
SELECT dept_name, COUNT(*)
        FROM EMP E,DEPT
                WHERE E.dept_code=DEPT.dept_cd AND salary>25000
                        GROUP BY dept_name
                                HAVING COUNT(*)>=ALL(SELECT COUNT(*)
                FROM EMP GROUP BY dept_code);
```

(iv) Name of employee(s) who earn maximum salary in the organization.

```
SELECT name FROM EMP
        WHERE salary=(SELECT MAX (salary) FROM EMP);
```

Q.112 Let R = (A,B,C,D) and F be the set of functional dependencies for R given by {A→B, A→C, BC→D} **(4)**

Prove A→D.

Ans: Given set of functional dependencies for a relation R is {A→B, A→C, BC→D}

By using Armstrong Axioms, named projectivity, we can show that

A→BC (as A→B, A→C)

Since BC→D, so by transitivity rule,

A→BC and BC→D means A→D

Hence Proved.

Q113 **a.** Consider a relation TENANT (NAME, CITY, STREET#, APT#, APT_TYPE, RENT, LANDLORD, ADDRESS) where following functional dependencies hold

APT#STREET#CITY □ADDRESS

ADDRESS→APT_TYPE

NAME ADDRESS →RENT

LAND_LORD APT_TYPE→RENT

(i) Are the following relation schemes in 3NF?

APARTMENT (APT_TYPE, ADDRESS, RENT)

DWELLER (NAME, ADDRESS, RENT)

(ii) What updating and insertion anomalies do you foresee in TENANT, APARTMENT & DWELLER relations.

Do the APARTMENT & DWELLER relations have lossless join?

(iii) Find a key of this relation. How many keys does TENANT have? **(2x4)**

(iv) Find a key of this relation. How many keys does TENANT have? **(2x4)**

b. Find the decomposition of TENANT into 3NF having lossless join and preserving dependencies.

Ans:

(a) (i) Consider the relation APARTMENT (E,H,F)

Given H→E

The key in this relation will be (HF).

This relation will not be in 3NF as it is not in 2NF. The reason is that there exists a partial function dependency H→E, where (HF) is the key attribute. To make it in 2NF, decomposition will be R1(E,H) and R2(H,F) means E1(APT_TYPE, ADDRESS) and R2 (ADDRESS, RENT)

For another relation DWELLER (A,H,F), the key will be (AH) as AH → F. This relation does not have any partial as well as transitive functional dependency, therefore in 3NF.

(ii) TENANT relation is having redundant data, which needs updation and deletion in all corresponding records if an updation or deletion is made for one value. TENANT relation is not in 2NF. As APT_TYPE is functionally dependent on ADDRESS, therefore every time a apartment type is changed, its corresponding address will be changed every where, otherwise leading to inconsistent state. Moreover if an address is having only one record in the relation and it gets deleted then the information about the apartment type will also be lost. As the DWELLER relation is normalized, therefore there will not be any anomalies in this relation.

(iii) A relation scheme R can be decomposed into a collection of relation schemes to eliminate anomalies contained in the original scheme R. However any such decomposition required that the information contained in the original relation be maintained or in other words the joining of the decomposed relations must give the same set of tuples as the original relation. To check whether the decomposition is lossless or not, the following algorithm is used :

A decomposition of relation schema R<(X,Y,Z), F>into R1<(X,Z), F2> is lossless if the common attribute X of R1 and R2 form a super key of at least one of these i.e. X→Y or X → Z.

Applying the algorithm for the two relations, APARTMENT and DWELLER, the common attribute is (H,F), which is a key of APARTMENT RELATION. Therefore the two relations given have lossless join.

(iv) To find the key in the relation, following rule will be applied :

Given a relation scheme R {A1, A2, A3,…..An} and a set of functional dependencies F, a key of R is a subset of R such that K → A1, A2, …. An is in the closure of F and for any Y → K, Y → A1, A2, ….. An is not in closure of F.
For TENANT relation, we have to find a K such that al other attributes are functionally dependent on it but not on any its subset. The key in this case will be (ABCDG) as all other attributes are FD on it but not on any subset of it.

b. The key for the TENANT relation will be (A,B,C,D,G). Since H→E and H is FD on key i.e. (A,B,C,D,G) therefore a transitive FD exists here. Therefore the relation will be decomposed in R1(A,B,C,D,F,G,H) and R2(H,E). This decomposition is lossless decomposition as the common element in both the relation is H and H→E (as definition given in (iii).

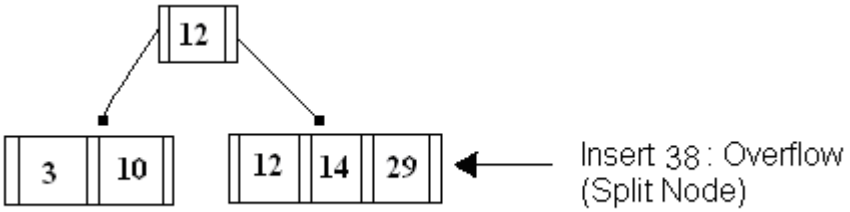
Q.114 Construct a B+ tree for the following set of key values under the assumption that the number of key values that fit in a node is 3. .Key values (3,10,12,14,29,38,45,55,60,68) Show the step involve in the following insertions (use your algorithm)
Insert 11, insert 30. **(10)**

Ans:

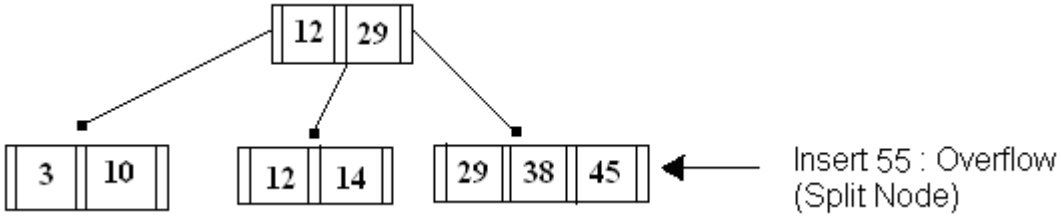
Construct a B+ tree for the following set of key values
n=4
K = n-1 =3
Max = 3 values allowed in leaf node
Min = n-1 | (n-1)/2 | = 2 values allowed in leaf node

Insert 3, 10 , 12

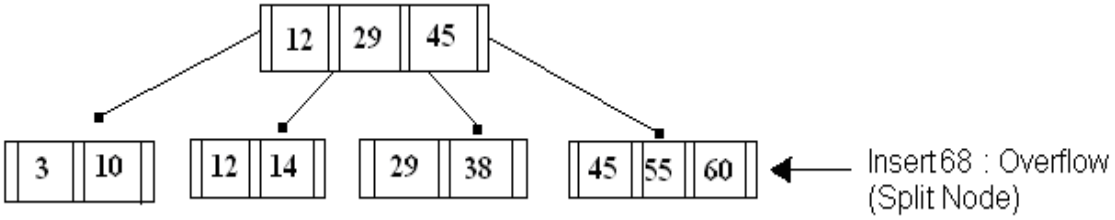
Insert 14,19



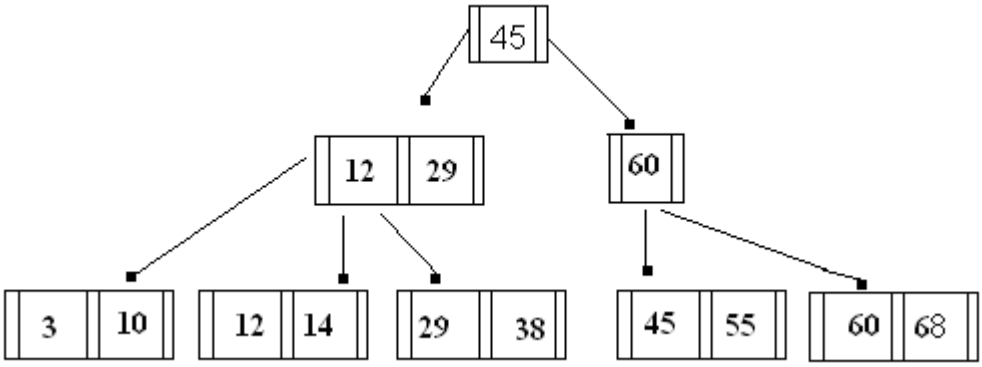
Insert 38,45



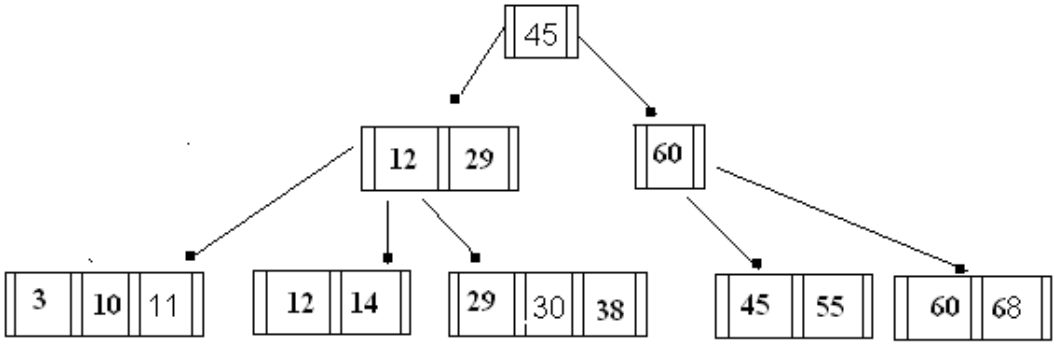
Insert 55,60



Insert 68



Insert 11 and 30



Q.115 Explain the concept of two phase locking and show that it guarantees serialisability. **(6)**

Ans: It concerns the positions of locking and unlocking operations in every transaction. In this protocol each transaction issue lock and unlock requests in two phases.

Growing Phase : In this phase a transaction may obtain locks but may not release any lock. Here the . of locks increases from zero to the maximum for the transaction.

Shrinking Phase : In this phase a transaction may release locks but may not obtain any new locks. Here the number. of locks decreases from maximum to zero.

The basic concept of this protocol is that, all the locks are first acquired before any of the locks released. No new locks can be granted after the releasing of any lock. There are several versions of this protocol

(i) **Static (or conservative) Two Phase Locking:** In this scheme, all the data items are locked before any operations on them and are released only after the last operation performed on any data item.

X(X)
X (Y)
Read (X)
Write (X)
Read (Y)
Y = Y+N
Write Y
Unlock (X)
Unlock (Y)

In static 2PL, requests from other transactions for data items locked by the previous transaction will be delayed unnecessarily, thus causing a serious impact on system performance.

(ii) **Dynamic Two-Phase Locking:** Here a transaction locks a data item immediately before any operation is applied on the data item. After finishing all the operations on all the data items, it releases all the locks. An example of dynamic 2PL is given below.

X(X)

Read (X)

X=X+M

Write (X)

X (Y)

Read (Y)

Y=Y+N

Write Y

Unlock (Y)

Unlock (X)

(iii) Strict Two-Phase locking : A transaction T does not release any of its exclusive (X) locks until that transaction commits or aborts. In this way no other transaction can access the item that is written by T unless the transaction T commits. This is a better technique than the two discussed.

In 2PL, a transaction may not be able to release its locks even after using it. The data item will remain locked until the transaction finishes all the actions on all the data items. If every transaction in a schedule follows 2PL then conflict serializability can be ensured. This is because if the schedule is not serializable, the precedence graph for schedule S consisting of transactions T1, T2 ….. Tn will have a cycle … Suppose that a cycle consists of T1, T2 ….. Tn ,T1. This means that a lock operation by T2 is followed by an unlock operation by T1; A Lockoperation T3 is followed by an unlock operation by T2 … and so on. However this is a contradiction of the assertion that T1 is using two phase locking protocol. Thus the assumption that graph is having a cycle is wrong and hence S is serializable.

Q.116 Explain recovery process after system failure using checkpoint. **(6)**

Ans:. Checkpoint scheme is an additional component of the log based recovery system. This scheme issued to limit the volume of log information that has to be handled and processed in the event of a system failure involving the loss of volatile data. The checkpoint operation is performed periodically copies log information onto the stable storage.

At each checkpoint, following information is kept.

- A start-of-checkpoint record, which gives the identification that it is a checkpoint.
- All log information from the buffers in the volatile storage is copied to the log on stable storage.
- All database updates from the buffers in the volatile storage are propagated to physical database.
- An end-of-checkpoint record is written and the address of the checkpoint record is saved on a file accessible to the recovery routine on start-up after a system crash.

When failure do occur, it is often possible to resume processing from the most recent checkpoint. All transactions that were saved before the checkpoint time need not be considered for the recovery operation. All transactions that were started before the checkpoint time but were not committed at that time are placed in an undo list, which is a list of transactions to be undone.

Now the recovery system scans the log in a backward direction from the time of the system crash. If it finds that a transaction in the undo list has committed, that transaction is removed from the undo list and placed in the REDO list, which contains all the transactions which have to be redone. Thus a system crash occurring during the checkpoint operation requires recovery to be done using the most recent previous checkpoint.

Q.117 What are the advantages of the tree locking protocol over the two-phase locking protocol? Explain the phantom phenomenon. Why this phenomenon may lead to an incorrect concurrency execution despite the use of a protocol that ensures serializability?

(12)

Ans: A tree locking protocol can be defined as follows :

- All locks are exclusive locks.
- Locking a node does not automatically lock any descendent of the locked node.
- The first item locked by a transaction can be any data item including the root node.
- Except for the first data item locked by a transaction, a node cannot be locked by a transaction unless the transaction has already successfully locked its parent.

- No items are locked twice by a transaction; thus releasing a lock on a data-item implies that the transaction will not attempt another lock on the data item.

This protocol is having an advantage over two-phase protocol in the sense that in tree locking protocol, a data item can be released earlier by a transaction if the data item and its unlocked descendants in the sub-tree are not required by the transaction. In this way a greater amount of concurrency is obtained.

Phantom Phenomenon : The phantom phenomenon can be explained by using the following example. Consider a transaction T1 that executes the following SQL statement on the employee relation.

```
SELECT SUM (sal) FROM employee
    WHERE deptno=10;
```

In this statement, transaction T1 requires to access all the tuples of relation employee pertaining to department number 10.

Problem will be encountered if there is another transaction, which is run to reflect the receive of SUM(sal). Let T2 be another transaction which issues an insert statement to the same relation, employee :

```
INSERT INTO employee
    VALUES
        (100, 'Ashok', 5000, 'A-34 Ashoka Road', 10):
```

Let S be the schedule involving T1 and T2. These two transactions can conflict from following two reasons:

- If T1 uses the tuple newly inserted by T2 in computing SUM(sal), then T1 read a value written by T2. Thus in a serial schedule equivalent to S, T2 must come before T1.
- If T1 does not use the newly created tuple by T2 in computing SUM(sal) then in a serial schedule equivalent to S, T1 must come before T2.

In second case, T1 and T2 do not access any tuple in common, yet they conflict with each other. Here locking of records did not prevent the creation of a new tuple, which was created after the existing records have been locked. If concurrency control is performed at the tuple granularity, this conflict would go undetected. This phenomenon is called as Phantom Phenomenon. The problem could be prevented if the locking of records also prevents the addition of such phantom records. The locking of a record belonging to a record type must guarantee that no new record occurrences of the record type can be added until the lock is released.

Q.118 Prove that a relation which is 4NF must be BCNF. **(4)**

Ans Let R be in 4NF. Assume it is not in BCNF. Therefore, there exists an FD $X \rightarrow Y$ in R such that x is not a super key. But by the rule M1 $X \rightarrow Y \models x \rightarrow\rightarrow Y$. Again x here is not a super key. This contradicts the fact that R is in 4NF. Therefore, our assumption is false. Every R in 4NF must be in BCNF.

Q.119 Explain ill effects of concurrency in terms of the 3 problems which occur. **(10)**

Ans: There are three ways in which a transaction through correct in itself can produce the wrong answer if interference occurs on the part of some other transaction.

The three problems are :

(i) The lost update problem

Example :

Transaction A	Time	Transaction B
Retrieve P	t1	
	t2	Retrieve P
Update P	t3	
	t4	Update P

Transaction A losses an update at time t4 .

(ii) The uncommitted dependency problem

Transaction A	Time	Transaction B
	t1	
	t2	Update P
Retrieve P	t3	
	t4	Rollback

Transaction A becomes dependent on an uncommitted change at time t3.

(iii)

Transaction A	Time	Transaction B
	t1	
	t2	Update P
Update P	t3	t4 Rollback

Transaction A updates change at time t3 and losses that update at time t4.

Q.120 Compare the two log-based recovery scheme in terms of ease of implementation and overhead cost. **(6)**

Ans: There are two log based recovery techniques : **deferred update** and **immediate update** scheme, which are also called as NO-UNDO/REDO and UNDO/NO-REDO techniques respectively.

In deferred update scheme, actual updates to the database are deferred or postponed until after a transaction completes its execution successfully and reaches its commit point. Before reaching commit, all transactions updates are recorded in the log and in the cache buffer. After the transaction reaches its commit point and the log is force written to the disk, the updates are recorded in the database. If a transaction fails before it reaches the commit point, it would not have modified the database and so no undo is required. But it may be required to redo some of the operations as their effects may not have reached the database. In case of failure, log files are used to perform recovery operations. We examine the log file starting the last entry and go back till the most recent checkpoint. The redo procedure performs all the writes to the database using the after-image log records for the transaction, in the order in which they were written to the log. Thus this method guarantees that we will update any data item that was not properly updated prior to the failure.

In immediate update technique, the database may be updated by some operations of transaction before the transaction reaches its commit point. In case of failure, we will have to redo the updates of committed transactions and undo the effects of uncommitted transactions. Similar to deferred update scheme, here also log files are used to perform the recovery scheme. Write-ahead protocol is used to record the update operations in the log (on disk) before it is written to the database. If a transaction aborts, the log can be used to undo it, since it contains all the old values for the update fields.

Q.121 Briefly describe the different kinds of users of a DBMS. **(6)**

Ans: Different types of DBMS users are:

(i) **Software Engineers:** These are the people responsible for developing application programs using DBMS.

(ii) **End users:** These are the people whose jobs require access to the database for querying, updating, and generating reports.

(iii) **Database Administrator**: They are responsible for authorizing access to the database, coordinating and monitoring its use.

(iv) **Database designers**: They are responsible for identifying data to be stored in the database.

Q.122 Define the concept of aggregation. Give two examples where this concept is useful. **(4)**

Ans: Aggregation transforms a relationship between objects into a higher-level object. A new data type, called aggregate, is developed which, under certain criteria of "well-definedness," specifies aggregation abstractions. Relational databases defined as collections of aggregates are structured as a hierarchy of n-ary relations. To maintain well-definedness, update operations on such databases must preserve two invariants. Well-defined relations are distinct from relations in third normal form. It is shown that these notions are complementary and both are important in database design. A top-down methodology for database design is described which separates decisions concerning aggregate structure from decisions concerning key identification. It is suggested that aggregate types, and other types which support real-world abstractions without introducing Key identification**.**

Q.123 Explain the following. Give an example

(i) Superkey

(ii) Weak entity set

(iii) Attribute inheritance **(6)**

Ans: (i) Superkey: A **superkey** is defined in the relational model of database organization as a set of attributes of a relation variable 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 are functionally dependent.

(ii) **Weak Entity Set:** An Entity set that does not have sufficient attribute to form a primay key is termed as weak entity set. For example, dependents of an employee in an organization do not have an identifying attribute.

(iii) **Attribute inheritance:** During the rendering of the objects in a view, attribute sets of objects higher in the view hierarchy are inherited by objects below them. For example, if the attribute set of a view specifies a particular diffuse color, then all objects in that view are rendered with that diffuse color, *unless* some other attribute set overrides the color specified in the view attributes. That is, if some face of some object has an attribute set containing a different diffuse color, the face's diffuse color overrides the diffuse color that otherwise would have been inherited from the view attribute set.

Q.124 Define the following

(i) A relation.

(ii) Atom of domain relational calculus. **(4)**

Ans: (i) A relation: A database relation is a predefined row/colum format for storing information in a relational database. Relations are equivalent to tables.
(ii) Atom of domain relational calculus: An atom has the form
$< x_1, x_2, \ldots, x_n > \rightarrow r$, where *r* is a relation on *n* attributes and $x_1, x_2, \ldots, x_n$ are the domain variables or domain constraints.
$x\Theta y$, where *x* and *y* are domain variables and Θ is the comparison operator ($<$,$>$, $\leq$, $\geq$, $=$, $\neq$). It is required that *x* and *y* have domains that can be compared by Θ.
$x\Theta c$, where *x* is a domain variable, Θ is a comparison operator, and *c* is a constraint in the domain of attributes for which x is a domain variable.

Q.125 Given the relations R(A, B, C) and S(C,D,E,F) give an expression in tuple relational calculus that is equivalent to each of the following

(i) $\Pi_{A,B,C}(R)$

(ii) $\sigma_{E=10}(S)$

(iii) $R \bowtie S$

(iv) $R \cup S$ **(12)**

Ans: (i) $\{ t \mid t \in R \}$
(ii) $\{ t \mid t \in S \wedge t[E] = 10 \}$
(iii) $\{ t,P \mid t \in R \wedge P \in S \wedge t[C] = P[C] \}$
(iv) R U S is not defined as these two relations are not union- compatible

Q.126 Given the relations Staff (staff No, position, salary) and Property (number, rent, staff No) given below. The staff looks after a given property.

Staff

Staff No	position	salary
SL21	Manager	50000.00
SL37	Assistant	15000.00
SG14	Supervisor	25000.00
SG5	Manager	45000.00

Property

Number	Rent	Staff No
PA14	5000.00	SL21
PG4	6000.00	SG5
PL94	10000.00	SL21

Give the result table for the following SQL queries

(i)
```
SELECT  position, COUNT(staff No) AS POS, my count
        FROM    Staff
```

(ii)
```
SELECT  staff No
        FROM    Staff
        WHERE   salary > (SELECT AVG(salary) FROM Staff)
```

(iii)
```
SELECT      staff No
FROM        Property
GROUP By    staff No
HAVING      COUNT(*) > 1
```
(iv)
```
INSERT INTO  Staff
VALUES       ('SG33', 'Assistant')
```
(16)

Ans:(i) This query will return an error message as it is not possible to use an aggregation function and an attribute without using the group by clause.

(ii)

Staff No
SL21
SG5

(iii)

Staff No
SL21

(iv)

Staff No	**Position**	**Salary**
SL21	Manager	50000.00
SL37	Assistant	15000.00
SL14	Supervisor	25000.00
SG5	Manager	45000.00
SG33	Assistant	

Q.127 Derive the union rule, decomposition rule and the pseudoTransitivity rule using the three Armstrong's axioms. **(6)**

Ans: **Union** $\{ X \rightarrow Y, X \rightarrow Z\} \models X \rightarrow YZ$
Proof: $X \rightarrow Y$
$X \rightarrow Z$
$X \rightarrow YZ$ (Augmentation)
$XY \rightarrow YZ$
$X \rightarrow YZ$
Decomposition $\{ X \rightarrow YZ\} \models X \rightarrow Y$
Proof: $X \rightarrow YZ$
$YZ \rightarrow Y$
$X \rightarrow Y$
Pseudo transitivity $\{ \mathbf{X} \rightarrow Y, WY \rightarrow Z\} \models WX \rightarrow Z$
Proof: $X \rightarrow Y$
$WY \rightarrow Z$
$WX \rightarrow WY$
$WX \rightarrow Z$

Q.128 Define multivalued dependency. What do understand by trivial multivalued dependency? **(4)**

Ans: A **multivalued dependency** *is a full constraint* between two sets of attributes in a relation. In contrast to the functional dependency, the **multivalued dependency** requires that certain tuples be present in a relation. Therefore, a multivalued dependency is a special case of *tuple-generating dependency*. The multivalued dependency plays a role in the 4NF database normalization.
An MVD $X \rightarrow\rightarrow Y$ in R is called a trivial MVD if (a) Y is a subset of X or

(b) X U Y = R

Q.129 Given R (A, B, C, D, E) and M the set of multivalued dependencies
(i) $A \rightarrow\rightarrow BC$
(ii) $B \rightarrow\rightarrow CD$
(iii) $E \rightarrow\rightarrow AD$
Is R in 4NF? Justify your answer. If it is not, decompose R into 4NF. **(6)**

Ans: A table is in 4NF if and only if, for every one of its non-trivial multivalued dependencies $X \rightarrow\rightarrow Y$, X is a superkey—that is, X is either a candidate key or a superset thereof.

Q.130 Describe the four main ways of optimising disk block access. **(8)**

Ans: 1. Disk
2. Non-volatile write buffers
3. File organization (***Clustering***)
4. Log-based file system

Q.131 Describe the algorithm for updating indices for a single level index when a record is
(i) Inserted **(ii)** deleted
What will be the modification if there are multilevel indices? **(8)**

Ans: **Inserted:** The time it takes to insert a new data item. This value includes the time it takes to find the correct place to insert the new data item, as well as the time it takes to update the index structure.
Deleted: The time it takes to delete a data item. This value includes the time it takes to find the item to be deleted, as well as the time it takes to update the index structure

Q.132 How do you estimate the query cost for natural join when
(i) $R \cap S = \phi$
(ii) $R \cap S$ is a foreign key **(6)**

Ans: (i) If $R \cap S = \Phi$, then $r \bowtie s$ is the same as $r \times s$.
(ii) If $R \cap S$ in S is a foreign key in S referencing R, Then the number of tuples in $r \bowtie s$ is exactly the same as the number of tuples in S.
The case for $R \cap S$ being a foreign key referencing S is symmetric

Q.133 Given two relations R (A,B) and S (B,C) with number of tuples in R and S equal to 500 and 1000 respectively and B is the foreign key in R, what is the number of tuples in $R \bowtie S$. **(4)**

Ans: The number of tuples in $R \bowtie S$ is 500000.

Q.134 Explain Thomas' Write rule. Show how it is different from timestamp ordering proto**(6)**

Ans: The Thomas Write rule is a rule in timestamp-based concurrency control.

Given a Timestamp on a transaction T, TS(T) and Write Timestamp on an object O, WTS(O):
It states if TS(T) < WTS(O), the current write action has been made obsolete by the most recent write of O, which follows the current write according to timestamp ordering.
Given a non-conflict serializable transaction schedule:
Text: T1:R(A), T2:W(A), T2 Commit, T1: W(A), T1 Commit.
The Thomas Write Rule relies on the fact that T1's write on object A is never seen by any transaction and postulates that the schedule above is equivalent to the schedule below where T2 occurs strictly after T1, and that hence the write of T1 can be ignored:
Text: T1:R(A), T1: W(A), T1 Commit, T2:W(A), T2 Commit.
This schedule has the same effect as the first and is conflict serializable

Q.135 Explain
(i) recoverable schedule. **(ii)** cascadeless schedule **(8)**

Ans: Recoverable Schedule: A recoverable schedule is one where for each pair of transactions Ti and Tj such that Tj reads a data item previously written by Ti, the commit operation of Ti appears before commit operations of Tj.
Cascadeless Schedule A cascadeless schedule is one where for each pair of transactions Ti and Tj such that Tj reads a data item previously written by Ti, the commit operation of Ti appears before the read operation of Tj.

Q.136 Define two-phase locking protocol. **(2)**

Ans: According to the *two phase locking* protocol, locks are handled by a transaction in two distinct, consecutive phases during the transaction's execution:
Phase 1: locks are acquired and no locks are released.
Phase 2: 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.

Q.137 Consider the following two transactions
T1 : read (A);
read (B);
B = A + B;
write (B)
T2 : write (A)
read (B)
Add lock and unlock instructions so that the transaction T1 and T2 observe two-phase locking protocol. Is it deadlock free? **(6)**

Ans:T1: lock- S(A)
Read (A)
Lock –X (B)
Read (B)
B = A +B
Write (B)
Unlock (A)

Unlock(B)
T2:lock-X(A)
Lock**-** S(B)
write (A)
Read (B)
Unlock(A)
Unlock(B)

Execution of these transactions can result in deadlock. For example, consider the following partial schedule:

T1	T2
Lock- S(A)	
Read(A)	
	Lock- X(A)
Lock – X(B)	
	Lock – S(B)

Q.138 Explain the recovery process of a checkpoint mechanism. How does the frequency of checkpoints affect
(i) system performance when no failure occurs.
(ii) the time it takes to recover from a system crash **(8)**

Ans: A checkpoint log record indicates that a log record and its modified data have been written to stable storage and that the transaction needs not to be redone in case of a system crash. Obviously, the more often checkpoint are performed, the less likely it is that redundant updates will have to be performed during the recovery process. System performance when no failure occurs—If no failures occur, the system must incur the cost of performing checkpoints that are essentially unnecessary. In this situation, performing checkpoints essential often will lead to better system performance.
(i) The time it takes to recover from a system crash—The existence of a checkpoint record means that an operation will not have to be redone during system recovery. In this situation, the more often checkpoints were performed, the faster the recovery time is from a system crash
(ii) The time it takes to recover from a disk crash—The existence of a checkpoint record means that an operation will not have to be redone during system recovery. In this situation, the more often checkpoints were performed, the faster the recovery time is from a disk crash.

Q.139 Write short notes on
(i) hash file organization.
(ii) physical and logical independence. **(8)**

Ans:i) Hash file organization
1. **Hashing** involves computing the address of a data item by computing a function on the search key value.
2. A **hash function h** is a function from the set of all search key values *K* to the set of all bucket addresses *B*.

We choose a number of buckets to correspond to the number of search key values we will have stored in the database.

To perform a lookup on a search key value K_i, we compute hk_i, and search the bucket with that address.
If two search keys *i* and *j* map to the same address, because $h(K_i)=h(K_j)$, then the bucket at the address obtained will contain records with both search key values.
In this case we will have to check the search key value of every record in the bucket to get the ones we want.
Insertion and deletion are simple

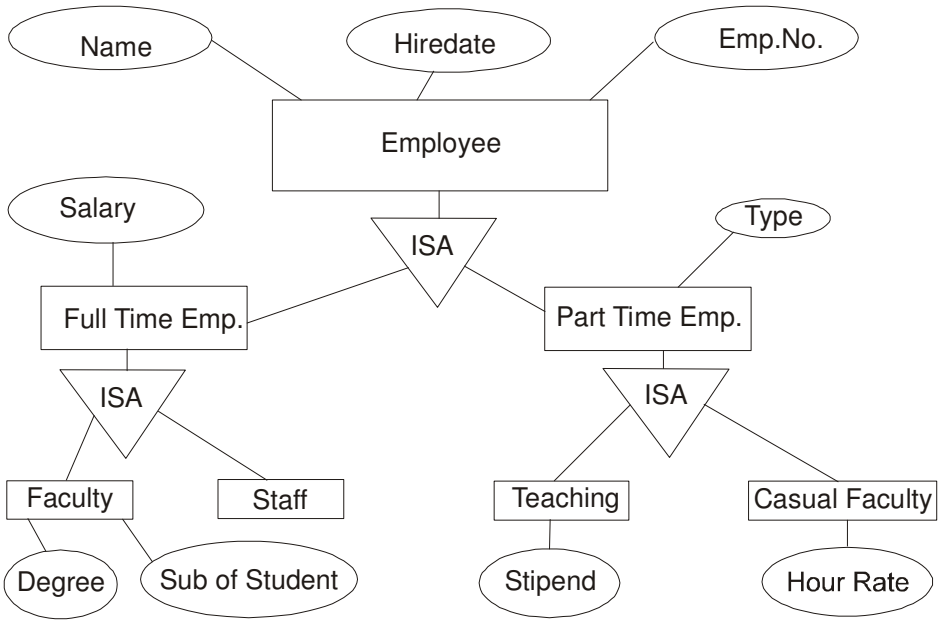
(ii) Physical and Logical independence

Physical data independence: The capability to modify physical level without causing application program to be rewritten.
Logical independence: The capability to modify logical level without causing application program to be rewritten.

Q.140 The entity set EMPLOYEE is a generalization of the entity sets FULL_TIME_EMPLOYEE and PART_TIME_EMPLOYEE. The former is a generalization of the entity sets FACULTY with attributes degree and subject of interest and STAFF with attribute classification; the latter, is a generalization of the entity sets TEACHING with attribute stipend and CASUAL_FACULTY with attribute hour rate. STAFF inherits the attribute Salary of the entity set FULL_TIME_EMPLOYEE and the latter, in turn, inherits the attributes of EMPLOYEE. FULL_TIME_EMPLOYEE is a specialization of the entity set EMPLOYEE and is differentiated by the additional attribute Salary. Similarly, PART_TIME_EMPLOYEE is specialization differentiated by the presence of the attribute Type. Each employee must have attributes empno, name and hire_date.

(i) Draw an E-R diagram for the system.
(ii) Convert this E-R diagram to relational tables. **(12)**

Ans. (i)



(ii) Emp(name, hiredate,empno)
Full Time Emp (name, hiredate, empno, salary)
Part Time Emp (name,hiredate,empno,type)
Faculty (empno,salary,degree,subofinterest)
Staff (empno,sname,sno,salary)
Teaching (empno, type, stipend)
Casual Faculty (empno,type,hourrate)

Q.141 Explain the difference between a one-to-many and a many-to-many relationship. Which logical data structures have one-to-many and which have many-to-many relationship?**(4)**

Ans: One-to-Many: means that at most one entity in set A is assigned to any number of entities in set B. For example Employee in set A has any number of Phone numbers in set B.
Many-to-Many: means that any numbers of entities in set A assigned to any number of entities in set B. For example several employees in set A may have several accounts in set B.

Q.142 What are the DBMS languages? Explain. **(6)**

Ans:1) **Data Definition Languages(DDL):** It is used to specify database schema e.g, CREATE DROP statements etc.
2) Data Manipulation language(DML): DML is used to express database queries and update. Eg, SELECT, UPDATE, DELETE statements, etc.

Q.143 What are Armstrong's inference rules?
Suppose we are given relation R with attributes A, B, C, D, E, F, and the FDs,
A→BC
B →E
CD →EF
Prove that FD AD→F also holds in R. **(10)**

Ans: Armstrong's axioms are a set of axioms (or, more precisely, inference rules) used to infer all the functional dependencies on a relational database. The axioms are sound in that they generate only functional dependencies in the closure of a set of functional dependencies (denoted as F^+) when applied to that set (denoted as F). They are also complete in that repeated application of these rules will generate all functional dependencies in the closure F^+.
Given: A → BC CD → EF
To Prove: AD → F
After applying decomposition rule
A→ B ----------1
A→C ----------2
B→E ----------3
CD→E----------4
CD→F----------5
Applying pseudotransitivity rule on 2 and 5
AD→F
Hence Proved

Q.144 Consider the relations:

PROJECT(proj#, proj_name)
EMPLOYEE(emp#, emp_name)
ASSIGNED(proj#, emp#)
Use relational algebra to express the following queries:

(i) Find the employee number of employees who work on at least all of the projects that employee 107 works on.

(ii) Get the employee number of employees who work on all projects. **(8)**

Ans: (i) ASSIGNED_TO ÷ $\Pi_{\text{Project \#}}$ ($\sigma_{\text{emp\# = 107}}$(ASSIGNED_TO))) -107

(ii) ASSIGNED_TO ÷ $\Pi_{\text{Project \#}}$ (PROJECT)

Q.145 Use tuple and domain calculus to express the following query Compile a list of employee number of employees who work on all projects **(8)**

Ans: Tuple calculus query:

t[Emp #] / t ∈ ASSIGNED_TO Λ∀
P(P ∈ PROJECT → ∃ u(u ∈ ASSIGNED_TOΛP[Project#]=u[Project #] Λ t[Emp#] =u[Emp #]))}

Domain Calculus query

{ e| ∃ p(<p,e> ∈ ASSIGNED_TO Λ ∀ P1(<P1,n1,c1> ∈ PROJECT → <p1,e> ∈ ASSIGNED_TO))}

Q.146 What is the difference between serial and sequential files? How searching is applied on both? **(8)**

Ans: A **serial file** is one in which the records have been stored in the order in which they have occurred. They have not been sorted into any particular order.
An **example** of a serial file is an **unsorted transaction file**.
A **shopping list** is an example of a non-computerised serial file. Items are appended to the list when that item runs low.
Serial files can be stored on tape, disc or in memory.
A **sequential file** is one in which the records are stored in **sorted** order of one or more key fields.
An **example** of a sequential file is a **sorted transaction file**.
A **class register** is an example of a non-computerised sequential file sorted on surname

Q.147 Discuss the problem of Spurious tuples and how we may prevent it. **(8)**

Ans: A spurious tuple is, basically, a record in a database that gets created when two tables are joined badly. In database-ese, spurious tuples are created when two tables are joined on attributes that are neither primary keys nor foreign keys. To avoid spurious tuples, avoid joining relations that contain matching attributes that are not (foreign key, primary key) combinations because joining on such attributes may produce spurious tuples.

Q.148 Explain the followings:

(i) Third normal form

(ii) Query Processing
(iii) Relational Completeness
(iv) Radix conversion method **(16)**

Ans: (i) Third Normal Form: The **third normal form** (**3NF**) is a normal form used in database normalization. 3NF was originally defined by E.F. Codd in 1971. Codd's definition states that a table is in 3NF if and only if both of the following conditions hold:The relation R (table) is in second normal form (**2NF**)
Every non-prime attribute of R is non-transitively dependent (i.e. directly dependent) on every key of R.

(ii) Query Processing

Data Query Processing allows a user to report or analyze structured and unstructured data pulled from multiple data sources.

(iii) Relational Completeness

Codd defined the term ***relational completeness*** to refer to a language that is complete with respect to first-order predicate calculus apart from the restrictions he proposed. In practice the restrictions have no adverse effect on the applicability of his relational algebra for database purposes.

(iv) Radix conversion method

One clever way to convert binary numbers to BCD notation (binary-coded decimal) is the "double dabble algorithm". It can be adapted to convert binary numbers directly to ASCII digits, and to convert binary numbers into other bases. The double dabble algorithm also works for mixed bases -- for example, for converting a binary number of seconds into the decimal digits for days, 10s of hours, hours, 10s of minutes, minutes, 10s of seconds and seconds

Q.149 Consider the relations
EMP(ENO,ENAME,AGE,BASIC)
WORK_ON(ENO,DNO)
DEPT(DNO,DNAME,CITY)
Express the following queries in SQL
(i) Find names of employees whose basic pay is greater than average basic pay.
(ii) Find the sum of the basic pay of all the employees, the maximum basic pay, the minimum basic pay and the average basic pay. **(8)**

Ans: (i) Select ENAME from EMP where BASIC > (select avg(BASIC) from EMP);
(ii) Select sum(BASIC), max(BASIC), min(BASIC), avg(BASIC) from EMP;

Q 150 What are the General Transformation Rules for Relational operations? **(8)**

Ans: Transformation rules transform one relational algebra expression to AN EQUIVALENT ONE

- Used by the query optimizer to optimize query tree
- Any rule, if applied, makes sure that the resulting tree is equivalent ➔ resulting execution plan is equivalent

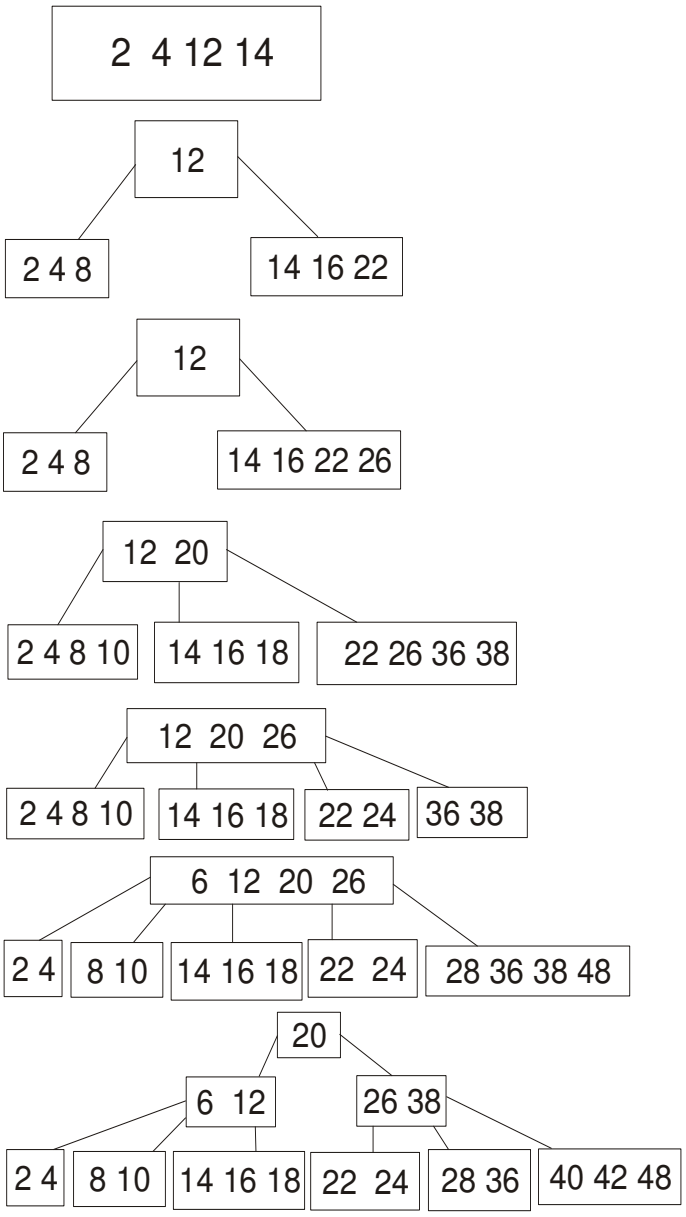
General Transformation Rules:

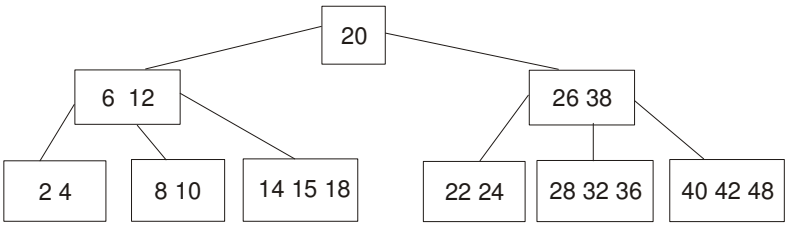
- *Commutativity of s*: The s operation is commutative:
  - $s_{c1}(s_{c2}(R)) = s_{c2}(s_{c1}(R))$
  - More selective selections first

Q.151 Draw a B-tree of order 5 by inserting the following data
(2 14 12 4 22 8 16 26 20 10 38 18 36 24 6 48 28 40 42 32)
After constructing the B-tree delete the following data:
(i) delete 36
(ii) delete 32

Ans.

(6+4=10)

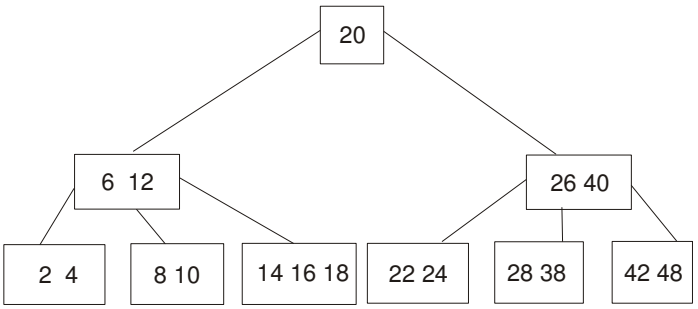




Final tree

While delete 36, remove it (no change in structure)

While delete 32, the B-tree looks as



Q.152 What is a view? Create a view of EMP table named DEPT 20, to show the employees in department 20 and their annual salary. **(6)**

Ans: View is a virtual table which does not contains it own data it fetch the data from the original table.

Create view DEPT20 as select * from dept;
Where dno=20 group by dno;

Q.153 Discuss the timestamp ordering protocol for concurrency control. How does strict timestamp ordering differs from basic timestamp ordering? **(8)**

Ans: The timestamp ordering protocol ensures that any conflicting read and write operations are executed in timestamp order. This protocol operates as follows:

1.Suppose that transaction T_i issues read (Q).

a. If *TS* (T_i) < W-timestamp (Q), then T_i needs to read a value of Q that was already overwritten. Hence, the read operation is rejected, and T_i is rolled back.

b. If *TS* (T_i) W-timestamp (Q), then the read operation is executed, and R timestamp (Q) is set to the maximum of R-timestamp (Q) and TS (T_i).

2. Suppose that transaction T_i issues write (Q).

a. If *TS* (T_i) < R-timestamp (Q), then the value of Q that T_i is producing was needed previously, and the system assumed that that value would never be produced. Hence, the system rejects the write operation and rolls T_i back.

b. If *TS* (T_i) < W-timestamp (Q), then T_i is attempting to write an obsolete value of Q. Hence, the system rejects this write operation and rolls T_1 back.

c. Otherwise, the system executes the write operation and sets W-time stamp (Q) to *TS* (T_i).

TS (T_i) = *R/W* - TS (Q).
If a transaction T_i is rolled back by the concurrency-control scheme as result of issuance of either a read or writes operation, the system assigns it a new timestamp and restarts it.

Q.154 Explain the shadow paging recovery technique. **(8)**

Ans: Shadow paging is a technique used to achieve atomic and durable transactions, and provides the ability to manipulate pages in a database. During a transaction, the pages affected by the transaction are copied from the database file into a workspace, such as volatile memory, and modified in that workspace. When a transaction is committed, all of the pages that were modified by the transaction are written from the workspace to unused pages in the database file. During execution of the transaction, the state of the database exposed to the user is that in which the database existed prior to the transaction, since the database file still contains the original versions of the modified pages, as they existed before being copied into the workspace. If a user accesses the database before the transaction is complete, or upon recovery of a failure, it will appear as though the transaction has not occurred

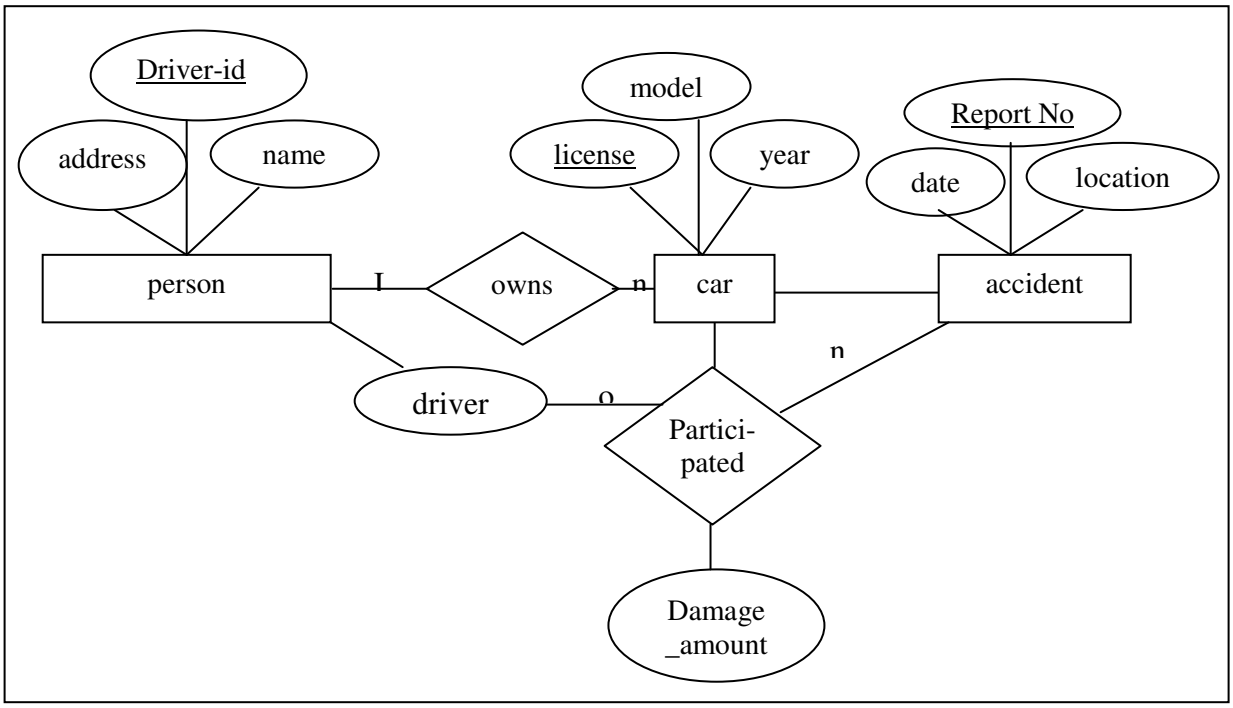
Q.155 Explain the steps for reduction of E-R model into relational model. **(8)**

Ans:(i) Entity set in E-R model will be treated as table name in relational Model.

(ii) Attributes of entity set in E-R diagram will be considered as fields or column name of table.
(iii) Underline attributes of Attribute in E-R model will be treated as Primary key of the table.

Q.156 Construct an E-R diagram for a car insurance company that has a set of customers,each of whom owns one or more cars. Each car has associated with it zero to any number of recorded accidents **(8)**

Ans:



Q.157 Consider the following relations **(8)**

SALESPERSON(SSN, Name, Start_Year, Dept_No)
TRIP(SSN,From_City,To_City,Departure_Date,Return_Date,Trip_ID)
EXPENSE(Trip_ID, Account#, Amount)
Write queries in relational algebra

(i) Give the details (all attributes to TRIP relation) for the trips that exceeded $2000 in expenses.
(ii) Print the SSN of salesman who took trips to 'Chandigarh'.
(iii) Print the total trip expenses incurred by the salesman with SSN ='234-56-7890'.

Ans:(i) $\Pi_{SSN, From\_City, To\_City, Departure\_Date, Return\_Date, Trip\_D}(\sigma_{AMT>2000}(TRIP \bowtie EXPENSE))$

(ii) $\Pi_{SSN}(\sigma_{To\_City = \text{"Chandigarh"}}(SALESPERSON \bowtie TRIP \bowtie EXPENSE))$

(iii) $\Pi_{AMOUNT}(\sigma_{SSN = '234\text{-}56\text{-}7890'}(TRIP) \bowtie EXPENSE)$

Q.158 How Relational Calculus is different from Relational Algebra? What do understand by TRC queries and DRC queries? **(8)**

Ans: Relational calculus consist of two calculi, the tuple relational calculus and the domain relational calculus, that are part of the relational model for databases and provide a declarative way to specify database queries. This in contrast to the relational algebra which is also part of the relational model but provides a more procedural way for specifying queries.
The relational algebra might suggest these steps to retrieve the phone numbers and names of book stores that supply *Some Sample Book*:

1. Join books and titles over the BookstoreID.
2. Restrict the result of that join to tuples for the book *Some Sample Book*.
3. Project the result of that restriction over StoreName and StorePhone.

The relational calculus would formulate a descriptive, declarative way:
Get StoreName and StorePhone for supplies such that there exists a title BK with the same BookstoreID value and with a BookTitle value of *Some Sample Book*.
The relational algebra and the relational calculus are essentially logically equivalent: for any algebraic expression, there is an equivalent expression in the calculus, and vice versa. This result is known as Codd's theorem.
TRC stands for tuple relational calculus and DRC stands for domain relational calculus. The TRC is based on specifying a query in terms of tuple variable and DRC is based on domain variables.

Q.159 What is DDL? Make a list of commands with short description used in DDL **(8)**

Ans: Data Definition Language (**DDL**) is a computer language for defining data structures. The term was first introduced in relation to the Codasyl database model, where the schema of the database was written in a Data Definition Language describing the records, fields, and "sets" making up the user Data Model.
List of DDL commands:

(i) To define a table/ relation
CREATE TABLE ------
(ii) To remove table/relation definition

DROP TABLE ----

(iii) To change the definition of an existing table
ALTER TABLE-------

Q.160 Consider the insurance database, where the primary keys are underlined.
person (ss#, name, address)
car (license, year, model)
accident (date, driver, damage-amount)
owns (ss#, license)
log (license, date, driver)
Construct the following SQL queries for this relational database.

(i) Find the total number of people whose cars were involved in accidents in 1989.

(ii) Find the number of accidents in which the cars belonging to "John Smith" were involved.

(iii) Add a new customer to the database.

(iv) Add a new accident record for the Toyota belonging to "Jones" **(8)**

Ans:(i) Select count(driver) from accident where date is between '01-Jan-89'and '31-DEC-89';

(ii) Select count(*) from accident,person,owns,log where name='John Smith' and person.ss# = owns.ss# and owns.License= log.License and log.driver = accident.driver

(iii) insert into person values(1,'Raj', 'A10 RajNagar Gzb');

(iv) insert into car values('L20', 1982, 'Toyata');

insert into person values('S1', 'Jones', 'A2 Kavi Nagar Gzb');
insert into owns values ('S1', 'L20');
insert into accident values (Sysdate,"Jones",10000)

Q.161 Explain Boyce-Codd Normal Form with example and also Compare BCNF and 3NF.**(8)**

Ans:
BCNF : For every functional dependency X->Y in a set *F* of functional dependencies over relation *R*, either:

- Y is a subset of X or,
- X is a *superkey* of R

whereas
3NF: For every functional dependency X->Y in a set *F* of functional dependencies over relation *R*, either:

- Y is a subset of X or,
- X is a *superkey* of R, or
- Y is a subset of K for some key K of R
  - no subset of a key is a key

Q.162 What are the reasons of bucket overflow? Explain any two methods for solving this problem. **(8)**

Ans: It is common for file structures to be divided into equal-length partitions, called buckets, into which records arrive for insertion and from which records are physically deleted. We give a simple algorithm which permits calculation of the average time until

overflow for a bucket of capacity n records, assuming that record insertions and deletions can be modelled as a stochastic process in the usual manner of queuing theory. We present some numerical examples, from which we make some general observations about the relationships among insertion and deletion rates, bucket capacity, initial fill, and average time until overflow. In particular, we observe that it makes sense to define the stable point as the product of the arrival rate and the average residence time of the records; then a bucket tends to fill up to its stable point quickly, in an amount of time almost independent of the stable point, but the average time until overflow increases rapidly with the difference between the bucket capacity and the stable point.

Q.163 What is irreducible set of dependencies? Relation R with attributes A, B, C, D, and FDs,
$A \rightarrow BC$
$B \rightarrow C$
$A \rightarrow B$
$AB \rightarrow C$
$AC \rightarrow D$
compute an irreducible set of FDs that is equivalent to this given set. **(8)**

Ans: Irreducible function depending set
A functional depending set S is irreducible if the set has three following properties:

1. Each right set of a functional dependency of S contains only one attribute.
2. Each left set of a functional dependency of S is irreducible. It means that reducing any one attribute from left set will change the content of S (S will lose some nformation).
3. Reducing any functional dependency will change the content of S.
   Sets of functional dependencies with these properties are also called *canonical* or *minimal*.

Step 1: After one attribute on LHS, FDs are:
A→ B---------- (1)
A→ C---------- (2)
B→ C---------- (3)
A → B---------- (4)
AB→C---------- (5)
AC→D---------- (6)
Since (1) and (4) are same, therefore removing (4)
Step 2: Checking if there is any FD implied by other FD for which closure of LHS of FD say **a**, is found by excluding **a** in set of FD. If closure has RHS of **a** then FD is redundant. Therefore

(1)	$\Rightarrow A^+ = ACD$	Since RHS of (1) is not in A^+ therefore (1)is redundant
(2)	$\Rightarrow A^+ = ABC$	Since RHS of (1) is in A^+ therefore (2) is redundant
(3)	$\Rightarrow B^+ = B$	implies non-redundant
(4)	$\Rightarrow AB^+ = ABC$	implies redundant
(5)	$\Rightarrow AC^+ = AC$	implies non-redundant

Therefore set of FDs left:
A→ B---------- (1)
B→ C---------- (3)
AC→ D---------- (6)

Step 3: These can be further reduced as
A→ B---------- (1)
B→ C---------- (3)
A→ D---------- (7)
Since adding A to both sides of (1) and (3) we get
AA→AB
AB→ AC
Implies A→ AC---------- (8)
(6) and (8) implies A→D which is (7).

Q.164 What is indexed sequential file organization? What are the applications of this organization? **(8)**

Ans: An index file can be used to effectively overcome the problem of storing and to speed up the key search as well. The simplest indexing structure is the single-level one: a file whose records are pairs key-pointer, where the pointer is the position in the data file of the record with the given key. Only a subset of data records, evenly spaced along the data file, are indexed, so to mark intervals of data records.
A key search then proceeds as follows: the search key is compared with the index ones to find the highest index key preceding the search one, and a linear search is performed from the record the index key points onward, until the search key is matched or until the record pointed by the next index entry is reached. In spite of the double file access (index + data) needed by this kind of search, the decrease in access time with respect to a sequential file is significant

Q.165 What are the General Transformation Rules for Relational Algebra Operations? **(5)**

Ans: General transformation rule for Relational Algebra are:

1. Cascade of s: A conjunctive selection condition can be broken up into a cascade (sequence) of individuals operations
   $s_{c1\ AND\ c2\ AND\ ...\ AND\ cn}(R) = s_{c1}(s_{c2}(...(s_{cn}(R))...))$
2. Commutativity of s: The s operation is commutative:
   $s_{c1}(s_{c2}(R)) = s_{c2}(s_{c1}(R))$
3. Cascade of p: In a cascade (sequence) of p operations all but the last one can be ignored:
   $p_{List1}(p_{List2}(...(p_{Listn}(R))...)) = p_{List1}(R)$
4. Commuting s with p: If the selection condition c involves only the attributes A1,….., An in the projection list, the two operations can be commuted:
   $p_{A1, A2, ..., An}(s_c(R)) = s_c(p_{A1, A2, ..., An}(R))$

Q.166 How does a query tree represent a relational algebra expression? **(5)**

Ans: This involves transforming an initial expression (tree) into an equivalent expression (tree) which is more efficient to execute. Two relational algebra expressions are said to be equivalent if the two expressions generate two relation of the same set of attributes and contain the same set of tuples although their attributes may be ordered differently.
The query tree is a data structure that represents the relational algebra expression in the query optimization process. The leaf nodes in the query tree corresponds to the input relations of the query. The internal nodes represent the operators in the query. When

executing the query, the system will execute an internal node operation whenever its operands available, then the internal node is replaced by the relation which is obtained from the preceding execution.

Q.167 Differentiate between static hashing and dynamic hashing. **(6)**

Ans: Static Hashing has the number of primary pages in the directory fixed. Thus, when a bucket is full, we need an overflow bucket to store any additional records that hash to the full bucket. This can be done with a link to an overflow page, or a linked list of overflow pages. The linked list can be separate for each bucket, or the same for all buckets that overflow. When searching for a record, the original bucket is accessed first, then the overflow buckets. Provided there are many keys that hash to the same bucket, locating a record may require accessing multiple pages on disk, which greatly degrades performance.
The problem of lengthy searching of overflow buckets is solved by Dynamic Hashing. In Dynamic Hashing the size of the directory grows with the number of collisions to accommodate new records and avoid long overflow page chains. Extendible and Linear Hashing are two dynamic hashing techniques.

Q.168 What is the two-phase locking protocol? How does it guarantee serializability? **(6)**

Ans: According to the *two phase locking* protocol, locks are handled by a transaction in two distinct, consecutive phases during the transaction's execution:
Phase 1: locks are acquired and no locks are released.
Phase 2: 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

Q.169 Explain the lost update problem and dirty read problem. **(6)**

Ans: Lost update problem: This problem occurs when two transactions that access the same database items have their operations interleaved in a way that makes the value of some database items incorrect. Eg: consider the following interleaved schedule:

T1	T2
read(X)	
X=X-N	
	read(X)
	X=X+N
write(X)	
	write(X)

In this case, X has incorrect value because its update by T1 is lost.
Dirty Read Problem: This problem occurs when one transaction updates a database item and then the transaction fails for some reason. The updated item is accessed by another transaction before it is changed lock to its original value. Consider the following interleaved schedule

T1	T2
	Sum=0
	read (A)

```
                    sum = sum + A
read(X)
X=X-N
Write(X)
                    read(X)
                    sum=sum+X
```

Suppose T1 fails this, then T2 has incorrect sum because change of T1 is not valid and should not have included when sum is computed.

Q.170 What are deadlocks? How can they be avoided? **(4)**

Ans: Deadlock refers to a specific condition when two or more processes are each waiting for another to release a resource, or more than two processes are waiting for resources in a circular chain. Deadlock is a common problem in multiprocessing where many processes share a specific type of mutually exclusive resource known as a *software,* or *soft,* lock. Computers intended for the *time-sharing* and/or *real-time* markets are often equipped with a *hardware lock* (or *hard lock*) which guarantees *exclusive access* to processes, forcing serialization. Deadlocks are particularly troubling because there is no *general* solution to avoid (soft) deadlocks.
One of the methods of deadlock avoidance related to timestamps there are two variations: wait-die and wound-wait.

Q.171 Explain the following:

(i) Normalization.
(ii) Joins in relational algebra.
(iii) Role of a database administrator.
(iv) B^+tree index files. **(4x4=16)**

Ans:(i) Normalization:
Database normalization, sometimes referred to as *canonical synthesis*, is a technique for designing relational database tables to minimize duplication of information and, in so doing, to safeguard the database against certain types of logical or structural problems, namely data anomalies. For example, when multiple instances of a given piece of information occur in a table, the possibility exists that these instances will not be kept consistent when the data within the table is updated, leading to a loss of data integrity. A table that is sufficiently normalized is less vulnerable to problems of this kind, because its structure reflects the basic assumptions for when multiple instances of the same information should be represented by a single instance only.

(ii) Joins in relational algebra:

a) Natural join

Natural join is a binary operator that is written as (R* S) where R and S are relations. The result of the natural join is the set of all combinations of tuples in R and S that are equal on their common attribute names. In this only one column along attributes having same name is retained

b) θ-join and equijoin

Consider tables *Car* and *Boat* which list models of cars and boats and their respective prices. Suppose a customer wants to buy a car and a boat, but she doesn't want to spend more money for the boat than for the car. The θ-join on the relation $CarPrice \geq BoatPrice$

produces a table with all the possible options. If the condition uses equality operator then it is also known as equijoin.

c) Outer join

They can be used when we want to keep all the tuples in R, or all those in S or all those in both relations in the result of the JOIN regardless of whether or not they have matching tuples in the other relation

(iii) Role of a database administrator

- **Defining the Schema**

The DBA defines the schema which contains the structure of the data in the application. The DBA determines what data needs to be present in the system ad how this data has to be represented and organized.

- **Liaising with Users**

The DBA needs to interact continuously with the users to understand the data in the system and its use.

- **Defining Security & Integrity Checks**

The DBA finds about the access restrictions to be defined and defines security checks accordingly. Data Integrity checks are also defined by the DBA.

- **Defining Backup / Recovery Procedures**

The DBA also defines procedures for backup and recovery. Defining backup procedures includes specifying what data is to backed up, the periodicity of taking backups and also the medium and storage place for the backup data.

- **Monitoring Performance**

The DBA has to continuously monitor the performance of the queries and take measures to optimize all the queries in the application.

(iv) B^+ tree index files

Indexing mechanisms used to speed up access to desired data.

E.g., author catalog in library

Search Key - attribute to set of attributes used to look up records in a file.

An index **file** consists of records (called index **entries**) of the form

Index files are typically much smaller than the original file

Ordered indices: search keys are stored in sorted order.

- All paths from root to leaf are of the same length
- Each node that is not a root or a leaf has between $[n/2]$ and n children.
- A leaf node has between $[(n–1)/2]$ and $n–1$ values

Special cases:

If the root is not a leaf, it has at least 2 children.

If the root is a leaf (that is, there are no other nodes in the tree), it can have between 0 and $(n–1)$ values

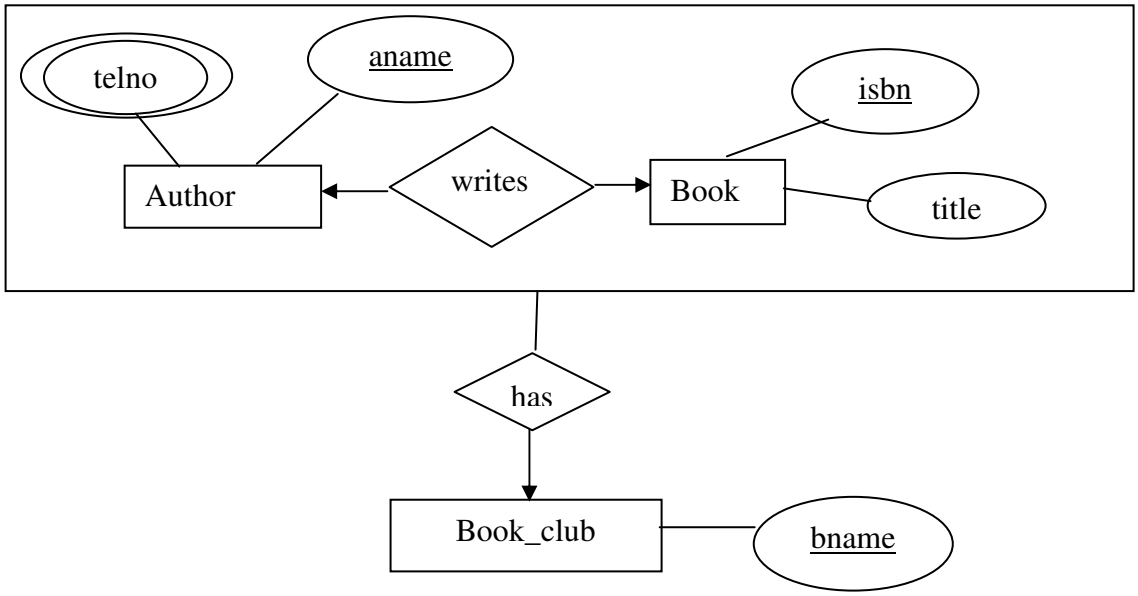
Q.172 Define cardinality and participation constraints on a relationship type, completeness constraint on generalization. **6**

Ans: Cardinality expresses the number of entities to which another entity can be associated via a relationship set. It may be 1:1, 1:N or M:N.

Participation of a entity set in a relationship set can be total or partial. It is total if every entity participates in the relationship. If only some entities participate in a relationship then it is partial.

Completeness constraint can be partial or total. In the case of total each higher level entity must belong to a lower level entity and in the case of partial some higher level entities may not belong to lower level entities.

Q.173. Consider the following ER diagram



Map the diagram to tables. Specify the table names and their attributes. **10**

Ans. The tables are

Author, with attributes aname

Book, with attributes isbn, title

Writes, with attributes isbn, aname

Has, with attributes isbn, aname, bname

Book_club, with attributes bname

Telno, with attributes phoneno, aname

Q.174 Describe the structure of a well formed formulae in relational calculus. **6**

Ans:

- An atom is a formula
- If P1 is a formula, then so are ¬P1 and (P1)
- If P1 and P2 are formulae, then so ar P1 V P2, P1 ^ P2, P1 → P2
- If P1(s) is a formula containing a free variable s and r is a relation then

$\exists s \varepsilon r$ (P1(s)) and $\forall s \varepsilon r$ (P1(s)) are also formulae

Q.175. Consider the relations given below with keys underlined
Branch(branchno, street, city)
Staff(staffNo, name, salary, branchNo, position, DOB)
propertyForRent(propertyNo, staffNo, rent)
Answer the following queries in relational algebra
1. Find the names of Staff who work in Delhi
2. List the staffNo of Staff who have not rent any property **6**

Ans.
1. $\prod_{name} (\sigma_{city = 'Delhi'}$ (Staff join Branch))
2. $\prod$ staffNo (Staff) - $\prod$ staffNo (propertyForRent)

Q.176 What is the result relation if we perform the relational algebra operator as shown below
OFFICE $\div \prod_{HQ} (\sigma_{NAME = 'Signet' \text{ or } NAME = 'Dell'} (PUBLISHER))$
for the relations given below **4**

PUBLISHER

PID	NAME	HQ
01	Acer	Mumbai
02	Signet	New York
03	Dell	London
04	HP	Sydney

OFFICE

PID	CITY
01	New York
01	London
02	London
02	Sydney
03	London
03	New York
04	New York
04	London

Ans.

PID
01
03
04

Q.177 In SQL, differentiate between
1. a subquery and a join
2. WHERE and HAVING clauses
3. DROP and DELETE
4. LEFT OUTER JOIN and RIGHT OUTER JOIN **8**

Ans:
1. If we need to obtain information from one or more tables then either subquery or join can be used. If the columns that are to appear in the result table are from different tables then a join must be used.
2. Predicates in the WHERE clause applies to each tuple whereas the predicate in the HAVING clause applies to groups.
3. DELETE deletes one or more tuples of a given relation from the database. DROP is used to remove a table from the schema. Whereas DROP removes the definition of

the relation as well as the data in the given relation, delete only deletes the tuples but maintains the table definition.

4. In left outer join tuples from the left-hand-side relation that do not match any tuple in the right-hand-side relation are padded with nulls and are added to the result of the left outer join. Similarly, in right outer join tuples from the right-hand-side relation that do not match any tuple in the left-hand-side relation are padded with nulls and are added to the result of the left outer join.

Q.178 For the relations given in Q 175, answer the following queries in SQL

1. How many properties cost more than RS.5000/- per month
2. Give all Managers a 5% pay increase
3. Find the number of staff working in each branch and the sum of their salaries

(2 + 3 + 3)

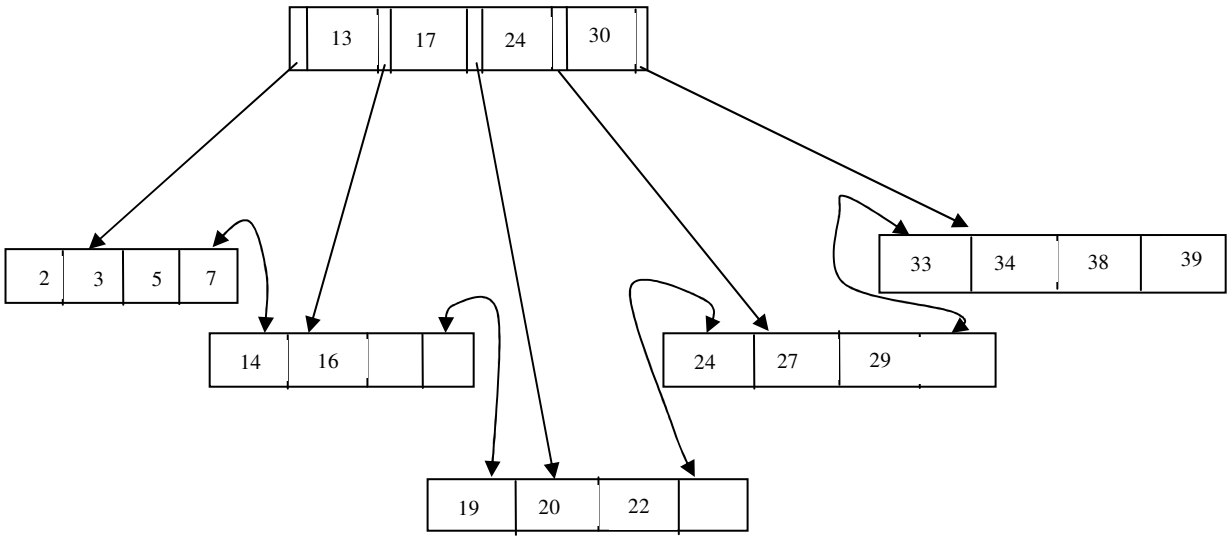
Ans:

1. SELECT COUNT(*)
   FROM PropertyForRent
   WHERE rent > 5000
2. UPDATE Staff
   SET salary = salary*1.05
   WHERE position = 'Manager'
3. SELECT branchno, COUNT(staffNo), SUM(salary)
   FROM Staff
   GROUP BY branchNo

Q.179 Define a B+ tree. **(2)**

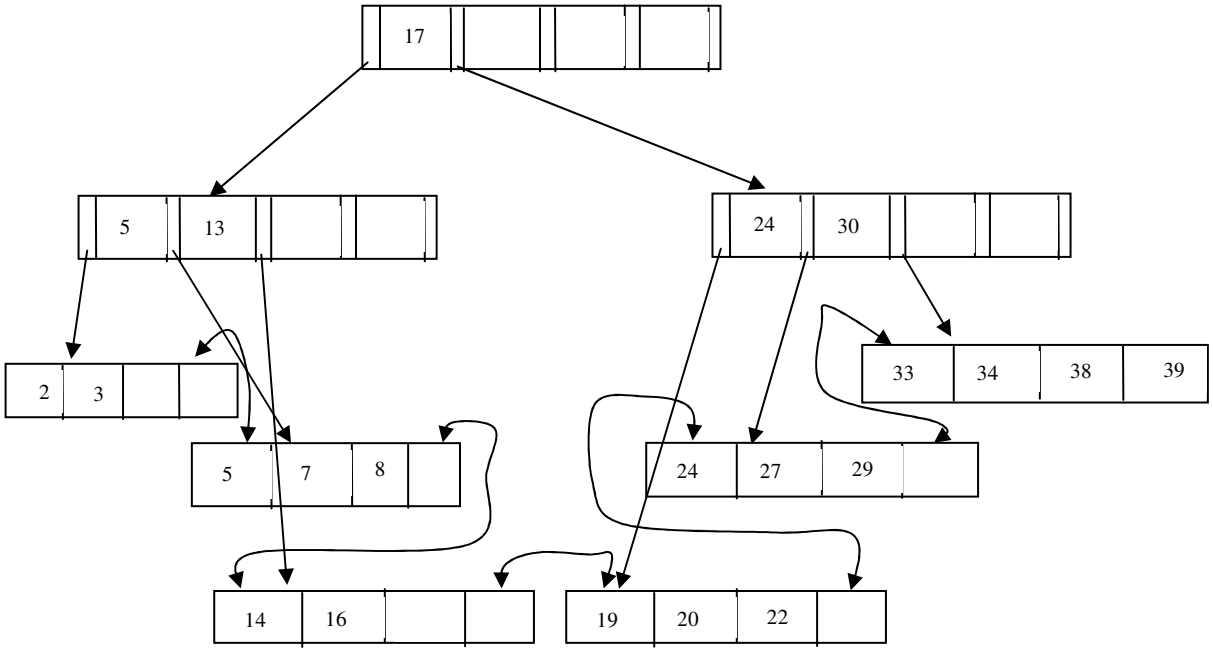
Ans: A B+ tree is a dynamic, height balanced index structure . Each node except the root has between d and 2d entries. The number d is called the order of the tree.

Q.180 Consider a B+ tree with order 2 with the following elements

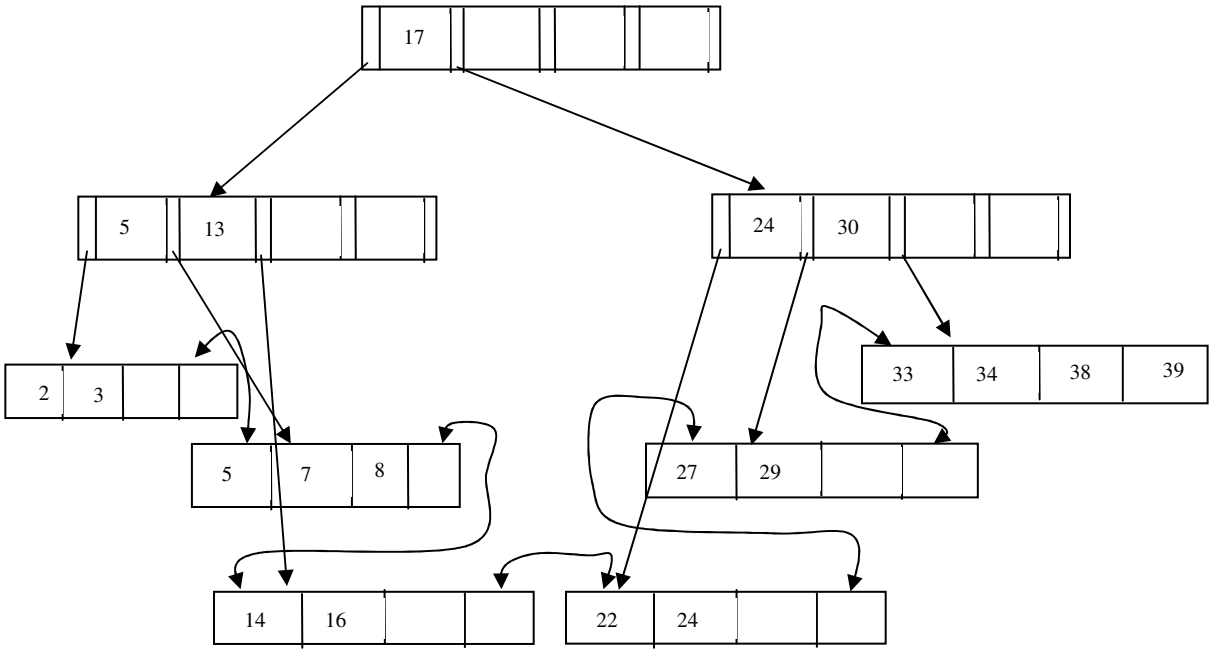


(i) Insert entry 8
(ii) Delete entry 19 and 20 5 * 2

Ans**. (i)**



(ii)



Q.181 Describe the definition of a transaction in SQL

Ans. The SQL transaction begins implicitly. Transactions are ended by either Commit work which commits the transaction and begins a new one Rollback work which causes the transaction to abort.
If a program terminates without either of these statements then updates are either committed or rolled back and the choice is implementation dependent.

Q.182 Define dependency preserving and lossless join decomposition **4**

Ans: A decomposition of R into $R_1$and R_2 is a lossless decomposition, if at least one of the following dependencies is in F^+ :
$R_1 \cap R_2 \rightarrow R_1$
$R_1 \cap R_2 \rightarrow R_2$
A decomposition of R into R_1, R_2 , … R_n is a dependency preserving decomposition, if either F' = F or $F'^+ = F^+$ where $F' = F_1 \cup F_2 \cup \ldots \cup F_n$ and F_i is the restriction of F to R_i.

Q.183 Given R = ABCD with the FD set F = { A → B, B → C, C → D}
Determine all 3NF violations. Decompose the relations into relations which are in 3NF. **8**

Ans: The only candidate key is A. The FD B → C violates 3NF as B is not a superkey and C is not a prime attribute. The FD C → D violates 3NF as C is not a superkey and D is not a prime attribute. The decomposition is R1 (AB) with key A and FD = A → B
R2 (BC) with key B and FD = B → C and R3 (CD) with key C and FD = C → D

Q.184 Determine a candidate key for R = ABCDEG with the FD set
F = { AB → C, AC → B, AD → E, B → D, BC → A, E → G } **4**

Ans: $BC^+ = R$. $B^+ = BD$ and $C^+ = C$. Therefore, BC is the candidate key.

Q.185 What are the objectives of query processing? **3**

Ans: The aim of query processing is to transform a query written in a high level language into a correct and efficient execution strategy expressed in a low level language and execute the strategy to retrieve the data.

Q.186 Explain the transformation rules that apply to
(a) Selection
(b) Projection
(c) Theta join operations **9**

Ans:
(i) Conjunctive selection operations can cascade into individual selection operations
$\sigma_{p \wedge q \wedge q \wedge r}(R) = \sigma_p(\sigma_q(\sigma_r(R)))$
(ii) Commutativity of selection operations
$\sigma_p(\sigma_q(R)) = \sigma_q(\sigma_p(R))$
(iii) In a sequence of projection operations only the last one is required
(iv) Commutativity of projection and selection
(v) Commutativity of theta join
(vi) Commutativity of selection and theta join
(vii) Commutativity of projection and theta join

(viii) Associativity of theta join

Q.187 When are two schedules said to be view equivalent? 4

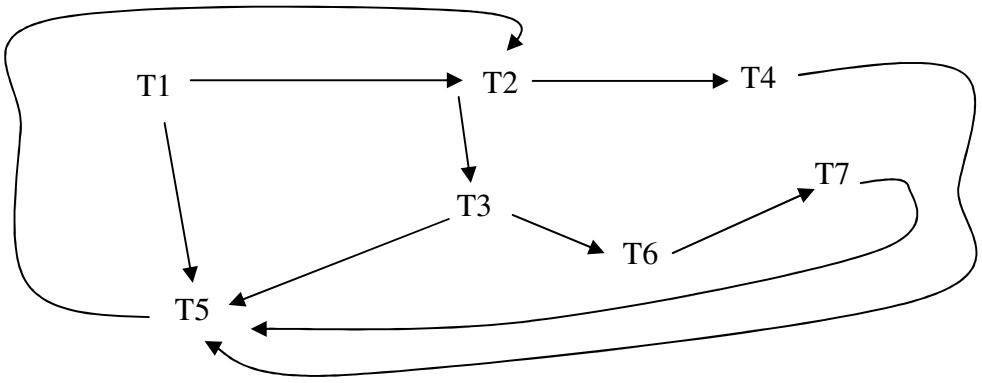
Ans:
Two schedules S and S' are said to be view equivalent if the following three conditions are met

a) For each data item Q, if a transaction Ti reads the initial value of Q in schedule S, then transaction Ti must in schedule S' also read the initial value of Q.
b) For each data item Q, if a transaction Ti executes read(Q) in schedule S and that value was produced by Tj (if any) , then transaction Ti must in schedule S' also read the value of Q produced by Tj..
c) For each data item Q, if a transaction if any that performs the final write(Q) in schedule S must perform the final write(Q) in schedule S'.

Q.188 Define deadlock. Produce a wait-for graph for the following transaction scenario and determine whether a deadlock exists

Transaction	Data items locked	Data items waiting for	
T1	x2	x1,x3	
T2	x3, x10	x7, x8	
T3	x8	x4, x5	
T4	x7	x1	
T5	x1, x5	x3	
T6	x4, x9	x6	
T7	x6	x5	**2 + 8**

Ans: An impasse that may result when two or more transactions are each waiting for locks to be released that are held by the other



Since there is cycle in the wait- for graph ($T_2 \rightarrow T_4 \rightarrow T_5 \rightarrow T_2$) in the graph, therefore, system is in a deadlock.

Q.189 Define granularity, hierarchy of granularity of locks and multiple granularity locking. Describe the modified two phase locking with multiple granularity locking. **6**

Ans: The size of the data item chosen as a unit of protection by a concurrency control protocol is granularity.
When the granularity is represented in a hierarchical structure where each node represents data items of different sizes we get the hierarchy of granularity.
In multiple granularity locking there are three types of locks – shared, exclusive and intention lock.
Modified two-phase locking strategy

a) No lock can be granted once any node has been unlocked
b) No node may be locked until its parent has an intention lock
c) No node may be unlocked until all its descendants are unlocked.

Q.190. What are the advantages of using a DBMS

8

Ans: The main advantages of using a DBMS are

Controlled redundancy

Every user group maintains its own files in traditional systems. For example, in the University system two different programmers will maintain different files for Courses, Teachers and Student information. Much of the information about the teachers teaching different courses to different students will be repeated in these files. This redundancy leads to several problems. Firstly, there is no single place to update data. Secondly, there is a waste of storage. Thirdly, the files may become inconsistent. The information about student registration may be different in different files.

Restricting Unauthorized access

When multiple users use the data base it is necessary to control access to sensitive data. Not all people may have access to the salary of employees in an organization. This is ensured by the DBMS.

Persistent Storage

The data that is stored survives beyond the program that created it

Deductive rules

Deductive DBMS permit the specification of inference rules to derive new information from the stored facts.

Multiple user interfaces

Some DMBS provide different user interfaces so that the user can choose the on that best suits his needs. Query languages, Data manipulation languages, graphical user interfaces are some of the interfaces that are provided.

Complex relationships

A DBMS is capable of representing a variety of complex relationships among data.

Integrity Constraints

It is possible to express constraints which are enforced. For example, if there is a constraint that no employee can be aged more than 62 years of age that this can be specified and enforced by a DBMS.

Backup and Recovery

In a DBMS environment, the system automatically recovers the data base to a consistent state when there are program and system failures.

Q.191 Define the following

1. Attribute Inheritance
2. Cascading rollback
3. Exec statement in SQL

4. Project-Join normal form **8**

Ans:

1. **Attribute Inheritance**
   The attributes of a higher level entity set are inherited by a lower level entity set created by specialization-generalization hierarchy. That is, all the attributes of the higher level entity set are also the attributes of the lower level entity set.
2. **Cascading rollback**
   The phenomenon in which a single transaction failure leads to a series of transaction rollbacks is called Cascading rollback.
3. **Exec statement in SQL**
   Exec statement has the form
   EXEC SQL <embedded SQL statement> END-EXEC
   Embeded SQl statements are of a form similar to non embedded SQL statements.
4. **Project-Join normal form**
   A relation R is in PJNF with resoect to a et of functional ependencies D if for all join dependencies of the form $*(R_1, R_2, \ldots, R_n)$ and $R = R_1 \cup R_2 \cup \ldots, \cup R_n$ at least one of the following holds:

$*(R_1, R_2, \ldots, R_n)$ is trivial
Every R_i is a superkey of R

Q.192 What are the disadvantages of file-processing system? **(4)**

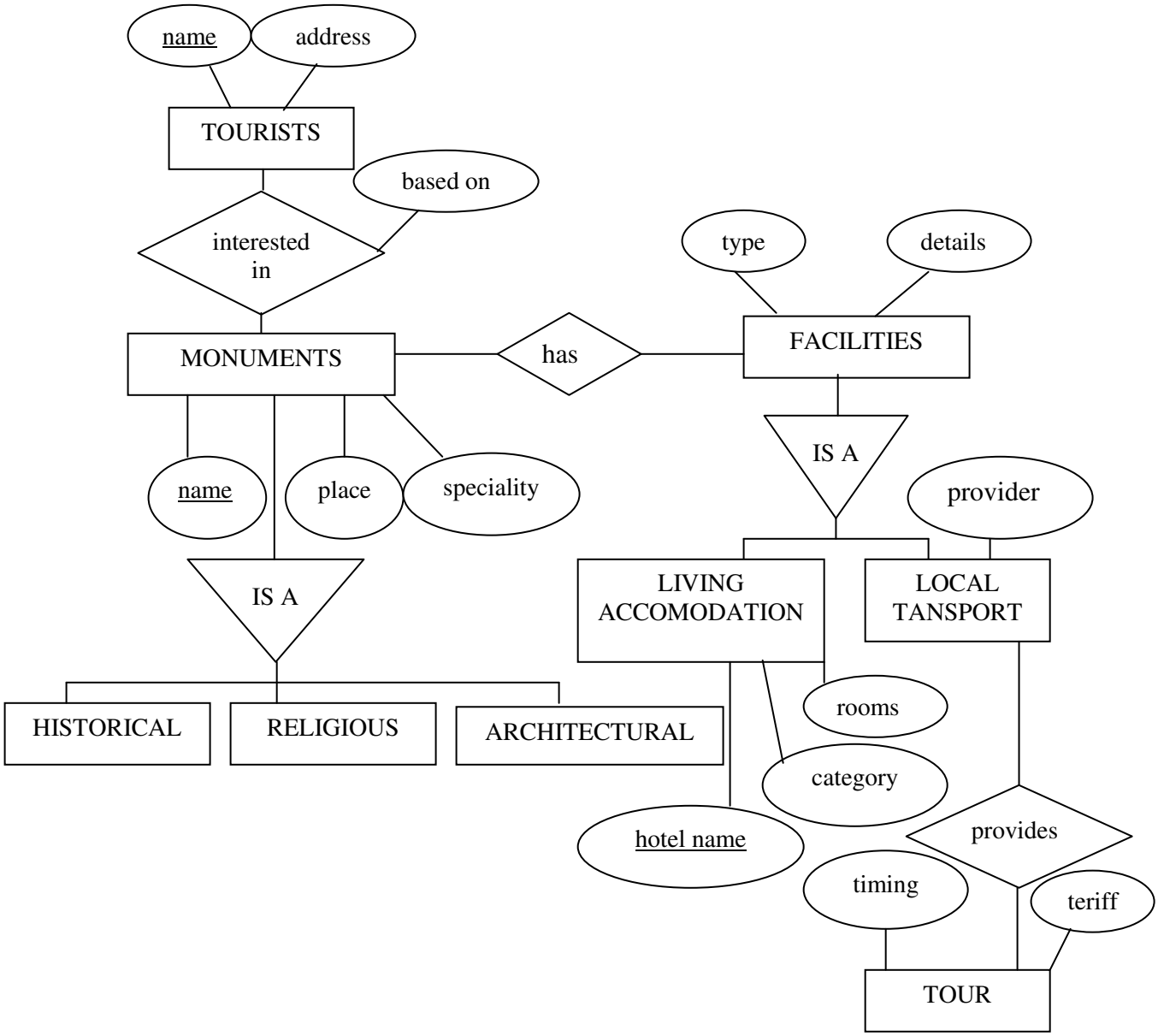
Ans:

The Disadvantages Of File-Processing System

1. **Data redundancy and inconsistency** – Data redundancy means unnecessary duplication of data. In file processing systems, the information may be duplicated in several places (files) due to the different structure. This redundancy leads to higher storage access cost. In addition, it may lead to data inconsistency, means the various copies of the same data may no longer agree.
2. **Difficulty in accessing data –** The conventional file-processing environments do not allow needed data to be retrieved in convenient and efficient manner
3. **Data isolation –** Because data are scattered in various files and may be in different formats, writing new program to retrieve the appropriate data is difficult.
4. **Security problem –** In file processing system. Enforcing security constraints is difficult.

Q.193 The tourism department wishes to computerize its data. The information consists of monuments of tourist interest, their location and history. Monuments are classified according to historical, religious and architecture importance. The list of facilities available at each sport is also available. These give (i) living accommodation in terms of hotels, their names, category and the number of rooms available and (ii) local transport facilities in terms of service provider name, tours with their tariff and timing. Arrive at an E-R diagram by identifying the entities, relationships, attributes, primary keys and cardinality. **(10)**

Ans:



Q.194 Describe the division and the join operation of the relational algebra. Give an example for each. Express each of them in terms of the basic operations. **(5)**

Ans: Join (⋈)- Allows the combining of two relations to form a single new relation. The combined operations of Cartesian product followed by selection operation is the join operation. There are different types of join operations: Theta, Equi, Natural. and outer Join. Theta join can be

$$R=P_{A\theta B} \bowtie Q$$

Join can be defined in terms of basic operations as: $\sigma_{A\theta B}$ (PxQ)
For example

Employee

E#	Name	D#
101	Jones	D1
103	Smith	D1
104	Lalonde	D2

Department

D#	Dname
D1	Sales
D2	Accounts

The natural join of employee and department

E#	Name	D#	Dname
101	Jones	D1	Sales
103	Smith	D1	Sales
104	Lalonde	D2	Accounts

Division (÷) Produces a relation R(X) that includes all the tuples t[X] in P(Z) that appear in relation P in combination with every tuple from Q(Y), where Z=X∪Y. The Cartesian product of relation Q and R is a subset of P (P⊇Q x R). Useful when a query involves the phrase " for all objects having all the specified properties."

For example : R= P÷Q

P

A	B
a_1	b_1
a_1	b_2
a_2	b_1
a_3	b_1
a_4	b_2
a_5	b_1
a_5	b_2

Q

B
b_1
b_2

R(result)

A
a_1
a_5

Q

B
b_1

Then R is

A
a_1
a_2
a_3
a_2

(a)

Q

B
b_1
b_2
b_3

Then R is

A

(b)

Q

B

Then R is

A
a_1
a_2
a_3
a_4
a_5

Division can be expressed as a sequence of projection, cross product, and difference operations as follows:

1. $R_1 \leftarrow \Pi_A(P)$
2. $R_2 \leftarrow \Pi_A(Q \times R_1) - P)$
3. $R \leftarrow R_1 - R_2$

Q.195 Consider the schema
Airport (code, name, city, country)
Flight (number, airline, from_airport_code, to_airport_code)
Reservation(flight_number, seat_number, date, passenger_name)
Answer the following using relational algebra

(i) List the flight numbers of flights that take off from India
(ii) List the passenger who are on flight number 'SA 747'.
(iii) List all the flight information for Indian Airlines and Jet Airways.

Ans: (i) Π_{number} ($\sigma_{country=India}$ (FLIGHT $\bowtie$ from _airport_code=codeAIRPORT))
(ii) $\Pi_{passenger\_name}$ ($\sigma_{flight\_number='Sa\ 747'}$(RESERVATION))
(iii) $\sigma_{airline('IndianAirlines'\ \wedge\ 'Jet\ Airways')}$ (FLIGHT)

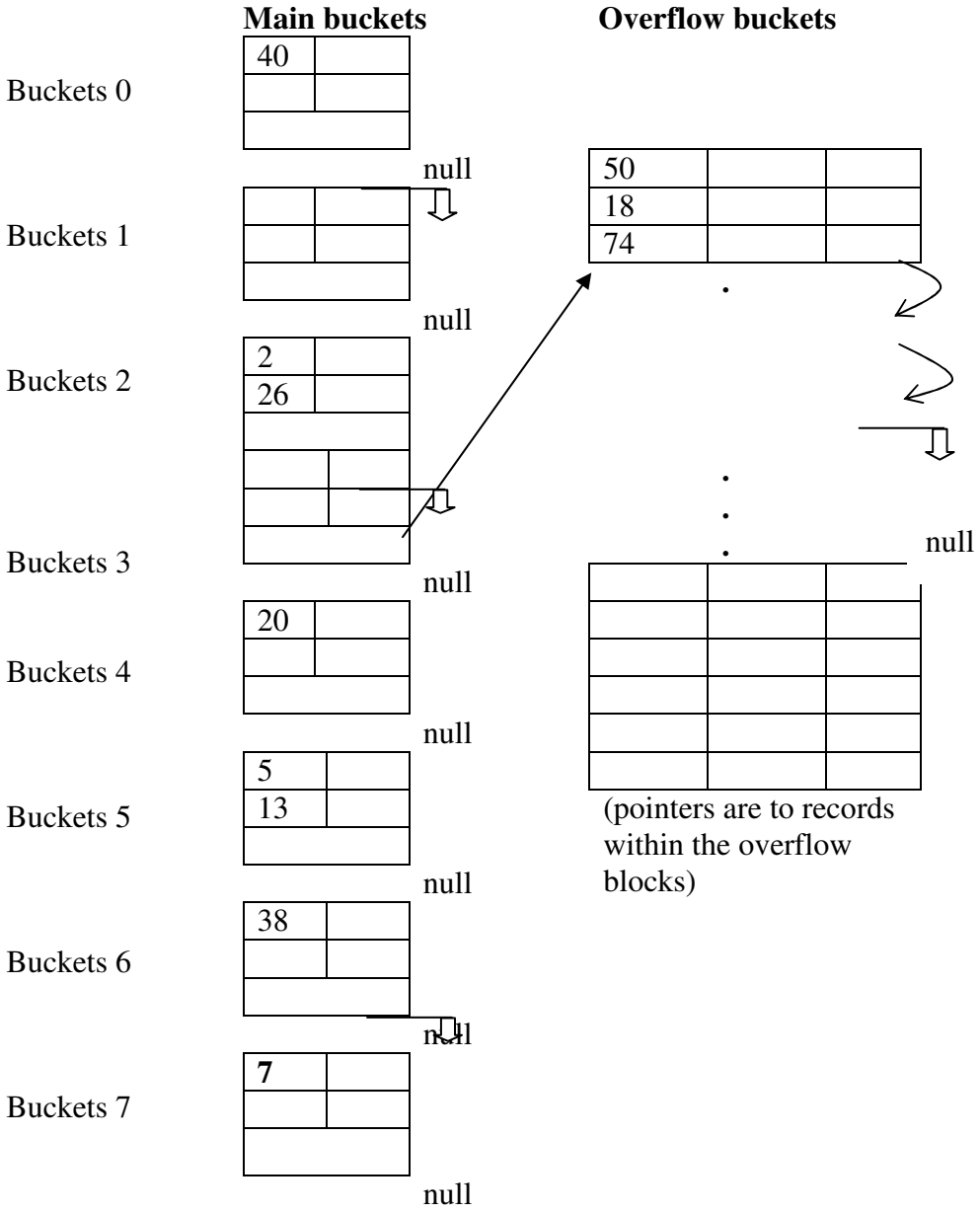
Q. 196 What is the difference between primary index and secondary index? **(4)**

Ans:

Primary Index	Secondary index
1. It is an ordered file whose records are of fixed length with two fields.	1. It provides a secondary means of accessing a file for which some primary access already exists.
2. Only based on the primary key.	2. May be based on candidate key or secondary key.
3. The total number of entries in the index is the same as the number of disk blocks in the ordered data file.	3. It has a large number entries due to duplication.
4. Primary index is a king of nondense (sparse) index.	4. Secondary index is a kind of dense index.
5. There may be at most one primary index for a file	5. There may be more than one secondary indexes for the same file.
6. Needs less storage space.	Needs more storage space and longer search time.

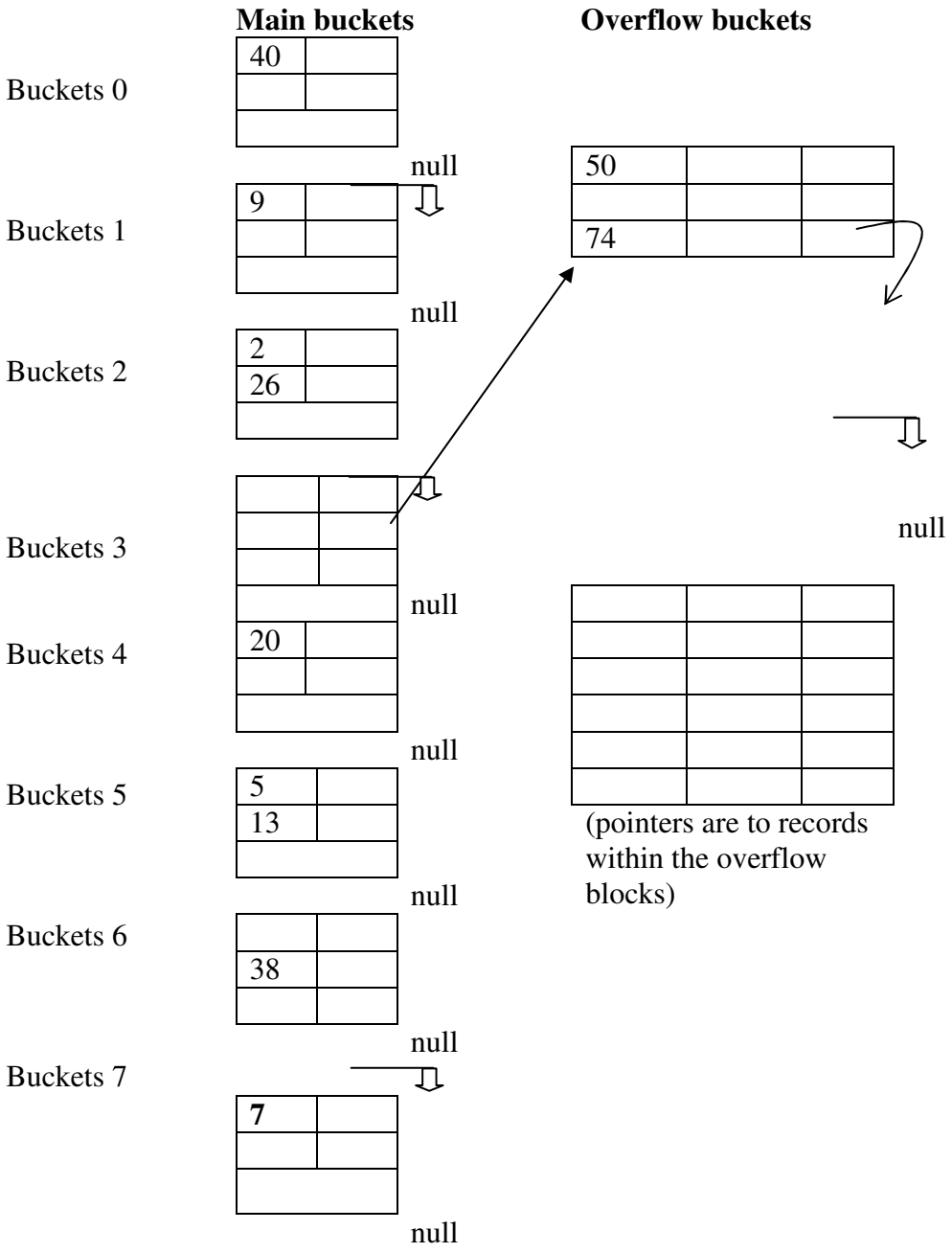
Q.197 Suppose there is a hash function h(x)=x mod 8. Show the hash structure for a file with the following key values 2,5,7,26,13,20,38,40,50,18,74. Assume a bucket size of 2 records. **(6)**

Ans:



Q.198 Explain how you would delete the record with key 18 and then add a record with key 9? What is the hash structure after the deletion and after the insertion?

Ans: To delete the record with key 18 the record pointer of the previous record i.e. 50 will be pointing to the next record of the 18, i.e. 74, and the space of the record with key 18 will be added to the pool of free space. The record with key 9 is added to the main bucket 1 because there is a space available in it. The hash structure after the deletion and after the insertion will be:



Q. 199 Consider the data base with two areas A1 and A2 with files F1, F2 and A2 with files F3. F1 has records r_{11} to r_{19}, and F2 has records r_{21} to r_{29}, F3 has records r_{31} to r_{39}.

(i) What is a granularity hierarchy? Draw the granularity hierarchy for the above database **(4)**

(ii) Suppose transaction T1 wants to read record r_{35} and T2 wants to read file F3 what are the locks to be placed? Can T1 and T2 execute concurrently? Justify. **(4)**

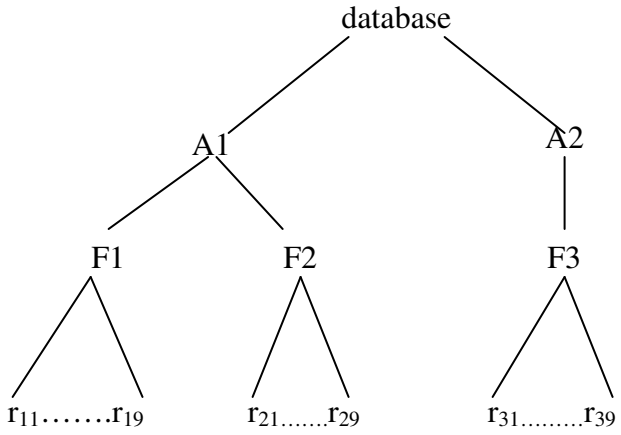
(iii) Is it possible for T1 to unlock file F3. Justify. **(3)**

(iv) If a transaction T3 now wants to write r_{11} will it be allowed to proceed? **(3)**

Ans: (i) All concurrency control techniques assumed that the database was formed of a number of named data items. A database item could be chosen to be one of the following:

- A field value of a database record
- A database record
- A disk block
- A whole file
- The whole database

It seems appropriate that a database system support multiple levels of granularity, where the granularity level can be different for various mixes of transactions, called as granularity hierarchy.



Granularity Hierarchy

(ii) In granularity hierarchy, a lock can be requested at any level. However, additional types of locks will be needed to efficiently support such protocol. When transaction T1 want to read the record r_{35}, the record r_{35} will be locked with shared lock and the higher level items are locked with intention shared mode locks. When transaction T2 want to read the file F3, the file F3 will be locked with shared lock and the higher level items are locked with intention shared mode locks. T1 and T2 can execute concurrently because both the transactions performing the only the read – read operations. Read operations will not effect the consistency of the database.

(iii) Yes, The transaction T1 can unlock the file F3 because if any transaction acquired a lock then it can release that lock at any time.

(iv) Yes, transaction T3 will be allowed to write record r_{11} because neither its lower level items nor higher level items are already locked. Hence, it will not cause any inconsistency.

Q.200 When shadow paging is used, show the steps that are executed when a transaction executed write (X) operation and X resides on the 1^{th} page.

Ans:

In the shadow page scheme, the database is considered to be made up of logical units of storage called pages, the shadow page scheme handles the write operation, e.g., write(x),to a given page(1^{th} page) as follows;

- A free block of non-volatile storage is located from the pool of free blocks accessible by the database system.
- The page to be modified (1^{th} page) is copied onto this block.

- The original entry in the current page table is changed to point to this new block.
- The updates, i.e. write(X), are propagated to the block pointed to by the current page table, which in this case would be the newly created block.

Q.201 Define entity integrity and referential integrity. How does SQL allow specification of these?

Ans:
Entity Integrity Rule- If the attribute A of relation R is a prime attribute of R then A cannot accept null values.
Referential Integrity Rule- In referential integrity. It is ensured that a value that appears in one relation for a given set of attributes also appears for a certain set of attributes in another relation.
In SQL, entity integrity and referential integrity rules are implemented as constraints on the relation called as primary Key constraint and reference Key constraint respectively.
These constraints can be specified with relation at the time of creation of the relations or after the creation of the relations by altering the definition of the relations. For example:

```
Create Table Dept
      (Deptno      Number primary key.
       DName       Varchar2(15));
Create Table Emp
      (Empno       Number primary Key.
       EName       Varchar2(15),
       Job         Varchar2(10),
       Deptno      Number  References Dept(Deptno));
```

Q.202 Let R (A,B,C) and S(B,C,D) be the relations

R

A	**B**	**C**
a1	b1	c2
a2	b4	c1
a3	b3	c3
a2	b3	c5
a1	b1	c2

S

B	C	D
b1	c1	c2
b2	c4	d1
b3	c3	d4

Computer the following SQL statements for the above relations
(i) Select*From R,S
(ii) Select A from R,S Where R,B=S,B
(iii) Select A,C From R Group by B
(iv) Select R,A From R where R,B=NOT ALL
(Select S,B From S Where R.C= S.C) **(10)**

Ans: (i) It will returns the Cartesian product of relations R and S.

A	R.B	R.C	S.B	S.C	D
a1	b1	c2	b1	c1	c2
a1	b1	c2	b2	c4	d1
a1	b1	c2	b3	c3	d4
a2	b4	c1	b1	c1	c2
a2	b4	c1	b2	c4	d1
a2	b4	c1	b3	c3	d4
a3	b3	c3	b1	c1	c2
a3	b3	c3	b2	c4	d1
a3	b3	c3	b3	c3	d4
a2	b3	c5	b1	c1	c2
a2	b3	c5	b2	c4	d1
a2	b3	c5	b3	c3	d4
a1	b1	c2	b1	c1	c2
a1	b1	c2	b2	c4	d1
a1	b1	c2	b3	c3	d4

(ii)

A
a1
a1
a3
a2

(iii) This is an invalid SQL command because in SELECT clause we can use only group expressions (e.g.. column name(s) on which the grouping is done and/ or aggregate functions) when we are using GROUP BY clause. In the given command A and C are not group expressions.

(iv) **<>ALL**

A
a1
a2
a2
a1

Q.203 Consider the relation R(A, B, C, D, E) with the set of function dependencies F={A, B → C, D → E, A → D}

(i) Is AB a candidate Key? Justify. **(2)**

(ii) Giving reasons find out whether R is in 3NF or BCNF. **(4)**

(iii) Consider the decomposition of R into R1 (A,B,C) and R2 (A,D,E). Is the decomposition lossless and dependency preserving? Justify. **(6)**

(iv) Define fifth normal form. **(2)**

Ans:

(i) Yes, AB qualifies as a candidate of the given relation R. Because, AB can derive all other attributes CDE as illustrated below:

- A can derives the value of D based on the FD $A \rightarrow D$.
- D can derives the value of E based on the FD $D \rightarrow E$ or transitively A derives the value of E.
- AB can derive the value of C based on the FD $AB \rightarrow C$.

(ii) Above relation neither in 3NF nor in BCNF. Even it is not in 2NF because the Key of the relation is AB and there is partial functional dependency $A \rightarrow D$. which is not permissible in 2NF relation. Therefore, it is justified the while the given relation not in 2NF then it is neither in 3NF nor in BCNF.

(iii) Yes, the given decompositions R1 and R2 of the relation R is dependency preserving because $(F1 \cup F2)^+ \equiv F^+$. The given decompositions are also lossless because the Key AB of the relation R is preserved in one of the decomposition R1.

(iv) A relation schema R is in fifth normal form (5NF) (or project join normal form [PJNF]) with respect to a set of F of functional. Multivalued, and join dependencies if, for every nontrivial join dependency JD $(R_1, R_{2,\ldots} R_n)$ in F^+ (that is, implied by F), every R_1 is a superkey of R,

Q.204 Discuss with respect to SQL

(i) How nulls are treated in comparison operator.
(ii) The count function.
(iii) The check clause.
(iv) The difference between drop table R and delete from R. **(8)**

Ans:

(i) With comparison (relational) operators, the null values are ignored because we cannot derive the relation with the given operand. Nulls cannot be equate. Nulls can be checked by containment using IS NULL or IS NOT NULL operators.

(ii) The COUNT function returns the number of tuples or values specified in a query. The count function has two types of syntax: (1) COUNT (*) – returns the number of rows in the result query, (2) COUNT ([DISTINCT] <column>) – in this way, the NULL values are discarded and if used then the duplicates are also discarded in the count.

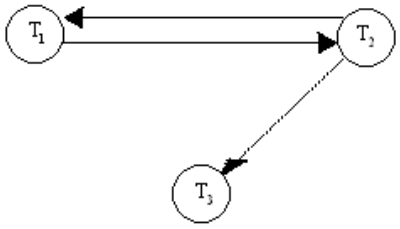
(iii) The CHECK clause is used to at the end of a CREATE TABLE statement specify table constraints. This is called table-based constraint because it applies to each tuple individually and are checked whenever a tuple is inserted or modified. The CHECK clause can also be used to specified more general constraints using the CREATE ASSERTION statement of SQL.

(iv) DROP TABLE command deletes all the records along with the table definition. This command will automatically committed and cannot be rolled back. But . DELETE FROM <table> command deletes only the records; the table definition remains existed. The effect of this command may be rolled back, if required.

Q.205 Consider the schedule of three transactions T1,T2 and T3.

$S: r_2(A); r22_1(B); w_2(A); r_2(B); r_3(A); w_1(B); w_3(A); w_2(B)$

Where r stands for READ, w for WRITE and determine if the schedule is seializable. If so, give the schedule.



The given schedule is not serializable because the precedence graph is cyclic.

Q.206 Under what condition does the selection operation distribute over theta join operation? **(2)**

Ans:

The conditions are:

- If all the attributes in the selection condition c involve only in the attributes of one of the relations being joined-say, R – the two operations can be commuted a follows:

$$\sigma_C(R \bowtie S) \equiv (\sigma_C(R)) \bowtie R$$

- If the selection condition c can be written as c1 and c2, where condition c1 involves only the attributes of R and/or condition c2 involves only the attributes of S, the operations can commute as follows:

$$\sigma_C(R \bowtie S) \equiv (\sigma_{C1}(R)) \bowtie (\sigma_{C2}(S))$$

OR

$$\sigma_C(R \bowtie S) \equiv \sigma_{C2}((\sigma_{C1}(R)) \bowtie S)$$

OR

$$\sigma_C(R \bowtie S) \equiv \sigma_{C1}((R \bowtie (\sigma_{C2}(S)))$$

Q.207 For the relations

Emp(name, fname, address, dno)
Dept(deptno, dname, address)

And the query

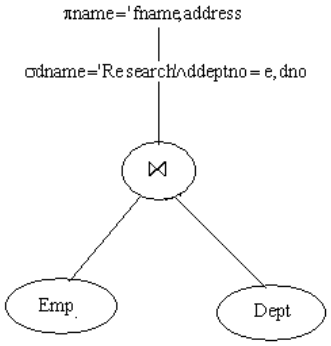
Π fname, ename, address (σ dname= 'Research' AND d,deptno=e. dno(EMP$\bowtie$Dept))

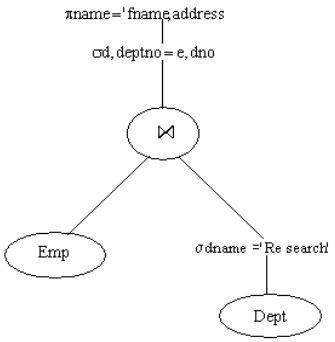
(i) Draw the initial query tree. **(2)**

(ii) Optimize the query write(X) operation and X resides on the 1th page. **(4)**

Ans:

$\pi_{\text{name, fname, address}}$ ($\sigma_{\text{dname='Research'^d.deptno=e.dno}}$(Emp$\bowtie$Dept))





Distribution of selection peration

The initial query tree **The optimized query tree**

Q.208 Compare shadow paging with log based recovery methods. **(4)**

Ans:

Shadow paging – In the shadow page scheme, the database is considered to be made up of logical units of storage called pages. The shadow paging scheme uses two page tables for transaction that is going to modify the database. The original page table is called the shadow page table and the transaction addressed the database using another page table known as the current page table. In case of a system crash, before the transaction commits, the shadow page table and the corresponding blocks containing the old database, which was assumed to be in a consistent state, will continue to be accessible. The recovery from system crash is relatively inexpensive, is an advantage of this scheme. The main disadvantages of the shadow scheme are: (1) the problem of scattering, and (2) the original version of the changed blocks pointed to by the shadow page table have to be returned to the pool of free blocks.

Log Based Recovery Method – The system log, which is usually written on stable storage, contains the redundant data required to recover from volatile storage failures and also from errors discovered by the transaction or the database system. System log is also called as log and has sometimes been called he DBMS journal. In case of system crash the log is processed from the beginning to the point where system crashes. All the transaction, those have been recorded their-of-transaction marker, will be committed otherwise rolled back. The simplicity and easier to use the main advantages of this method. The disadvantage are: (1) inefficient for the application, which have large volume of information, (2) in case of system crash with loss of volatile information, the log information collected in buffers will also be lost and transaction that had been completed for some period prior to the system crash may be missing their respective end-of-transaction markers in the log; if such transactions are rolled back then likely to be partially undone.

Q.209 What is a checkpoint? List the operations to be performed by the system when a checkpoint is to be taken. What does the recovery system do if there is a crash. **(6)**

Ans:

In a large on-line database system there could be hundreds of transactions handled per minute. The log for this type of database contains a very large volume of information. A scheme called checkpoint is used to limit the volume of log information that has to be handled and processed in the event of a system failure involving the loss of volatile information. The checkpoint scheme is an additional component of the logging scheme.

A checkpoint operation, performed periodically, copies log information onto stable storage and put the checkpoint marker at that time in the log. For all transaction active at checkpoint, their identifiers and their database modification actions. Which at that time are reflected only in the database buffers, will be propagated to the appropriate storage.

If there is a crash, it scans the log in a backward direction from the time of system crash. Initially, all the transaction are added to the undo list. After that, if it finds that a transaction in the undo list has committed, that transaction is removed from the undo list and placed in the redo list. The redo list contains all the transactions that have to be redone. All transactions that were committed before the checkpoint time need not be considered for the recovery operation. Therefore, the overhead of the recovery from a system crash is reduced. However, in a system that uses the transaction paradigm. Checkpoint is a strategy to minimize the search of the log and the amount of undo and redo required to recover from a system failure with loss of volatile storage.

Q.210 Write short notes on

(i) Clustering file organization.
(ii) Deadlock detection and recovery.
(iii) Outer join
(iv) Mapping of ISA relationship of E-R diagram to tables. **(3.5x4=14)**

Ans:

(i) If a record of file are physically ordered on a nonkey field – which does not have a distinct value for each record – that field is called the clustering field. The type of index that is created using clustering field is called as clustering index. Clustering index is used to speed up retrieval of records that have the same value for the clustering field. The file that is organized based on the clustering field and clustering index is called clustering file organization.

(ii) Deadlock Detection – A deadlock is said to occur when there is a circular chain of transaction, each waiting for the release of a data – item held by the next transaction in the chain, the algorithm to detect a deadlock is based on the detection of such a circular chain in the current system in wait – for – graph. The algorithm generates the wait – for – graph at regular intervals and examines it for a chain. The algorithm starts with the assumption that there is no deadlock.

Deadlock Recovery – To recover from deadlock. The cycles in the wait – for – graph must be broken. The common method of doing this is to roll back one or more transactions in the cycles until the system exhibits no further deadlock situation. The process of deadlock recovery must ensure that a given transaction is not continuously the one selected for rollback. The selection of the transaction to be rolled back is based on the following considerations:

- The progress of the transaction and the number of data – items it has used and modified.
- The amount of computing remaining for the transaction and the number of data – items that have yet to be accessed by the transaction.

- The relative cost of rolling back a transaction.

(iii) Outer join – If there are any values in the one table that do not have corresponding value(s) in the other, in an equi-join that will not be selected. Such rows can be forcefully selected by using the outer join, The corresponding columns for that row will have NULLs. There are actually three forms of the outer-join operation: left outer join (_X), right outer join (X_) and full outer join (_X_).

(iv) There are several option for mapping of IS_A relationship to relation. For the mapping. Convert each specialization with m subclasses and superclass C into relation schemas using one of the four following options:

(a) Multiple relations – Superclass subclasses: In this option, one relation schema is created for the subperclass and for the each subclasses, the separate relation schemas are with the respective attributes and the key of the superclass. This option works for any specialization (total or partial, disjoint or overlapping).

(b) Multiple relation – Subclass relations only: In this option. Only the relation schemas are created for each subclasses separately by including all the attributes of the superclass with each subclass. (This option only works for a specialization whose subclasses are total.

(c) Single relation with one type attribute: In this option, only one relation is created by combining all the attributes of the superclass and all the subclasses with an extra attributes may named as 'type' to distinguish each entity. This option works only for a specialization whose subclasses are disjoint, and has the potential for generating many null values if many specific attributes exist in the subclasses.

(d) Single relation with multiple type attributes: In this option, only one relation is created by combining all the attributes of the superclass and all the subclasses with some extra attributes to distinguish each entity. This option works for a specialization whose subclasses are overlapping (also works for disjoint specialization).