CHAPTER 8

DATABASES

Syllabus: ER model, relational model (relational algebra, tuple calculus), database design (integrity constraints, normal forms), query languages (SQL), file structures (sequential files, indexing, B and B+ trees), transactions and concurrency control.

8.1 INTRODUCTION

Database is a collection of organized information so that it can easily be searched, retrieved, managed and updated. A database management system (DBMS) is a set of programs designed to manage a database. It enables users to store, retrieve and modify information in a database with utmost efficiency along with security features. DBMS is applicable to various day-to-day fields such as transactions in banking; airline/railway/hotel reservations; maintenance of student information in schools/universities; online retails; marketing and sales etc. It also allows its users to create their own databases. Different types of DBMS are available such as hierarchical, network, relational and object oriented.

8.1.1 Traditional File Processing Approach

File processing approach is generally more accurate and faster than the manual database system. Each user is responsible for the defining and implementation of the files required for the specific application. The implementation of required files sometimes creates redundancy of data, for example, one user keeps records for the savings account of the customer and another user may create the loan account of the same customer. This causes the duplicity of the records of the same customer. So, this practice is not feasible for real time applications. There is a need of centralized management of data. The data is created once and then accessed by different users. The data should be shared for different transactions. It should be self-describing in nature, which means the database system contains not only data but it describes the description of the database structure.

8.1.2 Database Management System

A database is a collection of related data. Data is a collection of raw facts or figures, processed to form information. Database management system is a collection of programs for the creation and maintenance of database. It is an efficient and reliable approach to retrieve data for many users. It provides various functions such as:

1. **Redundancy control:** It provides redundancy by removing duplicity of data by following rules of normalization.
2. **Data independence:** It provides independence to application programs from details of data representation and storage. It also provides an abstract view of the data to insulate application code from such details.
3. **Data integrity:** It promotes and enforces some integrity rules for reducing data redundancy and increasing data consistency.
4. **Concurrency control:** It supports sharing of data, so, it has to provide an approach for managing concurrent access of the database. Hence, preserving the inconsistent state and integrity of the data.
5. **Transaction management:** It provides an approach to ensure that either all the updates for a given transaction will execute or that none of them would execute.
6. **Backup and recovery:** It provides mechanisms for backing up data periodically and recovering from different types of failures, thus, preventing loss of data.
7. **Non-Procedural query language:** It provides with query language for retrieval and manipulation of data.
8. **Security:** It protects unauthorized access in the database. It ensures the access to authorized users.

8.2 COMPONENTS OF DATABASE SYSTEMS

DBMS consists of several components, namely software, hardware, data, procedures and data access language. These components are responsible for the definition, collection, management and use of data within the environment. Figure 8.1 shows the components of database system. The description of each component is as follows:

1. **Software:** It is the collection of programs used by the computers within the database system. It is used to handle, control and manage the database. It includes the following software:
   - Operating system software like Microsoft Windows, Linux OS, Mac OS.
   - DBMS software such as Oracle 8I, MySQL, Access (Jet, MSDE), SQL Server etc.
   - Network softwares are used for sharing share the data of database among multiple users.
   - Application programs are developed like C++, VB, dotnet etc. are used to access database in dbms. These are used to access and manipulate the data in the database.
2. **Hardware:** It consists of all system's physical devices such as computers, storage devices, I/O channels, electromechanical devices etc. It also includes peripherals, such as, keyboard, mouse, modems, printers, etc.
3. **Data:** It is the collection of facts. The database contains the data and the metadata.
4. **Procedures:** There are the instructions and rules to design and use the database system. These includes the following:
   - Steps for the installation of DBMS
   - Steps to use the DBMS or application program
   - Steps for the backup of DBMS
   - Steps to change the structure of DBMS
   - Steps for the generation of reports.
5. **Data access language:** The users can use it to access the data to and from the database. The function of data access language is the entry of new data, manipulation of the existing data and the retrieval of the existing data in the database. The most popular database access language is SQL (Structured Query Language). Users can perform these functions with the help of commands. The role of administrator is to access, to create and to maintain the database.
6. **People:** Persons involved to access, to create and to maintain the database are called users. These are of various types according to the role performed by them (Fig. 8.1). These are as follows:
   - **System Administrator:** The role of system administrator is to supervise the general operations of DBMS.
   - **Database Administrator:** The role of database administrator (DBA) is to manage the DBMS.
   - **Database Designer:** The role of database designer is to design the structure of the database.
   - **Application Programmer:** The role of application programmer is to create the data entry forms, reports and procedures.
   - **End User:** The role of end user is to use the application programs by entering new data and manipulating and accessing existing data.

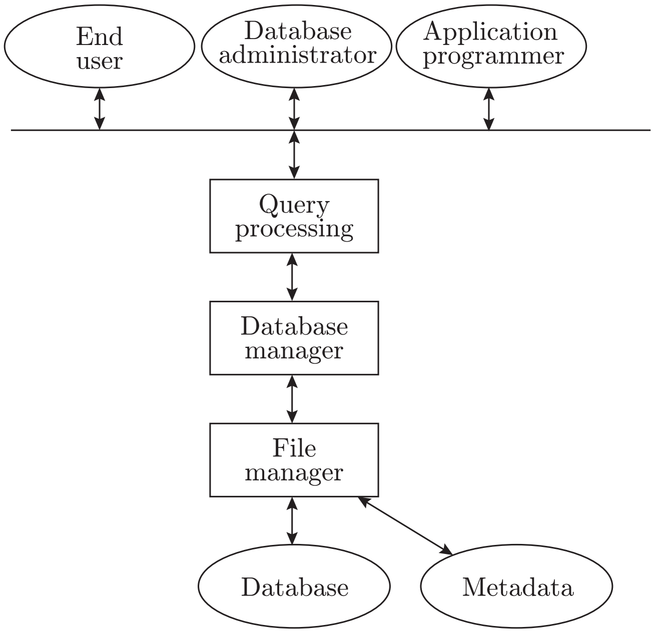


Figure 8.1 Components of database systems.

8.3 DBMS ARCHITECTURE

It is an approach to outlook the database by users. It is means for the representation of data in an understandable way to the users. DBMS architecture can be used to divide the whole system to related and independent modules. It can be of 1-tier, 2-tier 3-tier or *n*-tier.

8.3.1 3-Tier Architecture

DBMS can be most widely used as 3-tier architecture. In this architecture, the database is divided into three tiers depending upon the kind of users (see Fig. 8.2).

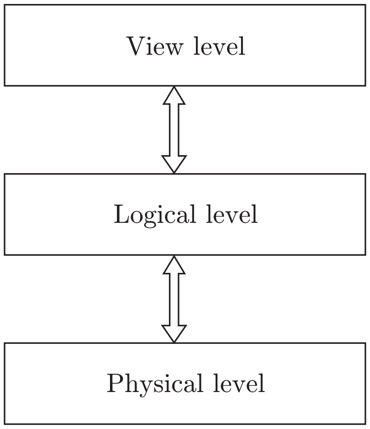


Figure 8.2 The 3-tier architecture of DBMS.

1. **Internal schema or physical level:** It is also called Database Tier. Database exists in this tier. It also includes query processing languages, all relations and their corresponding constraints. It describes the physical storage structure of the database.
2. **Conceptual schema or logical level:** It is also called Application Tier. Server and program exists in this tier. It also describes the structure of the database.
3. **External schema or view level:** It is also called Presentation Tier. End user exists in this tier. An end user is capable of multiple views of database. It also includes all views generated by applications.

8.3.2 Data Independence

Data independence is defined, as the change at one level does not affect the higher level. It is of two types:

1. **Logical data independence:** It is defined, as the change in conceptual schema does not affect the external schema. For example, if the format of the table changes, the data lies in the table should not be changed.
2. **Physical data independence:** It is defined, as the change in internal schema does not affect the conceptual schema. Thus, does not affect the external schema. For example, if the storage system has been changed, then it does not affect the logical structure of the database.

8.4 DATA MODELS

Data model is a collection of tools, which describes data, relationships, constraints and semantics. It gives the logical structure of the database. It describes the relationship of data in the database. Data models are of various types:

1. **Relational Model:** It is a collection of relations (tables). In relational model, each table is stored as a separate file.
2. **Entity–Relationship data model:** This model is based on the notion of real-world entities and relationship among them.
3. **Object-Based data model:** It defines the database as objects, its properties and its operations. In this model, objects with the similar structures comprise a class. The classes are organized into hierarchies. The operations on these classes are performed through methods.
4. **Semi-Structured data model:** It is also known as XML model. This model is used to exchange data over the web. It uses hierarchical tree structures. In this model, data can be represented as elements by using tags.

5. **Network model:** In this model, data is represented as record types. The data in this model has many-to-many relationship.
6. **Hierarchical model:** In this model, the data is represented as a hierarchical tree structure.

8.4.1 Relational Model

The database in relational model is represented as a collection of relations (tables). A relation is a kind of set. It is also a subset of a Cartesian product of an unordered set of ordered tuples. Relational model was proposed by E. F. Codd, which stores data in a tabular form. It consists of a table where rows represent records and columns represent the attributes. It has various terminologies as follows:

1. **Tuple:** It represents a single row of a table, which contains a single record for that relation.
2. **Relation instance:** It represents a finite set of tuples in the relational database system.
3. **Relation schema:** It represents the relation name, that is, table name, attributes and their names.
4. **Relation key:** It represents the unique key for the relation or table. Each row has one or more attributes, which can identify the row in the table uniquely.
5. **Attribute domain:** It represents the predefined value scope of each attribute.

8.4.1.1 Constraints in Relational Model

Constraints are the restrictions that one wishes to apply on database. The following constraints are applied on relational model.

1. **Key constraints:** Each relation has at least one minimal subset of attributes, which can identify a tuple uniquely.
   - No two tuples have identical value for key attributes.
   - Key attribute does not have NULL value.
2. **Domain constraints:** Attributes have specific domain values in real world. For example, value of age can only be positive.
3. **Referential integrity constraints:** If a relation refers to a key attribute of a different relation then that key element must exist.

8.4.1.2 Relational Algebra

It is a procedural query language. It takes instances of relations as input and returns instances of relations as output. Operators are used to perform queries.

Table 8.1 | Notations of fundamental operations of relational algebra

Fundamental Operation	Symbol
Select	σ
Project	π
Union	$\cup$
Intersection	$\cap$
Set difference	$-$
Cartesian product	$\times$
Rename	ρ
Natural join	$\bowtie$

Fundamental operations of relational algebra are as follows (see Table 8.1):

1. **Select:** It is used to select rows from a relation. It is denoted by σ.

   Syntax of select $\sigma_p(r)$, r is relation and p is prepositional logic.

   p uses connectors and operators $\wedge,\vee,=,\neq,<,>,\leq,\geq$.

   For example, $\sigma_{\text{empname}} = \text{"John" (emp)}$.
2. **Project:** It is used to project columns in a relation. It is denoted by π. The duplicate tuples are automatically eliminated.

   Syntax of projects $\pi_A(r)$, r is relation and A is the attribute name in a relation.

   For example, $\pi_{\text{empname, sal}}\text{ (emp)}$
3. **Union:** It returns a relation instance, which contains all tuples occurring in the first relation or in the second relation. It is denoted as $R \cup S$, where R and S are two relations. The duplicate tuples are automatically eliminated.

   This operation is valid for the following:
   - Both relations must have the same number of attributes.
   - Attribute domains must be compatible.
4. **Intersection:** It returns a relation instance, which contains all tuples occurring in both relations. It is denoted as $R \cap S$, where R and S are two relations.
5. **Set difference:** It returns a relation instance, which contains all tuples that occur in the first relation but not in the second relation. It is denoted as $R - S$, where R and S are two relations.
6. **Cartesian product:** It returns a relation instance, which contains all the fields of the first relation followed by all the fields of the second relation. It is denoted as $R \times S$, where R and S are two relations.

7. **Rename:** It returns a relation but without any name. It is used to rename the output relation. It is denoted as ρ.
8. **Joins:** It returns combined information from two or more relations.

   - *Condition joins:* It accepts a join condition c and a pair of relation instances as arguments, and returns a relation instance. It is denoted as $\sigma_c(R \times S)$.
   - *uijoin:* It is a special case of condition joins where the condition c contains equalities.
   - *Natural join:* It is a Cartesian product of two relations. It is denoted by $\bowtie$.

8.4.1.3 Tuple Calculus

It is a non-procedural query language. In this, number of tuple variables is specified. It is represented as $\{t \mid \text{Condition}\}$, wherte t is a tuple variable and Condition is a conditional expression.

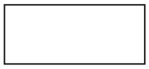
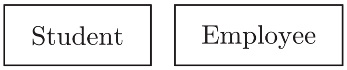
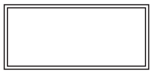
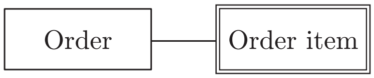
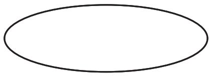
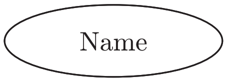
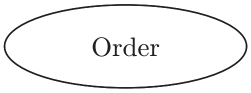
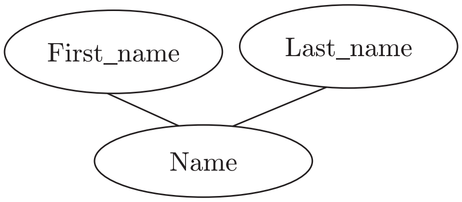
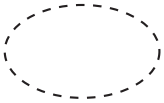
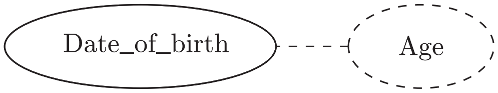
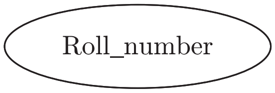
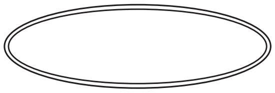
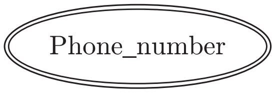
Preliminaries of tuple calculus are as follows:

1. Constants
2. Predicates
3. Boolean and, or, not
4. $\exists$ there exists
5. $\forall$ for all

8.4.2 ER Model

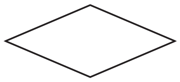
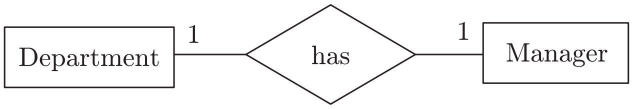
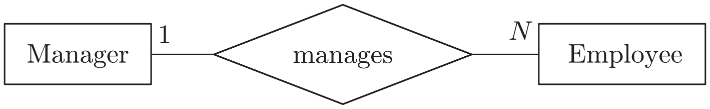
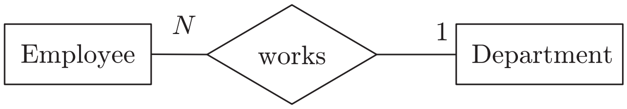
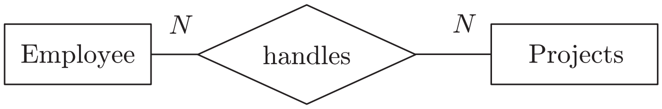
ER model represents the conceptual view of a database. It describes the relation of data to each other (Table 8.2). It was developed by Peter Chen in 1976. It views real-world data as systems of entities and relationships. ER model has three basic elements: entity, attribute and relationship. These are discussed in the following sections.

Table 8.2 Entity-relationship objects

Element	Symbol	Example
Entity		
Weak entity		
Attribute		
Simple attribute		
Composite attribute		
Derived attribute		
Single-valued attribute		
Multi-value attribute		

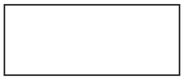
(*Continued*)

Table 8.2 Continued

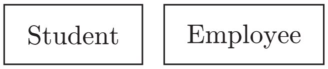
Element	Symbol	Example
Relationship		
One-to-One relationship		
One-to-Many relationship		
Many-to-One relationship		
Many-to-Many relationship		

8.4.2.1 Entity

Entities represent the real-world things. These are data objects which maintain different relationships with each other, for example, Employee, Department, etc. These are represented by means of rectangles.

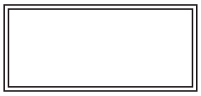


For example,

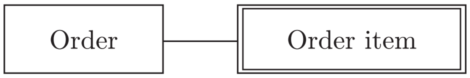


Entity set is a collection of similar types of entities. For example, all Employees set may contain all employees of all departments.

Weak entity depends on the existence of another entity. A weak entity cannot be identified by its own attributes. It uses a foreign key combined with its attributes to form the primary key. It is represented by means of double rectangles.

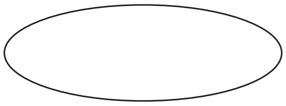


For example, Details of Employee's spouse, order item, etc.

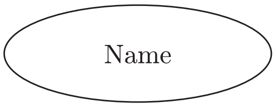


8.4.2.2 Attributes

Attribute is a property, trait or characteristic of an entity, relationship or another attribute. All attributes have values. A domain or range of values can be assigned to attributes. For example, name, class, age are attributes of the entity student. These are represented by ovals.



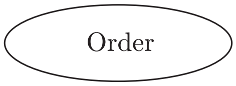
For example,



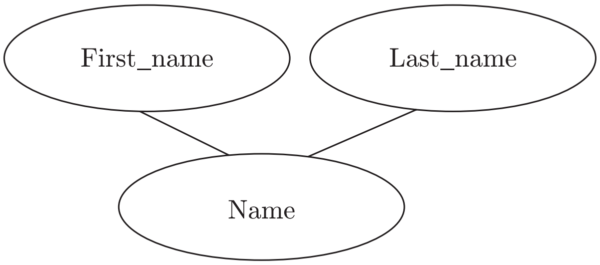
There are different types of attributes, listed as follows:

1. **Simple attribute:** Simple attributes contains atomic values. Atomic values cannot be divided

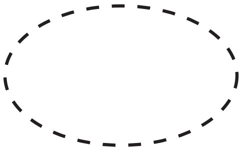
into sub parts. Examples are mobile number, roll number.



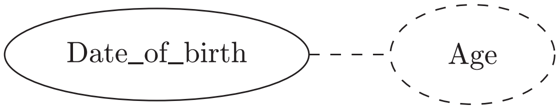
2. **Composite attribute:** Composite attributes are composition of many simple attributes. For example address can be divided into house number, street number, locality and city.



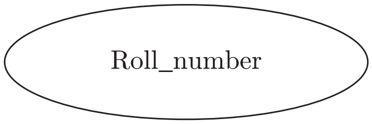
3. **Derived attribute:** Derived attributes are those whose value is derived from some other attribute in the database. For example, age of person can be calculated from date of birth. Dotted oval is used to represent derived attributes.



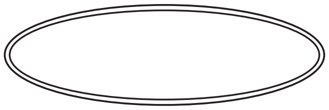
For example,



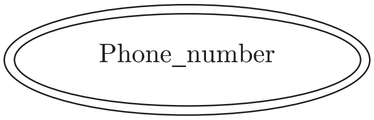
4. **Single-valued attribute:** Single valued attributes contain only one value for that attribute. For example age for person, blood group.



5. **Multi-value attribute:** In this, an attribute may contain more than one value. Multiple values are represented by double ovals.

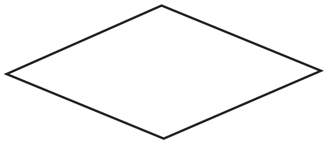


For example contact number, email ids.



8.4.2.3 Relationships

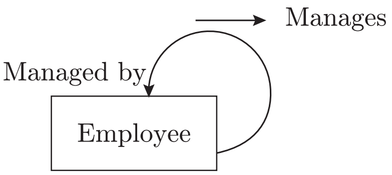
It represents the association among entities in a specified way. For example, employee entity has relation works at with department entity. Relationships are represented by diamond-shaped boxes.



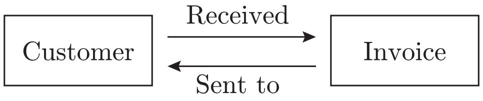
Some basic terminologies related to relationship are given below.

As we have discussed, relations are the core components in RDBMS, these relations are defined by two major characteristics—relationship set and the degree of relationship, defined in the following text.

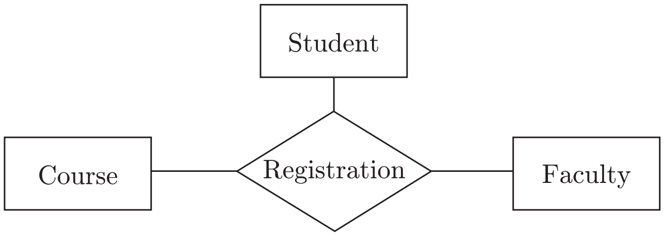
1. **Relationship set:** Relationship of similar type is called relationship set. It has attributes. These attributes are called descriptive attributes.
2. **Degree of relationship:** It defines the number of participating entities in a relationship. They are of the following types:
   - *Unary relationship (Degree 1):* One entity participates. For example,



   - *Binary relationship (Degree 2):* Two entities participate. For example,

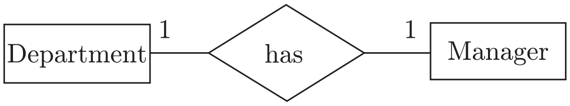


   - *Ternary relationship (Degree 3):* Three entities participate. For example,

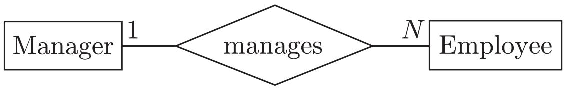


- *n-ary relationship (Degree n): n* entities participate.

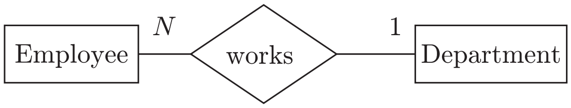
3. **Cardinality:** It defines the number of instances of an entity, which can be associated to the number of instances of other entity via relationship set.
   - *One to one:* One instance of an entity is associated with at most one instance of another entity with the relationship. It is represented as '1-1'. For example,



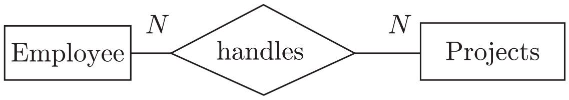
   - *One to many:* One instance of an entity is associated with more than one instance of another entity with the relationship. It is represented as '1-*N*'. For example,



   - *Many to one:* More than one instance of an entity is associated with another entity with the relationship. It is represented as '*N*-1'. For example,



   - *Many to many:* More than one instance of an entity is associated with more than one instance of another entity with the relationship. It is represented as '*N*-*N*'. For example,



4. **Generalization:** It is a collection of entity sets, having similar characteristics, brought together into one generalized entity. For example, salaried and contract employees are generalized as employee.
5. **Specialization:** It is the process of identifying subsets of an entity set. It is a reverse process of generalization. For example, employees are specialized as salaried and contract as shown in Fig 8.3.

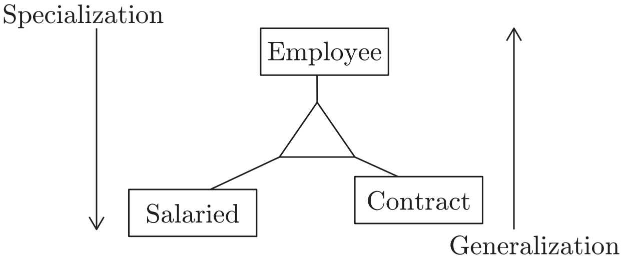


Figure 8.3 Specialization and Generalization.

6. **Aggregation:** It allows a relationship set participate in another relationship set (see Fig. 8.4).

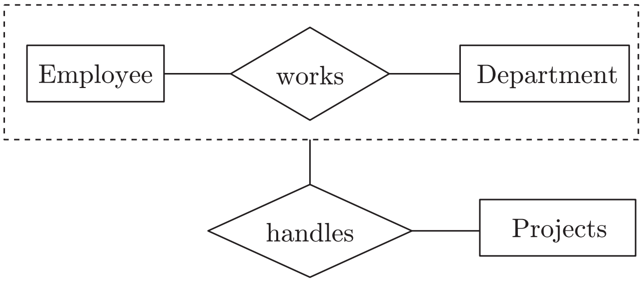


Figure 8.4 Aggregation.

8.5 DATABASE DESIGN

The design of a database consists of the following steps:

1. Identifying entities
2. Identifying relationships
3. Identifying attributes
4. Presenting entities and relationships: Entity relationship diagram (ERD)
5. Assigning keys
6. Defining the attribute's data type
7. Normalization

8.5.1 Integrity Constraints

Integrity constrains are applied to maintain the consistency in a database. This helps in providing the unique answer to a given query on the database. For example, if the answer to particular query is 'x' then it should be 'x' if such a query is carried out again (without adding/deleting/modifying) on the same table.

1. **Domain integrity:** It defines a valid set of values for an attribute, for example, length or size, data type, etc.

2. **Entity integrity constraint:** It defines that the primary keys cannot be null. There must be a proper value in the primary key field.
3. **Referential integrity constraint:** It is specified between two tables. It is used to maintain the consistency among rows between the two tables.
4. **Foreign key integrity constraint:** There are two types of foreign key integrity constraints:
   - *Cascade update related fields:* Whenever the primary key of a row in the primary table is changed, the foreign key values are updated in the matching rows in the related table.
   - *Cascade delete related rows:* Whenever a row in the primary table has been deleted, the matching rows are automatically deleted in the related table.

8.5.2 Normal Forms

Normalization is a technique of organizing the database tables, such that they have minimum redundancy. Following are the normal forms that are to be achieved for the process of normalization. The underlying concept is that, if we say a table to be satisfying an 'x' level normal form, then it is understood that it has also satisfied the 'x-1' level normal form.

1. **First normal form:** It defines that all the attributes in a relation must have atomic domains.
2. **Second normal form:** It defines that every non-prime attribute should be fully functionally dependent on prime key attribute. A prime attribute is a part of prime key. The relation must be in the First normal form.
3. **Third normal form:** It defines that no non-prime attribute is transitively dependent on prime key attribute. The relation must be in the Second normal form.
4. **Boyce–Codd normal form:** It defines that for any non-trivial FD, $X \rightarrow A$, then X must be a superkey. It is an extension of the Third normal form.
5. **Fourth normal form:** It defines that for every multivalued dependency $X \rightarrow\rightarrow Y$ that holds over R, one of the following statements is true: (a) Y is the subset of X and (b) X is a superkey. Also, sometimes called, Multi-valued Dependency Normal Form (MDNF).
6. **Fifth normal form:** It denies that for every join dependency $\bowtie \{R_1, \ldots, R_n\}$ that holds over R, one of the following statements is true: (a) $R_i = R$ and (b) the join dependency is implied by the set of those FDs over R in which the left side is a key. Also, sometimes called, Project-Join Normal Form (PJNF).

8.5.3 Attribute Closure

Set of all attributes functionally determined by X is called closure of X. Closure set of X is denoted by X^+.

Problem 8.1: Functional dependencies $F = \{A \rightarrow B, B \rightarrow C, C \rightarrow D\}$ is given, find closure of A, B, C and D.

Solution:

Closure of $A = A^+ = (A, B, C, D)$

Closure of $B = B^+ = (B, C, D)$

Closure of $C = C^+ = (C, D)$

Closure of $D = D^+ = (D)$

8.5.4 Key

It is a minimum set of attributes used to differentiate all the tuples of the table.

8.5.4.1 Superkey

It is a set of attributes that uniquely identifies each record in a table. It is a superset of candidate key. In other words, let R be the schema and X be the set of attributes over R. If closure of X (X^+) determines all the attributes of R, then X is called superkey.

Problem 8.2: Consider a schema $R(ABCDE)$ and FDs $\{AB \rightarrow C, C \rightarrow D, B \rightarrow E\}$. Find superkey.

Solution:

Find closure set of (A, B, C, AB).

Closure of $A = A^+ = \{A\}$

Closure of $B = B^+ = \{B, E\}$

Closure of $C = C^+ = \{C, D\}$

Closure of $AB = AB^+ = \{A, B, C, D, E\}$

So, AB is the superkey.

8.5.4.2 Candidate Key

It is an attribute or set of attribute that can act as a primary key of a table that uniquely identifies each record in a table. Every candidate key is a superkey but not vice versa.

In other words, candidate key is the minimal superkey. If X is a superkey and none of the proper subset of X is a superkey, then X is called the minimal superkey or candidate key.

Problem 8.3: Consider a schema $R(ABCDE)$ and FDs $\{AB \rightarrow C,\ C \rightarrow D,\ B \rightarrow EA\}$. Find the superkey.

Solution:

Find closure set of (A, B, C, AB)

Closure of $A = A^+ = \{A\}$

Closure of $B = B^+ = \{B, E, A, C, D\}$

Closure of $C = C^+ = \{C, D\}$

Closure of $AB = AB^+ = \{A, B, C, D, E\}$

So, AB and B are superkeys. But only B is the candidate key.

8.5.4.3 *Primary Key*

It is one or more data attributes that uniquely identify an entity. It does not allow null values.

1. **Alternate key:** The candidate key, which is not selected as a primary key.
2. **Composite key:** It consists of two or more attributes.
3. **Foreign key:** It is an entity that is the reference to the primary key of another entity.

8.5.5 Decomposition

It is required to eliminate redundancy from the schema. If a relational schema R has redundancy in the data, then decompose R into two R_1 and R_2 schema. There are two properties, which should be maintained when we perform decomposition, it should be a lossless join as well as dependency preserving.

8.5.5.1 *Lossless Join*

Let R be a relation schema and let F be a set of functional dependency (FD) over R. R is decomposed into R_1 and R_2. R_1 and R_2 are called lossless-join decomposition if $R = R_1 \bowtie R_2$ or if we can recover original relation from the decomposed relation.

(a) *Algorithm for finding decomposition is lossless:*

Step 1: Union of all decomposed sub-relation should be equal to relation R.

$$R_1 \cup R_2 \cup R_3 \cup \ldots \cup R_n = R$$

Step 2: Any two sub-relations R_i and R_j can be merged into R_{ij} with $R_1 \cup R_2$ only if

i. $R_i \cap R_j \neq \varnothing$ (null)

ii. $R_i \cap R_j = X$ then closure of $X(X^+) \supseteq R_i$

or

$R_i \cap R_j = X$ then closure of $X(X^+) \supseteq R_j$

Step 3: Repeat step 2 until 'N' relations become one relation. If 'N' relations become single relation, then composition is called lossless, otherwise not.

Problem 8.4: Consider a schema R $(ABCDEFGHIJ)$ and functional dependencies {FDs = $(AB \rightarrow C,\ A \rightarrow D,\ B \rightarrow F,\ F \rightarrow GH,\ D \rightarrow IJ)$} and decompositions

(a) $\{D = (ABCDE, BFGH, DIJ)\}$
(b) $\{D = (ABCD, DE, BF, FGH, DIJ)\}$

Check whether the decomposition is lossless or not.

Solution:

(a) Given

$$R_1 = (ABCDE)$$
$$R_2 = (BFGH)$$
$$R_3 = (DIJ)$$

Apply algorithm for lossless decomposition:

Step 1:

$$R_1 \cup R_2 \cup R_3 = \{(ABCDE) \cup (BFGH) \cup (DIJ)\}$$
$$= (ABCDEFGHIJ) = R$$

Step 1 satisfies the given condition, so it is true.

Step 2:

For R_1 and R_2:

$$R_1 \cap R_2 = (ABCDE) \cap (BFGH) = B$$

Find closure of $B = B^+ = \{B, F, G, H\}$.

Condition (ii) is satisfied, so R_1 and R_2 can be merged together. After merging R_1 and R_2,

$$R_{12} = (ABCDEFGH)$$

Now merge R_{12} and R_3.

$$R_{12} \cap R_3 = (ABCDEFGH) \cap (DIJ) = D$$

Find closure of $D = D^+ = \{D, I, J\}$.

Condition (ii) is satisfied, so R_{12} and R_3 can be merged together. After merging R_{12} and R_3

$$R_{123} = (ABCDEFGHIJ)$$

So, $R_{123} = R$, therefore, it is lossless decomposition.

(b) Given that

$$R_1 = (ABCD)$$
$$R_2 = (DE)$$
$$R_3 = (BF)$$
$$R_4 = (FGH)$$
$$R_5 = (DIJ)$$

Apply algorithm for lossless decomposition.

Step 1:

$$R_1 \cup R_2 \cup R_3 \cup R_4 \cup R_5 = \{(ABCD) \cup (DE) \cup (BF) \cup (FGH) \cup (DIJ)\} = (ABCDEFGHIJ) = R$$

Step 1 satisfies the given condition, so it is true.

Step 2:

For R_1 and R_2

$$R_1 \cap R_2 = (ABCD) \cap (DE) = D$$

Find closure of $D = D^+ = \{D, I, J\}$.

Condition (ii) of step 2 does not satisfy, so R_1 and R_2 cannot be merged together.

For R_1 and R_3

$$R_1 \cap R_3 = (ABCD) \cap (BF) = B$$

Find closure of $B = B^+ = \{B, F, G, H\}$.

Condition (ii) of step 2 is satisfied, so R_1 and R_3 can be merged together. After merging R_1 and R_3

$$R_{13} = (ABCDF)$$

For R_{13} and R_4

$$R_{13} \cap R_4 = (ABCDF) \cap (FGH) = F$$

Find closure of $F = F^+ = \{F, G, H\}$

Condition (ii) of step 2 is satisfied, so R_{13} and R_4 can be merged together. After merging R_{13} and R_4

$$R_{134} = (ABCDFGH)$$

For R_{134} and R_5

$$R_{134} \cap R_5 = (ABCDFGH) \cap (DIJ) = D$$

Find closure of $D = D^+ = \{D, I, J\}$

Condition (ii) of step 2 is satisfied, so R_{134} and R_5 can be merged together. After merging R_{134} and R_5

$$R_{1345} = (ABCDFGHIJ)$$

For R_{1345} and R_2

$$R_{1345} \cap R_2 = (ABCDFGHIJ) \cap (DE) = D$$

Find closure of $D = D^+ = \{D, I, J\}$.

Condition (ii) of step 2 does not satisfy, so R_{1345} and R_2 cannot be merged together.

So, finally we left with two decompositions which cannot be merged into a single relation. So condition given in step 3 does not satisfy, therefore, it is not lossless decomposition.

8.5.5.2 Dependency Preserving

All the dependency should be preserved after the decomposition of schema R.

Let R be the relational schema with FD set F decomposed into $R_1, R_2, R_3, \ldots, R_n$ with FD sets $F_1, F_2, F_3, \ldots, F_n$, respectively. The decomposition is said to be dependency preserved if

$$F_1 \cup F_2 \cup F_3 \cup \cdots \cup F_n = F$$

And composition is called non-dependency preserved if

$$F_1 \cup F_2 \cup F_3 \cup \cdots \cup F_n \subset F$$

(a) *Algorithm to check dependency is preserved or not:*

Step 1: Find all the FDs of sub-relations.

Step 2: Check that all the FDs of relation F is covered by FDs of sub-relation in any form directly or indirectly.

Problem 8.5: Consider a schema $R(ABCD)$ and FDs set $F = (A \to B, B \to C, C \to D, D \to A)$ and decompositions $D = (AB, BC, CD)$. Check decomposition is dependency preserving or not.

Solution:

Step 1: Find all the FDs of sub-relations.

	$R_1 = (AB)$	$R_2 = (BC)$	$R_3 = (CD)$
Direct Dependency	$A \to B$	$B \to C$	$C \to D$
Indirect Dependency	$B \to A$	$C \to B$	$D \to C$

To calculate indirect dependency, use closure property. In relation R_1, we have indirect dependency $B \rightarrow A$. We have to first find closure of all attributes in that decomposition (like in R_1 we have A and B) by using FD set F and then check that we are getting attribute A in closure of B. So closure of $B = B^+ = ABCD$. So we can write that indirectly as we have three dependencies $\{B \rightarrow A, B \rightarrow C$ and $B \rightarrow D\}$ but only $B \rightarrow A$ is valid for R_1 as we have only two attributes (A and B) in relation R_1.

Step 2: Check that all the FDs of relation F is covered by FDs of sub-relation in any form directly or indirectly.

(a) $A \rightarrow B$ (Directly covered by sub-relation FD)
(b) $B \rightarrow C$ (Directly covered by sub-relation FD)
(c) $C \rightarrow D$ (Directly covered by sub-relation FD)
(d) $D \rightarrow A$ (Indirectly covered by sub-relation FD by using $\{D \rightarrow C$ then $C \rightarrow B$ and then $B \rightarrow A\}$).

All the dependency is preserved, so it is dependency-preserved decomposition.

8.5.5.3 Relation between Two Functional Dependency (FD) Sets

Let F and G be two FD sets:

1. Set F and G are equal if and only if closure of $F(F^+)$ = Closure of $G(G^+)$
2. Set F and G are equal only if both of the below conditions satisfy:
   - F covers G: All FD in G is logically implied by F.
   - G covers F: All FD in F is logically implied by G.
3. If all FD in G is logically implied by F but all FD in F is not logically implied by G then $G \subset F$.

Problem 8.6: Let there be two FD sets F and G, given as follows:

$$F = \{A \rightarrow B, B \rightarrow C\}$$

$$G = \{A \rightarrow B, AB \rightarrow C, B \rightarrow C, A \rightarrow C\}$$

Find the relation between both FDs.

Solution:

(a) F covers G:

Functional Dependency	Covered by F
$A \rightarrow B$	Direct covered
$AB \rightarrow C$	Indirect covered ($AB^+ = ABC$), so AB can determine C.
$B \rightarrow C$	Direct covered
$A \rightarrow C$	Indirect covered ($A^+ = ABC$), so A can determine B and C.

(b) G covers F

Functional Dependency	Covered by G
$A \rightarrow B$	Direct covered
$B \rightarrow C$	Direct covered

In our example, F covers G and vice versa. So $F = G$.

8.6 QUERY LANGUAGES (SQL)

SQL stands for Structured Query Languages, which is a standard computer language for relational database management and data manipulation. It is used to query, insert, update and modify data in the table. Some common RDBMS that use SQL are Oracle, Microsoft SQL Server, Access, Ingres, Sybase, etc. Raymond Boyce and Donald Chamberlin developed it in the early 1970s at IBM, but commercially released by Relational Software Inc. (now, Oracle Corporation) in 1979. Looking into the history it was the software named- VULCAN, that was procured by Ashton Tate and then by FoxPro and later was purchased by Microsoft. Other popular softwares were/are – Clipper and Gupta Technologies.

8.6.1 SQL Commands

The SQL commands are used to interact with relational databases. These commands can be classified into the following groups (see Tables 8.3 and 8.4):

1. **Data Definition Language (DDL):** It contains metadata, that is, data about data. All the integrity constraints and data base schemas are defined through DDL. These commands are used to create, modify and delete database objects. CREATE, ALTER, TRUNCATE and DROP are part of DDL. TRUNCATE command is used to delete complete data from an existing table, while DROP command is used to remove a table definition and all data, indexes, triggers and constraints for a table.
   - Syntax of create command

```
CREATE TABLE table_name( column1
datatype, column2 datatype, column3
datatype, ..... columnN datatype,
PRIMARY KEY( one or more columns ) );
```

For example, `create table instructor (INST_ID char(5), name varchar(20), dept_name varchar(20), salary numeric(8,2));`

- Syntax of alter command

```
ALTER TABLE table_name ADD column_
name datatype;
ALTER TABLE table_name DROP COLUMN
column_name;
ALTER TABLE table_name MODIFY COLUMN
column_name datatype;
ALTER TABLE table_name ADD CONSTRAINT
Constraint UNI (column1, column2...);
ALTER TABLE table_name ADD CONSTRAINT
Constraint CHECK (CONDITION);
ALTER TABLE table_name ADD CONSTRAINT
PrimaryKey PRIMARY KEY (column1,
column2...);
```

For example, `ALTER TABLE CUSTOMERS ADD GRADE char(1);`

- Syntax of truncate command

```
TRUNCATE TABLE table_name;
```

For example, `TRUNCATE TABLE emp;`

- Syntax of drop command

```
DROP TABLE table_name;
```

For example, `DROP TABLE emp;`

Table 8.3 | SQL commands

Command	Description
CREATE	Creates a new table, a view of a table or other objects in the database
ALTER	Modifies an existing database (table)
DROP	Deletes a table, a view of a table or other objects in the database
SELECT	Retrieves data from one or more tables (database)
INSERT	Inserts new data record in the database
UPDATE	Modifies data in the database
DELETE	Deletes data from the database
GRANT	Gives a privilege to the user
REVOKE	Takes back privileges granted from the user

Table 8.4 | Clauses in SQL

Clause	Description
From	Equals to cross product
Where	Selects the tuples which satisfies the condition
Group by	Groups the table based on specified attribute
Having	Used to select groups based on condition

2. **Data Manipulation Language (DML):** DML is used to manipulate the data in database. It allows to insert, update and delete data items. SELECT, INSERT, DELETE and UPDATE are part of DML. Control statements BEGIN TRANSACTION, SAVEPOINT, COMMIT and ROLLBACK are also part of DML.

- Syntax of select command

```
SELECT * FROM table_name;
SELECT column1, column2, columnN
FROM table_name;
```

For example, `SELECT empno, ename FROM emp;`

- Syntax of insert command

```
INSERT INTO TABLE_NAME (column1,
column2, column3,...columnN)] VALUES
(value1, value2, value3,...valueN);
INSERT INTO TABLE_NAME VALUES
(value1,value2,value3,...valueN);
```

For example, `INSERT into emp (empno, ename, sal, dept) VALUES (100, ABC, 10000, Accounts);`

- Syntax of delete command

```
DELETE FROM table_name WHERE [condi-
tion];
```

For example, `DELETE FROM emp WHERE empno = 8;`

- Syntax of update command

```
UPDATE table_name SET column1 =
value1, column2 = value2....,
columnN = valueN WHERE [condition];
```

For example, `UPDATE mp SET sal = 12000 WHERE empid = 8;`

3. **Data Control Language (DCL):** To assign or revoke access rights data control language is used. GRANT and REVOKE are used for DCL.

- Syntax of grant command

  ```
  GRANT privilege_name ON object_name
  TO {user_name | PUBLIC | role_name}
  [with GRANT option];
  ```

  For example, `GRANT SELECT ON emp TO user1;`

- Syntax of revoke command

  ```
  REVOKE privilege_name ON object_name
  FROM {User_name | PUBLIC | Role_name};
  ```

  For example, `REVOKE SELECT ON emp TO user1;`

8.7 FILE STRUCTURES

File structure mainly deals with how files are stored on the disk. Various file organisations are described below.

8.7.1 Sequential Files

It is a file organisation system where every file record contains an attribute to uniquely identify a particular record. Records are placed in a sequential order with some unique key.

8.7.2 Indexing

Indexing is a data structure mechanism to efficiently retrieve records from the database, for example, book index. It is defined based on its indexing attributes. Indexing is of three types:

1. **Primary index:** Indexing is based on ordering key field of file.
2. **Secondary index:** Indexing is based on non-ordering field of file.
3. **Clustering index:** Indexing is based on ordering non-key field of file.

Ordering field is the field on which the records of file are ordered. Ordered indexing is of two types: dense index and sparse index.

1. **Dense index:** For every search key value in the database, there is an index record. Index record has two parts: search key value and the pointer. The pointer is pointed to the actual record.
2. **Sparse index:** In this no index record is created for every search key.

8.7.3 B Tree

A B tree is a data structure that stores data in such a manner that search, insertions and deletions can be done in logarithmic time. B trees are a general form of binary trees where a node can have more than one child.The B-trees are efficient for those systems that read and write large blocks of data, that is, databases and file systems.

B trees are self-balancing trees. All the leaf nodes are at the same level. A B tree with order p has:

1. Root node may contain minimum 1 key
2. Minimum number of child = $\left(\frac{p}{2}\right) - 1$
3. Maximum number of children = p
4. Maximum keys = $p - 1$

The order of B-tree can be found as follows:

$$p \times P + (p-1)(K + P_r) \leq \text{ Block size}$$

where p is order of the tree, P is the block pointer, P_r is the record pointer and K is the key pointer.

Problem 8.7: The order of B-tree index is the maximum number of children it can have. Suppose that a block pointer takes 6 bytes, the search field value takes 9 bytes, record pointer takes 7 bytes and the block size is 512 bytes. What is the order of the B tree?

Solution:

Given that block size = 512 B; record pointer (P_r) = 7 B; block pointer (P_B) = 6 B; key pointer (K) = 9 B. So, we have

$$p \times P + (p-1)(K + P_r) \leq \text{ Block size}$$

$$6p + (p - 1)(9 + 7) = 512$$

$$p = 24$$

8.7.4 B+ Trees

It supports multilevel indexing. The leaf nodes of B+ tree represent actual data pointers. It ensures all leaf nodes are balanced, that is, at the same height. Leaf nodes are linked using link list.

B+ tree structure is such that B+ tree is of order n and it is fixed for every B+ tree.

The internal nodes contain at least $[n/2]$ pointers, except the root node and at most n pointers. Leaf nodes contain at least $[n/2]$ record pointers, at least $[n/2]$ key values, at most n record pointers, at most n key values, and every leaf node contains one block pointer P to point to next leaf node and forms a linked list.

Order of internal nodes can be calculated as:

$$p \times P_B + (p-1) \times k \leq \text{Block size}$$

Order of leaf nodes:

$$P_{\text{leaf}} \times [k + P_r] + P_B \leq \text{Block size}$$

where p is the order of internal nodes, P_{r} is record pointer, P_{B} is block pointer, k is key pointer, P_{leaf} is the order of leaf nodes.

Problem 8.8: The order of an internal node in a B+ tree index is the maximum number of children it can have. Suppose that a block pointer takes 6 bytes, the search field value takes 9 bytes, record pointer take 7 bytes and the block size is 512 bytes. What is the order of the internal node and leaf node?

Solution:

Given that block size = 512 B, record pointer (P_{r}) = 7 B, block pointer (P_{B}) = 6 B and key pointer (k) = 9 B.

Order of internal nodes:

$$p \times P_B + (p-1)\ k \leq \text{Block size}$$
$$6p + (p - 1)9 = 512$$
$$p = 34$$

Order of leaf nodes:

$$P_{\text{leaf}} \times \left[k + P_r\right] + P_B \leq \text{Block size}$$
$$P_{\text{leaf}}\ [9 + 7] + 6 = 512$$
$$P_{\text{leaf}} = 31$$

8.8 TRANSACTIONS AND CONCURRENCY CONTROL

Transactions are series of read and write operations on database. When many users access same database at the same time, some control mechanism is required by databases to stay in consistent state. Following sections will provide description about transactions and their control mechanism.

8.8.1 Transactions

It is a series of reads and writes of database objects. It maintains the integrity of a database while running multiple concurrent operations. Transactions have four properties that a DBMS must ensure to maintain data. These are known as ACID properties:

1. **Atomicity:** Either all actions are carried out or none (no partial transaction).
2. **Consistency:** After the execution of transaction, the database must be in a consistent state (no concurrent execution of other transactions).
3. **Isolation:** All the transactions are carried out and executed as the only transaction in the system (no transaction will affect the existence of any other transaction).
4. **Durability:** Persistence of data even if the system fails and restarts.

8.8.2 Schedule

A chronological execution sequence of transactions is called schedule. It is a list of actions reading, writing, aborting or committing from a set of transactions. A schedule can have many transactions. Schedule can be further divided into two types—serial schedule and concurrent schedule.

8.8.2.1 Serial Schedule

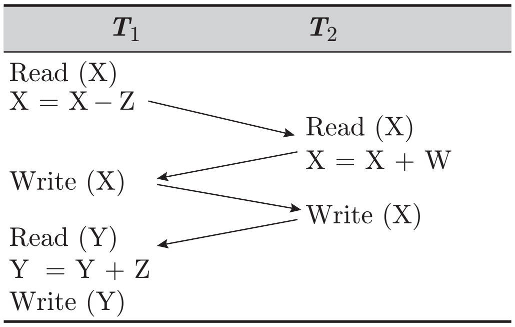
A schedule is called serial schedule if all transactions are executed in a non-interleaving manner. Let there are two transactions T_1 and T_2 schedule S then schedule S is called serial schedule if one transaction executes after another shown in Fig. 8.5.

T_1	T_2	T_1	T_2
Read (X)			Read (X)
$X = X - Z$			$X = X + W$
Write (X)			Write (X)
Read (Y)		Read (X)	
$Y = Y + Z$		$X = X - Z$	
Write (Y)		Write (X)	
	Read (X)	Read (Y)	
	$X = X + W$	$Y = Y + Z$	
	Write (X)	Write (Y)	

Figure 8.5 Serial schedule.

8.8.2.2 Concurrent Schedule

A schedule is called concurrent schedule if transactions are executed in an interleaving manner or simultaneous execution of two or more transactions. Let there be two transactions T_1 and T_2 in a schedule S, then schedule S is called concurrent schedule if both the transactions execute parallel. One of the scenarios is shown in Fig. 8.6.



T_1	T_2
Read (X)	
X = X – Z	
	Read (X)
	X = X + W
Write (X)	
	Write (X)
Read (Y)	
Y = Y + Z	
Write (Y)	

Figure 8.6 Concurrent schedule.

8.8.2.3 Comparison between Serial and Concurrent Schedule

Table 8.5 shows the comparison between serial and concurrent schedule.

Table 8.5 Serial schedule vs. concurrent schedule

Serial Schedule	Concurrent Schedule
All serial schedules are consistent schedules	All serial schedules are not consistent schedules
Less throughput	More throughput
Poor resource utilization	Good resource utilization
More response time	Less response time
For the given transactions, the number of serial schedule is very much less than the number of concurrent schedule	For the given transactions, the number of concurrent schedule is more than the number of serial schedule

8.8.2.4 Problems Occurring due to Concurrent Schedule

In concurrent schedule, more than one transaction is executed simultaneously; and due to this some problem arises with concurrent schedule given as follows:

1. **WR problem (Read after Write):** WR problem is also known as dirty read problem or uncommitted read problem. Let there be two transactions T_i and T_j of schedule S. If transaction T_j reads a data item which is written by T_i, but till that time transaction T_i is not committed, then WR problem can occur. If in transaction T_i rollback or failure occurs, then database will be inconsistent due to uncommitted read by T_j transaction (Fig. 8.7).

Transaction T_i	Transaction T_j
Read (A)	
A = A + X	
Write (A)	Dirty Read
	Read (A)
	Write (B)
	A = A + Y
Read (B)	
......	
......	
Failure or rollback	
	Write (A)

Figure 8.7 Uncommitted read problem (WR problem).

Example: Let value of data item A = 1000, X = 100 and Y = 200, then transaction T_i will update the database with value 1100. Transaction T_j will read data item value as 1100. Transaction T_j will update database with value 1300. As transaction T_i fails, the database value should be 1200. Therefore, this problem is known as uncommitted read problem or dirty read problem.

2. **RW problem (Write after Read):** RW problem is also known incorrect summary problem or unrepeatable read problem (Fig. 8.8). Let there be two transactions T_i and T_j of schedule S. Transaction T_i read a data item and T_j also read similar data item. Transactions T_i and T_j both have write operation on that data item. If in both transactions, read operation occurs before commit of the other transaction and before writing back that data item in database by other transaction then RW problem will arise.

Transaction T_i	Transaction T_j
Read (A)	
If (A > 0)	
A = A + 1	
	Read (A)
	If (A > 0)
	A = A + 1
Write (A)	
Commit	Write (A)
	Commit

Figure 8.8 Incorrect summary problem (RW problem)

Example: Let value of data item A = 100. Transaction T_i will read data item with value 100 and update database with value 101. Transaction T_j will read data item value as 100 because it will read before updating value of data item A by transaction T_i. Transaction T_j will update database with value 101. But data item 'A' value should be 102 because both the transaction have commit successfully. Therefore, this problem is known as unrepeatable read problem.

3. **WW Problem (Write after Write):** WW problem in concurrent schedule is also known as lost update problem (Fig. 8.9). Let there are two transactions T_i and T_j of schedule S. Transaction T_i reads a data item and updates it. Now, transaction T_j also writes similar data item with some other value and transaction T_j commits successfully. After commit of transaction T_j, transaction T_i will rollback. Thus, updated value of data item by transaction T_j will be lost.

Transaction T_i	Transaction T_j
Read (A)	
Write (A)	
	Write (A)
	Commit
Failure and Rollback	

Figure 8.9 Lost update problem (WW problem).

Example: Let the value of data item A=100. Transaction T_i read data item and update database with value 200. Let transaction T_j updated database with value 250 and commit successfully. After committing transaction T_j, transaction T_i rollback which set data item value to its initial value 100. So, this problem is known as lost update problem.

8.8.3 Classification of Schedule Based on Recoverability

8.8.3.1 Irrecoverable Schedule

Let there be two transactions T_i and T_j of schedule S. If transaction T_j reads a data item which is updated by transaction T_i and transaction T_j commits before the commit (or rollback) of transaction T_i, then the given schedule S is called irrecoverable schedule.

8.8.3.2 Recoverable Schedule

Let a schedule S have two transactions T_i and T_j. If transaction T_j reads a data item which is updated by transaction T_i and transaction T_j is not allowed to commit (or rollback) before the commit (or rollback) of transaction T_i, then the given schedule S is called recoverable schedule. Recoverable schedule may suffer from uncommitted read, lost update and incorrect summary problem.

8.8.3.3 Cascading Rollback Recoverable Schedule

Let there be four transactions (T_1, T_2, T_3, T_4) in a given schedule S. In schedule S, if rollback of a transaction (T_1) results in the rollback of the other transactions (T_2, T_3, T_4) (because of their dependency on each other), then this is called cascading rollback.

If a schedule is recoverable and has no cascading rollback, then it is called cascading rollback recoverable schedule. Incorrect summary and lost update problems may exist in cascading rollback recoverable schedule. Problems of no recoverability, Uncommitted Read problem (WR problem) and cascading rollback problem do not occur in such schedule.

8.8.3.4 Strict Recoverable Schedule

Let a schedule S have two transactions T_i and T_j. If S is called a strict recoverable schedule then it satisfies these two conditions:

1. Schedule S should be cascading rollback recoverable schedule.
2. If one transaction T_i writes a data item 'A', then the other transaction T_j is not allowed to write on that data item.

Only incorrect summary problem (RW problem) may arise in strict recoverable schedule. Irrecoverable, Uncommitted Read problem (WR problem), lost update and cascading rollback problems do not occur in such schedule.

8.8.3.5 Summary of Schedules Based on Recoverability

Schedule	Problem Exists	Problem Remove
Irrecoverable	All	None
Recoverable	Incorrect summary problem (RW), Uncommitted Read problem (WR problem), lost update (WW) and cascading rollback problem	Irrecoverable
Cascading rollback recoverable	Incorrect summary problem (RW) and lost update (WW).	Irrecoverable, Uncommitted Read problem (WR problem) and cascading rollback problem.
Strict recoverable	Incorrect summary problem (RW)	Irrecoverable, Uncommitted Read problem (WR problem), lost update (WW) and cascading rollback problem.

Problem 8.9: A concurrent schedule S has three transactions T_1, T_2, T_3. Transactions execute read/write operation in the following sequence:

Read$_1$(x), Read$_2$(z), Read$_3$(x), Read$_1$(z), Read$_2$(y), Read$_3$(y), Write$_1$(x), Commit$_1$, Write$_2$(z), Write$_3$(y), Write$_2$(y), Commit$_3$, Commit$_2$.

Find out the type of recoverable schedule?

Solution:

Transaction T_1	**Transaction T_2**	**Transaction T_3**
Read$_1$(x)	-----	-----
-----	Read$_2$(z)	-----
-----	-----	Read$_3$(x)
Read$_1$(z)	-----	-----
-----	Read$_2$(y)	-----
-----	-----	$Read_3$(y)
Write$_1$ (x)	-----	-----
Commit$_1$	-----	-----
-----	Write$_2$(z)	-----
-----	-----	Write$_3$(y)
-----	Write$_2$ (y)	-----
-----	-----	Commit$_3$
-----	Commit$_2$	-----

Step 1: Check for recoverable. By using the definition of recoverable schedule we can find that given schedule is recoverable or not. (In this problem, no transaction has read operation on a data item after that data item is written by another transaction, so the given schedule is recoverable.)

Step 2: Check for cascading recoverable. In cascading recoverable schedule, no uncommitted read is allowed. In the given problem, there is no uncommitted read so the given schedule is cascadless schedule.

Step 3: Check for strict recoverable. There should be commit operation between write operations on similar data items by two different transactions. In the given problem, transaction T_2 and transaction T_3 have write operations on data item (y), but no commit operation is performed by transaction T_3 before data item is updated by transaction T_2. Hence, the given schedule is not strict recoverable.

8.8.4 Classification of Schedule Based on Serializability

A concurrent transaction schedule is said to be a serializable schedule if its outcome (resulting database state) is equal to the outcome of the serial execution of transactions in the given concurrent schedule. Serializable schedule can be divided into two types: conflict serializability and view serializability.

8.8.4.1 Conflict Serializability

A concurrent schedule S is said to be conflict serializable schedule if S is conflict equivalent to a serial schedule.

1. **Conflict equivalent schedule:** Let there be two schedules S_1 and S_2. If after swapping consecutive non-conflict pairs of operation of transactions in schedule S_1 results in schedule S_2, then S_1 and S_2 schedules are called conflict equivalent schedules.
2. **Conflict and non-conflict pairs:** A pair of operations in a schedule is called conflict schedule if they have all three conditions:
   - At least one of the operations should be write operation.
   - Both the operations should be performed on the same data item.
   - Both operations should be performed by different transactions.

If a pair of operations do not fulfill the above three conditions, then the pair is call a non-conflict pair.

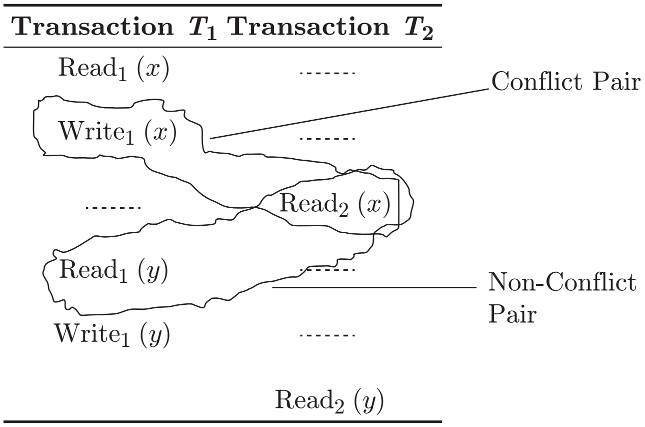
Problem 8.10: A schedule S_1 with two transactions T_1 and T_2 is given as:

Read$_1$(x), Write$_1$(x), Read$_2$(x), Read$_1$(y), Write$_1$(y), Read$_2$(y),....

Find conflict equivalent schedule to S_1 .

Solution:

Schedule S_1:



Transaction T_1	Transaction T_2
$\text{Read}_1\ (x)$	-------
$\text{Write}_1\ (x)$	-------
-------	$\text{Read}_2\ (x)$
$\text{Read}_1\ (y)$	-------
$\text{Write}_1\ (y)$	-------
	$\text{Read}_2\ (y)$

Conflict and non-conflict pairs in S_1.

After swapping non-conflict pair in schedule S_1 we get another schedule called S_2. The schedules S_1 and S_2 are conflict equivalent schedules.

Schedule S_2:

Transaction T_1	Transaction T_2
$\text{Read}_1\ (x)$	------
$\text{Write}_1\ (x)$	------
$\text{Read}_1\ (y)$	------
------	$\text{Read}_2\ (x)$
$\text{Write}_1\ (y)$	------
------	$\text{Read}_2\ (y)$

Conflict equivalent schedules.

3. **Testing method of conflict serializable schedule:** To check that a given schedule is conflict serializable or not, we use precedence graph. If precedence drawn from a schedule does not contain a cycle then only it is a conflict serializable schedule. If precedence graph contains a cycle then it is not a conflict serial schedule. A precedence graph G contains:
   (a) Vertex set V: denotes the transactions of the schedule.
   (b) Edge between two vertex from V_i to V_j will be draw where $i \neq j$ and V_i's operation occur before V_j's operation, if V_i *and* V_j have following cases:

   (i) V_i : Read(x) operation and V_j : Write(x) operation
   (ii) V_i : Write(x) operation and V_j : Read(x) operation
   (iii) V_i : Write(x) operation and V_j : Write(x) operation

Problem 8.11: Consider the given schedule S and draw precedence graph for S.

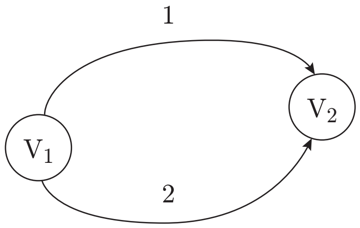
Transaction T_1	Transaction T_2
------	$\text{Read}_1\ (x)$
$\text{Write}_1\ (x)$	------
------	$\text{Read}_2\ (x)$
$\text{Write}_1\ (y)$	------
------	$\text{Read}_2\ (y)$

Solution:

Vertices V_1 and V_2 represent transactions T_1 and T_2.

Edge 1 for
V_2 : Read(x) operation and V_1 : Write(x) operation

Edge 2 for
V_1 : Write(y) operation and V_2 : Read(y) operation



Note: If two schedules S_1 and S_2 are given then they will be conflict equivalent if they satisfy following conditions:

- Precedence graph of S_1 and S_2 should be equal.
- Each transaction operation should be equal in S_1 and S_2. (The two transaction will be equal of they have same number of operations and type of operation are also same.)

Problem 8.12: Consider the given schedule S and identify that schedule S is conflict serial schedule or not by the use of precedence graph.

Transaction T_1	Transaction T_2	Transaction T_3
Read (x)	------	------
Write (x)	------	------
------	Read (x)	------
------	------	Write (x)
------	Write (y)	------
Write (y)	------	------
------	------	Read (y)

Solution:

Vertices V_1, V_2 and V_3 represent transactions T_1, T_2 and T_3.

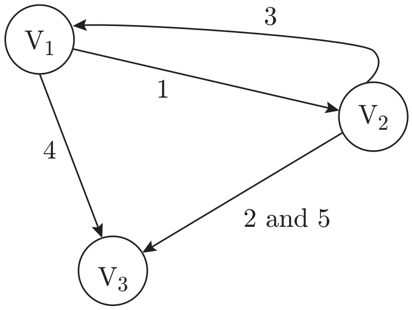
Edge 1 for
$V_1 : \text{Write}(x)\ \text{operation}\,\text{and}\,V_2 : \text{Read}(x)\ \text{operation}$

Edge 2 for
$V_2 : \text{Read}(x)\ \text{operation}\,\text{and}\,V_3 : \text{Write}(x)\ \text{operation}$

Edge 3 for
$V_2 : \text{Write}(y)\ \text{operation}\,\text{and}\,V_1 : \text{Write}(y)\ \text{operation}$

Edge 4 for
$V_1 : \text{Write}(y)\ \text{operation}\,\text{and}\,V_3 : \text{Read}(y)\ \text{operation}$

Edge 5 for
$V_2 : \text{Write}(y)\ \text{operation}\,\text{and}\,V_3 : \text{Read}(y)\ \text{operation}$



Precedence graph drawn from the given schedule contains cycle with vertices V_1 and V_2. So, the given schedule is not a conflict serializable schedule.

8.8.4.2 View Serializability

If a given concurrent schedule is view equivalent to one of the possible serial schedule, then it is called a view serializable schedule. Let there be two schedules S_1 and S_2 with the similar set of transactions. Schedules S_1 and S_2 will be view equivalent if they satisfy following three conditions:

1. If transaction T_i reads the initial value of data item (x) in schedule S_1, then transaction T_i must read the initial value of (x) in schedule S_2 .
2. In schedule S_1 , if transaction T_i reads the value of data item (x) produced by transaction T_j then T_i must reads the value of (x) that produced by transaction T_j in the schedule S_2.
3. If transaction T_i write the final value of data item (x) in schedule S_1, then transaction T_i must write the final value of (x) in schedule S_2.

Problem 8.13: Identify whether the given concurrent schedule is a view serializable schedule or not?

T_1	T_2	T_3
------	------	Read (x)
------	Read (x)	------
------	------	Write (x)
Read (x)	------	------
Write (x)	------	------

Solution: We have a concurrent schedule S_1 with three transactions, so the number of possible serializable schedules with three transactions is $8\,(2^3 = 8)$. So, we will pick one serializable schedule S_2 and if the given concurrent schedule is be view equivalent to that schedule, only then would the given schedule will be view serializable.

T_1	T_2	T_3
------	------	Read (x)
------	Read (x)	------
------	------	Write (x)
Read (x)	------	------
Write (x)	------	------

Schedule S_1

T_2	T_3	T_1
Read (x)		
	Read (x)	
	Write (x)	
		Read (x)
		Write (x)

Schedule S_2

Step 1: Initial Reads. Transactions T_3 and T_2 read initial value of data item (x) in schedule S_1 (will not consider T_1 because it reads updated value of x by T_3). T_3 and T_2 also read the initial value of (x) in serial schedule S_2. So, the first condition is true.

Step 2: W-R Conflicts. In the given schedule S_1, there is only one W-R conflict (between T_3 and T_1). Transaction T_1 reads the value of x produced by transaction T_3 in schedule. In schedule S_2, transaction T_1 also reads the value of x produced by transaction T_3 in schedule. So, the second condition is also true.

Step 3: Final Write. Transaction T_1 writes the final value of data item (x) in schedules S_1 and S_2 both. So, the third condition is also satisfied.

All the three conditions are satisfied, so the given concurrent schedule S_1 is view equivalent to one of the possible serial schedule S_2. Therefore, the given concurrent schedule is a view serializable schedule.

8.8.5 Concurrency Control Protocol

Concurrency control protocols are used to ensure serializability of transactions in a concurrent transaction execution environment. They can be lock based protocols and Time stamp ordering protocols.

8.8.5.1 Lock Based Protocol

Lock based protocols uses a mechanism through which a transaction has to obtain a read lock (for read operation) and write lock (for write operation) on a data item (x) without obtaining a lock transaction not allowed to perform any operation. There are two types of lock mechanisms supported by lock based protocol.

1. **Shared lock:** If a transaction T has shared lock on a data item (x), then it can only perform Read operation on that data item.
2. **Exclusive lock:** If a transaction T has exclusive lock on a data item (x), then it can perform Read or Write any operation on that data item.

Lock Compatibility Matrix is given in Table 8.6, which shows that if a transaction has shared/exclusive lock then for which lock it can make a request.

Table 8.6 | Lock compatibility matrix

Lock Hold **Lock Request**	**Shared Lock**	**Exclusive Lock**
Shared Lock	Yes	No
Exclusive Lock	No	No

1. **Two Phase Locking Protocol (2PL):** Two phase locking protocol is used by the transaction to lock a data item before reading and writing to maintained consistency in the database. The protocol uses two phases to apply the locking scheme as:
   - *Expanding phase:* Transaction acquired locks in this phase and no locks are released.
   - *Shrinking phase:* Transaction releases locks in this phase and no locks are acquired.

   Note: Two phase locking protocol ensure conflict serializability, but may not be free from irrecoverable, deadlock and starvation.
2. **Two Phase Locking Protocol with Lock Up-gradation:** This technique allows transaction holding a shared lock on a data item can upgrade shared lock into the exclusive lock on the data item without performing unlock for shared lock.
3. **Strict Two Phase Locking Protocol:** In this protocol, if a transaction has exclusive lock on a data item, then it cannot release the lock until commit or rollback is performed by that transaction.
4. **Rigorous Two Phase Locking Protocol:** In this protocol, if a transaction has any lock (exclusive or shared) on a data item, then it cannot release the lock until commit or rollback performed by that transaction.

8.8.5.2 Time Stamp Ordering Protocol

The time stamp ordering protocol uses time stamp values (unique) of the transactions assigned by database to ensure serializability among transactions. A time stamp is a unique value assigned by the database in ascending order. Suppose we have three transactions T_1, T_2 and T_3 arriving in database, then assigned time stamp values will be as $T_1 < T_2 < T_3$. A data item has two types of time stamp.

1. **Read Time Stamp (x):** It is the highest transaction time stamp that has performed Read operation successfully on a data item (x). It is also denoted by RTS(x). Initial value of RTS(x) is zero.
2. **Write Time Stamp (x):** It is the highest transaction time stamp that has performed Write operation successfully on a data item (x). It is also denoted by WTS(x). Initial value of WTS(x) is zero.
3. **Basic Time Ordering Protocol**
   - Transaction T_i issues Read(x) operation

     (a) If (WTS(x) > Time Stamp(T_i))

     Rollback T_i

     (b) Otherwise allowed execution of Read(x) by transaction T_i

     Set RTS(x) = Maximum (RTS(x), Time Stamp (T_i))

- Transaction T issues Write(x) operation
  (a) If (RTS(x) > Time Stamp(T_i))

  Rollback T_i

  (b) If (WTS(x) > Time Stamp(T_i))

  Rollback T_i

  Note: Basic Time Ordering protocol ensures serializability, equivalent serial schedule based on time stamp Ordering. It is deadlock free protocol, but basic time ordering protocol can have starvation and irrecoverable schedule (because it does not ensure order of commit).

4. **Strict Time Ordering Protocol:** In this protocol, an additional condition is added in the Basic Time Ordering protocol. This condition is as follows:

   Let a schedule have two transactions T_i and T_j where time stamp of (T_i) is less than time stamp of T_j. If transaction T_j issues Read(x)/ Write(x) operation with WTS(x) < Time Stamp (T_j) then (T_j) has to be delayed Read(x)/ Write(x) operation until commit or rollback of transaction T_i that has performed Write(x).

   Strict Time Ordering Protocol ensures serializability and deadlock free. But it may suffer from starvation.

5. **Deadlock Prevention:** To prevent deadlock situation in a database system it is very important to inspect all the operation performed by transactions in a schedule. There are various deadlock prevention scheme which uses time stamp ordering mechanism.

6. **Wait-Die Protocol:** Assume TimeStamp (T_1) < … < TimeStamp (T_n) and T_i and T_{j} are any two transactions and transaction T_i is older than transaction T_j. This implies

   $$\text{TimeStamp } (T_i) < \ldots < \text{TimeStamp } (T_j).$$

   - If transaction T_i required a lock which is hold by transaction T_j, then transaction T_i is allowed to wait.
   - If transaction T_j required a lock which is hold by transaction T_i then transaction T_j will rollback.

7. **Wound-Wait Protocol:** Let TimeStamp (T_1) < … < TimeStamp (T_n) and T_i and T_j are any two transactions and transaction T_i is older than transaction T_{j}. This implies

   $$\text{TimeStamp } (T_i) < \text{TimeStamp } (T_j)$$

   - If transaction T_{i} required a lock which is hold by transaction T_j then rollback transaction T_j.
   - If transaction T_j required a lock which is hold by transaction T_i then transaction T_j is allowed to wait.

IMPORTANT FORMULAS

1. All the 3NF and BCNF decomposition guarantee lossless join.
2. All 3NF and BCNF decomposition may not guarantee dependency preservation.
3. Any relation with two attributes is in BCNF.
4. Functional dependency F: $X \rightarrow Y$ implies that for any two tuples if $t_1[X] = t_2[X]$, they must have $t_1[Y] = t_2[Y]$.
5. CP: Child Pointer
6. KV: Key Value
7. Order of internal node in B+ tree is
   $P_{\text{I}} \cdot \text{CP} + (P_{\text{I}} - 1)\text{KV} \Leftarrow \text{Block size}$

Inference Rules:

1. **Reflexivity**
   If $Y \subseteq X$, then $X \rightarrow Y$
2. **Augmentation**
   If $X \rightarrow Y$, then $XZ \rightarrow YZ$ for any Z
3. **Transitivity**
   If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
4. **Union**
   If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
5. **Decomposition**
   If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$
6. **Pseudo-transitivity**
   If $A \rightarrow B$ and $BC \rightarrow D$, then $AC \rightarrow D$
7. Number of rows in cross product = $r_1 \times r_2$
8. Number of attributes in cross product = $m + n$
9. Complete set of operations are $\{\sigma, \pi, \cup, -, \times\}$

10. In clustering index, file is ordered on non-key field.

11. Secondary indexes are created on unordered file on either key or non-key field.

12. Different kind of keys – Primary Key, Secondary Key, Candidate Key, Alternate key, Foreign Key, Superkey, Compound key (or Composite key) (or concatenated key).

SOLVED EXAMPLES

1. Which of the following operations does not modify the database?

(a) Sorting (b) Insertion at the beginning
(c) Append (d) Modify

Solution: Sorting does not involving addition/insertion or modification of any element.

Ans. (a)

2. Which of the following operations is based on relational algebra?

(a) rename and union
(b) select and union
(c) only union
(d) rename, select and union

Solution: Rename, select and union are some of the operations of relational algebra.

Ans. (d)

3. ACID properties of a transaction are

(a) atomicity, constant, integrity, durability.
(b) atomicity, consistency, isolation, durability.
(c) atomicity, compact, in-built, durability.
(d) automatically, consistency, isolation, database.

Solution: ACID properties are atomicity, consistency, isolation and durability.

Ans. (b)

4. Which normal form is concerned with removing transitive dependency?

(a) 1NF (b) 2NF
(c) 3NF (d) BCNF

Solution: 3NF is also the highest normal form.

Ans. (c)

5. The concept Database Lock is used to overcome the problem of

(a) lost update and inconsistent data.
(b) uncommitted dependency and inconsistent data.
(c) inconsistent data only.
(d) lost updates, inconsistent data and uncommitted dependency.

Solution: Lost update, inconsistent data and uncommitted (WR) problems are overcome by database lock.

Ans. (d)

6. Database language may consist of the following?

(a) DDL and DML
(b) DML
(c) Query language
(d) DDL, DML, query language

Solution: DDL, DML, query language are all data base languages.

Ans. (d)

7. Which of the following is not a function of DBA?

(a) Network maintenance
(b) Routine maintenance
(c) Defining the schema
(d) Data access through authorization

Solution: The role of database administrator is to manage the DBMS.

Ans. (a)

8. In a relational database the category type of information is represented in

(a) tuple. (b) field.
(c) primary key. (d) database name.

Solution: Field shows type of information in relational database.

Ans. (b)

9. Which key is used to identify a tuple uniquely?

(a) Primary key (b) Tuple key
(c) Unique key (d) Domain key

Solution: Primary key.

Ans. (a)

10. An association of the information is represented by

(a) an attribute. (b) a relationship.
(c) a normal form. (d) the records.

Solution: Relationship shows association of information.

Ans. (b)

11. In which stage of database design, all the necessary fields and their types of a database are listed?

(a) Data definition (b) Data field definition
(c) E-R diagram (d) User definition

Solution: In data definition stage, all the fields and their types are listed.

Ans. (a)

12. The maximum length of an attribute of type text is

(a) 127. (b) 255.
(c) 256. (d) It is a variable.

Solution: 255

Ans. (b)

13. When a transaction has completed all its operations and is waiting for commit or rollback action, the state of transaction is

(a) partially commit. (b) ready commit.
(c) half commit. (d) commit and rollback.

Solution: The state of that transaction is partially commit.

Ans. (a)

14. Which of the following statement is not false?

(a) Time stamp protocols avoid deadlock.
(b) Locking technique is used to avoid deadlock.
(c) 2Phase locking does not provide serializability.
(d) 2Phase deals with input and storing phase.

Solution: Time stamp protocol is a deadlock free protocol.

Ans. (a)

15. Data integrity control makes use of ________ for maintaining the integrity of database.

(a) specific alphabets (uppercase and lowercase alphabets)
(b) passwords
(c) relational algebra
(d) storing on a backup hard disk

Solution: It promotes and enforces some integrity rules for the reducing data redundancy and increasing data consistency.

Ans. (a)

16. Data warehouse provides

(a) transaction responsiveness.
(b) storage, functionality responsiveness to queries.
(c) demand and supply responsiveness.
(d) storage of transactions.

Solution: It provides data storage and responsiveness to queries.

Ans. (a)

17. If D_1, D_2, … D_n are domains in a relational model, then the relation is a table, which is a subset of

(a) $D_1 \oplus D_2 \oplus \ldots \oplus D_n$ (b) $D_1 \times D_2 \times \ldots \times D_n$
(c) $D_1 \cup D_2 \cup \ldots \cup D_n$ (d) $D_1 \cap D_2 \cap \ldots \cap D_n$

Solution: $D_1 \times D_2 \times \ldots \times D_n$

Ans. (a)

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

(a) 2NF (b) 5NF
(c) 4NF (d) 3NF

Solution: Decomposition in BCNF is not always lossless and dependency preserving. So, 3NF is considered to be most adequate form in relational database.

Ans. (d)

19. Let $R = (A, B, C, D, E, F)$ be a relation scheme with the following dependencies $C \rightarrow F$, $E \rightarrow A$, $EC \rightarrow D$, $A \rightarrow B$. Which of the following is a key for R?

(a) CD (b) EC (c) AE (d) AC

Solution: To find key, we try to find closure.

$$(CD)+ = \{C, D, F\}$$
$$(EC)+ = \{A, B, C, D, E, F\}$$
$$(AE)+ = \{A, B, E\}$$
$$(AC)+ = \{A, B, C, F\}$$

Only EC satisfies the closure property.

Ans. (b)

20. Give the following relation instance:

x	y	z
1	4	2
1	5	3
1	6	3
3	2	2

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

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

Solution: Functional dependency should uniquely identify the value. From the given options only the following satisfies this condition:

$$XY \to Z,\ YZ \to X,\ Y \to Z \textbf{ AND } Y \to X$$

Ans. (b)

21. Relation R is decomposed using a set of functional dependencies F, and relation S is decomposed using another set of functional dependencies G. One decomposition is definitely BCNF, the other is definitely 3NF, but it is not known which is which. To make a guaranteed identification, which on one of the following tests should be used on the decompositions? (Assume that the closures of F and G are available.)

(a) Dependency-preservation
(b) Lossless-join
(c) BCNF definition
(d) 3NF definition

Solution: All the 3NF and BCNF decomposition guarantee lossless join. All 3NF and BCNF decomposition does not guaranty dependency preservation. According to this, only option (c) can be correct.

Ans. (c)

22. Consider the join of a relation R with a relation S. If R has m tuples and S has n tuples then the maximum and minimum sizes of the join, respectively, are

(a) $m + n$ and 0.
(b) mn and 0.
(c) $m + n$ and $|m - n|$.
(d) mn and $m + n$.

Solution: Number of tuples in their join gets multiplied with number of tuples in both the relations. So, the maximum and minimum tuples will be mn and 0, respectively.

Ans. (b)

23. Given the relations

Employee(name, salary, deptno) and department (deptno, deptname, address), which of the following queries cannot be expressed using the basic relational algebra operations (σ, π, $\bowtie$, E, $\cap$, $-$)?

(a) Department address of every employee
(b) Employees whose name is the same as their department name
(c) The sum of all employees' salaries
(d) All employees of a given department

Solution: Aggregate functions such as sum, avg, min and max cannot be expressed in terms of basic relational algebra operations. These require extended relational algebra.

Ans. (c)

GATE PREVIOUS YEARS' QUESTIONS

1. Which of the following scenarios may lead to an irrecoverable error in a database system?

(a) A transaction writes a data item after it is read by an uncommitted transaction.
(b) A transaction reads a data item after it is read by an uncommitted transaction.
(c) A transaction reads a data item after it is written by a committed transaction.
(d) A transaction reads a data item after it is written by an uncommitted transaction.

(GATE 2003: 1 Mark)

Solution: In option (d), a problem will occur. Suppose first transaction A is reading data written by second transaction B, but B has not committed yet. If A acts on that data and eventually commits, but B rollback. A will have "acted on" data that was never committed. So it will lead to irrecoverable error in database.

Ans. (d)

2. Consider the following SQL query:

SELECT distinct $a_1, a_2, K, a_n a$
FROM $r_1, r_2, L, r_m r$
WHERE P

For an arbitrary predicate P, this query is equivalent to which of the following relational algebra expressions?

(a) $\Pi_{a_1,a_2,\ldots,a_n} \sigma_p (r_1 \times r_2 \times \cdots \times r_m)$

(b) $\Pi_{a_1,a_2,K,\ a_n} \sigma_p \ (r_1 \bowtie r_2 \bowtie \mathrm{L} \bowtie r_\mathrm{m})$

(c) $\Pi_{a_1,a_2,K,a_n} \sigma_p \left(r_1 \cup r_2 \cup L \cup r_m\right)$

(d) $\Pi_{a_1,a_2,K,a_n} \sigma_p \left(r_1 \cap r_2 \cap L \cap r_m\right)$

(GATE 2003: 1 Mark)

Solution: Π is projection that by default selects distinct value of attributes. Cross product will consider all the tables r_1, r_2, … r_n. Row (σ) selects all the rows satisfying predicate P.

Ans. (a)

3. Consider the following functional dependencies in a database:

```
Date_of_birth → Age
Age → Eligibility
Name → Roll_number
Roll_number → Name
Course_number → Course_name
Course_number → Instructor
(Roll_number, Course_number) → Grade
```

The relation (Roll_number, Name, Date_of_birth, Age) is

(a) In Second normal form but not in Third normal form
(b) In Third normal form but not in BCNF
(c) In BCNF
(d) In none of the above

(GATE 2003: 2 Marks)

Solution: The applicable FDs are:

```
Date_of_birth → Age
Age → Eligibility
Name → Roll_number
Roll_number → Name
```

The candidate keys are

(Roll_number, Date_of_birth) and (Name, Date_of_birth)

For candidate key the FD Date_of_birth → Age is a partial dependency.

So, the relation is in 1NF.
Candidate keys for given relation are: (date of birth, name) and (date of birth, roll_number)

Data of Birth → Age

Age (non-prime) is partially dependent upon Date of Birth which is prime attribute. According to 2NF definition, every non-prime attribute should be fully functional dependent on key, so it is not in 2NF. Therefore, it will not be in 3NF and BCNF also.

Ans. (d)

4. Consider the set of relations shown below and the SQL query that follows:

```
Students: (Roll_number, Name, Date_of_
birth)
Courses: (Course number, Course_name,
Instructor)
Grades: (Roll_number, Course_number, Grade)
SELECT distinct Name
FROM Students, Courses, Grades
WHERE Students.Roll_number = Grades.
Roll_number
    and Courses.Instructor = Shyam
    and Courses.Course_number = Grades.
    Course_number
    and Grades.grade = A
```

Which of the following sets is computed by the above query?

(a) Names of students who have got an A grade in all courses taught by Shyam.
(b) Names of students who have got an A grade in all courses.
(c) Names of students who have got an A grade in at least one of the courses taught by Shyam.
(d) None of the above.

(GATE 2003: 2 Marks)

Solution: Use of distinct selects the name only once. Due to distinct selected names will be unique, so one person will be selected only once, in spite of getting grade A in any number of courses taught by Shyam.

Ans. (c)

5. Consider three data items D_1, D_2 and D_3, and the following execution schedule of transactions T_1, T_2 and T_3. In the diagram, $R(D)$ and $W(D)$ denote the actions reading and writing the data item D, respectively.

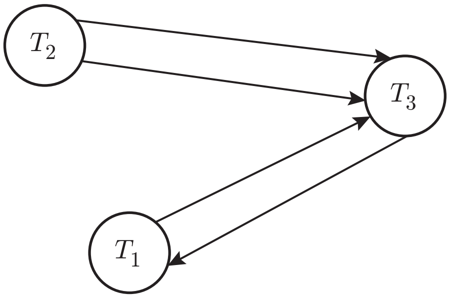
T	T_2	T_3
	$R(D_3)$;	
	$R(D_2)$;	
	$W(D_2)$;	
		$R(D_2)$;
		$R(D_3)$;
$R(D_1)$;		
$W(D_1)$;		
		$W(D_2)$;
		$W(D_3)$;
	$R(D_1)$;	
$R(D_2)$;		
$W(D_2)$;		
	$W(D_1)$;	

Which of the following statements is correct?

(a) The schedule is serializable as T_2; T_3; T_1.
(b) The schedule is serializable as T_2; T_1; T_3.
(c) The schedule is serializable as T_3; T_2; T_1.
(d) The schedule is not serializable.

(GATE 2003: 2 Marks)

Solution: Let us draw a flow diagram between the three schedules. An edge is drawn between two schedules if there exists read-write or write-write dependency between them.



The schedule has cycle between T_1 and T_3. So, the schedule is not serializable.

Ans. (d)

6. Let $R_1(A, B, C, D)$ and $R_2(D, E)$ be two relation schema where the primary keys are shown underlined, and let C be a foreign key in R_1 referring to R_2. Suppose there is no violation of the above referential integrity constraint in the corresponding relation instances r_1 and r_2. Which one of the following relational algebra expressions would necessarily produce an empty relation?

(a) $\Pi_D(r_2) - \Pi_C(r_1)$
(b) $\Pi_C(r_1) - \Pi_D(r_2)$
(c) $\Pi_D(r_1 \bowtie_{C \neq D} r_2)$
(d) $\Pi_D(r_1 \bowtie_{C = D} r_2)$

(GATE 2004: 1 Mark)

Solution: C is foreign key in relation R1 referring to relation R2. So, C would not contain any additional value that is not present in D. So $\Pi_C(r_1) - \Pi_D(r_2)$ will return no value.
Returns all the tuples which are in $\Pi_C(r_1)$ but not in $\Pi_D(r_2)$.

Ans. (b)

7. Consider the following relation schema pertaining to a student's database:

```
Students(rollno, name, address)
Enroll(rollno, courseno, coursename)
```

where the primary keys are shown underlined. The number of tuples in the student and Enroll tables are 120 and 8, respectively. What are the maximum and minimum number of tuples that can be present in (Student * Enroll), where '*' denotes natural join?

(a) 8, 8
(b) 120, 8
(c) 960, 8
(d) 960, 120

(GATE 2004: 1 Mark)

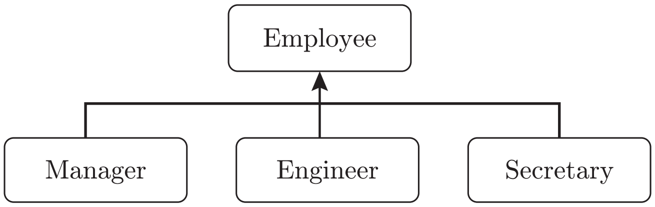
Solution: The maximum and minimum number of tuples presented in (Student * Enroll) would be represented by the minimum of these 120 and 8, that is, min(120, 8) = 8
Natural join has inbuilt condition of equality. So in both the relations, minimum and maximum tuples will be 8.

Ans. (a)

8. It is desired to design an object-oriented employee record system for a company. Each employee has a name, unique id and salary. Employees belong to different categories and their salary is determined by their category. The functions getName, getId and compute salary are required. Given the class hierarchy below, possible locations for these functions are:

(i) getId is implemented in the superclass
(ii) getId is implemented in the subclass
(iii) getName is an abstract function in the superclass
(iv) getName is implemented in the superclass
(v) getName is implemented in the subclass
(vi) getSalary is an abstract function in the superclass
(vii) getSalary is implemented in the superclass
(viii) getSalary is implemented in the subclass

(GATE 2004: 2 Marks)



Choose the best design

(a) (i), (iv), (vi), (viii)
(b) (i), (iv), (vii)
(c) (i), (iii), (v), (vi), (viii)
(d) (ii), (v), (viii)

Solution: getid and getname can be placed in base class. Because all the subclasses will have the same implementation for these functions. But getsalary is dependent on category of employee. So, abstract function can be placed in base class and its implementation in the subclasses.

Ans. (a)

9. The relation scheme Student Performance (Name, CourseNo, RollNo, Grade) has the following functional dependencies:

Name, CourseNo → Grade
RollNo, CourseNo → Grade
Name → RollNo
RollNo → Name

The highest normal form of this relation scheme is

(a) 2NF (b) 3NF
(c) BCNF (d) 4NF

(GATE 2004: 2 Marks)

Solution: Name and RollNo are candidate keys.
The attributes are not repeated. So, the relation schema is in 1 NF.
There is no partial dependency. So, the relation is in 2NF.
Grade is not fully dependent upon all candidate keys, so it is not in 3NF.
Candidate keys for given relation are: (rollNo, courseNo) and (name, courseNo) there is no partial and transitive dependencies. So the highest normal form is 3NF. name → rollNo and rollNo → name are violating the condition of BCNF.

Ans. (b)

10. Consider the relation Student (name, sex, marks), where the primary key is shown underlined, pertaining to students in a class that has at least one boy and one girl. What does the following relational algebra expression produce?

(Note: ρ is the rename operator).

$$\Pi_{\text{name}}(r_{\text{sex=female}}(\text{Student}) - \Pi_{\text{name}}(\text{Student}_{\text{sex=female}\,\Lambda x=\text{male}\Lambda \text{marks}\le m}\,\rho_{n,x,m}(\text{Student}))$$

(a) Names of girl students with the highest marks
(b) Names of girl students with more marks than some boy student
(c) Names of girl students with marks not less than some boy student
(d) Names of girl students with more marks than all the boy students

(GATE 2004: 2 Marks)

Solution: The query is Part 1 − Part 2. Part 2 of the query results in names of girls having marks less than or equal to marks of all the boys. And Part 1 selects the names of all the girls in class. So, Part 1 − Part 2 will result in names of girl students having more marks than all the boy students.

Ans. (d)

11. The order of an internal node in a B+ tree index is the maximum number of children it can have. Suppose that a child pointer takes 6 bytes, the search field value takes 14 bytes, and the block size is 512 bytes. What is the order of the internal node?

(a) 24 (b) 25
(c) 26 (d) 27

(GATE 2004: 2 Marks)

Solution: Order of internal node is $P_I \cdot CP + (P_I - 1)$ KV <= Block Size
Child pointer (CP) = 6 B
Key value (KV) = 14 B
Block size = 512 B
$6 + (P_I - 1) \times 14 = 512$
$P_I = 26$

Ans. (c)

12. The employee information in a company is stored in the following relation:

```
Employee(name, sex, salary, deptName)
```

Consider the following SQL query:

```
SELECT deptName
FROM Employee
WHERE sex = 'M'
GROUP by deptName
HAVING avg(salary)>
 (SELECT avg (salary) FROM Employee)
```

It returns the names of the department in which

(a) the average salary is more than the average salary in the company.
(b) the average salary of male employees is more than the average salary of all male employees in the company.
(c) the average salary of male employees is more than the average salary of employees in the same department.
(d) the average salary of male employees is more than the average salary in the company.

(GATE 2004: 2 Marks)

Solution: Inner query selects the average salary of employees in the company. Outer query selects average salary of male employees. So, the whole query finds the department name of whose average salary of male employees is more than the average salary in the company.

Ans. (d)

13. Which one of the following is a key factor for preferring B+ trees to binary search trees for indexing database relations?

(a) Database relations have a large number of records.
(b) Database relations are sorted on the primary key.
(c) B+ trees require less memory than binary search trees.
(d) Data transfer from disks is in blocks.

(GATE 2005: 1 Mark)

Solution: Indexing performs well for large data blocks.
B+ trees are preferred over the binary search trees.
B+ trees transfer data from disk to primary memory in form of data blocks.
In case of B+ trees, data moves in terms of blocks.

Ans. (d)

14. Which one of the following statements about normal forms is FALSE?

(a) BCNF is stricter than 3NF.
(b) Lossless, dependency-preserving decomposition into 3NF is always possible.
(c) Lossless, dependency-preserving decomposition into BCNF is always possible.
(d) Any relation with two attributes is in BCNF.

(GATE 2005: 1 Mark)

Solution: It is not always possible to decompose a table in BCNF and preserve dependencies. For example, a set of functional dependencies $\{AB \rightarrow C, C \rightarrow B\}$ cannot be decomposed in BCNF.

Ans. (c)

15. Let r be a relation instance with schema $R = (A, B, C, D)$. We define $r_1 = \Pi_{\text{A, B, C}}(R)$ and $r_2 = \Pi_{\text{A, D}}(r)$. Let $s = r_1 * r_2$ where $*$ denotes natural join. Given that the decomposition of r into r_1 and r_1 is lossy, which one of the following is TRUE?

(a) $s \subset r$ (b) $r \cup s = r$
(c) $r \subset s$ (d) $r * s = s$

(GATE 2005: 1 Mark)

Solution:

Table r

A	*B*	*C*	*D*
1	10	100	1000
1	20	200	1000
	20	200	2000

Table r_1

A	*B*	*C*
1	10	100
1	20	200

Table r_2

A	*D*
1	1000
1	2000

Table s (natural join of r_1 and r_2)

A	*B*	*C*	*D*
1	10	100	1000
1	20	200	1000
1	20	100	200
1	20	200	2000

Ans. (c)

16. Let E_1 and E_2 be two entities in an E/R diagram with simple single-valued attributes. R_1 and R_2 are two relationships between E_1 and E_2, where R_1 is one-to-many and R_2 is many-to-many. R_1 and R_2 do not have any attributes of heir own. What is the minimum number of tables required to represent this situation in the relational model?

(a) 2 (b) 3
(c) 4 (d) 5

(GATE 2005: 2 Marks)

Solution:

E_1
l
m
n

E_2
x
y
z

R_2

E_1	E_2
L	x
L	y
M	y

The one-to-many relationships are represented with entity set from one side. In one-to-many relationships, each entity in the entity set can be associated with at most one entity of the other. So, the table is not formed for R_1. Hence, the tables are formed for R_2, E_1 and E_2.
Using cross-reference technique, minimum of 3 tables are required.

Ans. (b)

17. The following table has two attributes A and C where A is the primary key and C is the foreign key referencing A with on-delete cascade.

A	*C*
2	4
3	4

(Continued)

Continued

A	*C*
4	3
5	2
7	2
9	5
6	4

The set of all tuples that must be additionally deleted to preserve referential integrity when the tuple (2, 4) is deleted is

(a) (3, 4) and (6, 4)
(b) (5, 2) and (7, 2)
(c) (5, 2), (7, 2) and (9, 5)
(d) (3, 4), (4, 3) and (6, 4)

(GATE 2005: 2 Marks)

Solution: When (2,4) is deleted, as C is a foreign key referring A with delete on cascade, all entries with value 2 in C must be deleted.
So, (5, 2) and (7, 2) are deleted. As a result of this, 5 and 7 are deleted from A, which causes (9, 5) to be deleted.

Ans. (c)

18. The relation book (title, price) contains the titles and prices of different books. Assuming that no two books have the same price, what does the following SQL query list?

```
SELECT title
FROM book as B
WHERE (SELECT count(*)
          FROM book as T
          WHERE T.price > B.price) < 5
```

(a) Titles of the four most expensive books
(b) Title of the fifth most inexpensive book
(c) Title of the fifth most expensive book
(d) Titles of the five most expensive books

(GATE 2005: 2 Marks)

Solution: The outer query selects all titles from book table. For every selected book, the sub-query returns count of those books, which are more expensive than the selected book. The where clause of outer query will be true for five most expensive books.

Ans. (d)

19. Consider a relation scheme $R = (A, B, C, D, E, H)$ on which the following functional dependencies hold:

$$\{A \rightarrow B, BC \rightarrow D, E \rightarrow C, D \rightarrow A\}.$$

What are the candidate keys of R?

(a) *AE, BE* (b) *AE, BE, DE*
(c) *AEH, BEH, BCH* (d) *AEH, BEH, DEH*

(GATE 2005: 2 Marks)

Solution: Let S be a candidate key of relation R if the closure of S is all attributes of R and there is no subset of S whose closure is all attributes of R.

Closure of AEH: $AEH+ = \{ABCDEH\}$

Closure of BEH: $BEH+ = \{ABCDEH\}$

Closure of DEH: $DEH+ = \{ABCDEH\}$

Ans. (d)

20. Consider the following log sequence of two transactions on a bank account, with initial balance 12000, that transfer 2000 to a mortgage payment and then apply a 5% interest.

1. T_1 start
2. T_1 B old = 1200 new = 10000
3. T_1 M old = 0 new = 2000
4. T_1 commit
5. T_2 start
6. T_2 B old = 10000 new = 10500
7. T_2 commit

Suppose the database system crashes just before log record 7 is written. When the system is restarted, which one statement is true of the recovery procedure?

(a) We must redo log record 6 to set B to 10500.
(b) We must undo log record 6 to set B to 10000 and then redo log records 2 and 3.
(c) We need not redo log records 2 and 3 because transaction T_1 has committed.
(d) We can apply redo and undo operations in arbitrary order because they are idempotent.

(GATE 2006: 1 Mark)

Solution: When a transaction is committed, no need to redo or undo operations.

Ans. (c)

21. Consider the relation account (customer, balance) where customer is a primary key and there are no null values. We would like to rank customers according to decreasing balance. The customer with the largest balance gets rank 1. Ties are not broke but ranks are skipped: if exactly two customers have the largest balance they each get rank 1, and rank 2 is not assigned.

Query 1: `SELECT A.customer, count(B.customer) FROM account A, account B WHERE A.balance ⇐ B.balance group by A.customer`

Query 2: `SELECT A.customer, 1+count(B.customer) FROM account A, account B WHERE A.balance < B.balance group by A.customer`

Consider these statements about Query 1 and Query 2.

I. Query 1 will produce the same row set as Query 2 for some but not all databases.
II. Both Query 1 and Query 2 are correct implementation of the specification.
III. Query 1 is a correct implementation of the specification but Query 2 is not.
IV. Neither Query 1 nor Query 2 is a correct implementation of the specification.
V. Assigning rank with a pure relational query takes less time than scanning in decreasing balance order assigning ranks using ODBC.

Which two of the above statements are correct?

(a) II and V (b) I and III
(c) I and IV (d) III and V

(GATE 2006: 2 Marks)

Solution: Both query1 and query2 perform the same task, but none of them sort the customer according to the decreasing balance. So, option (c) is correct.

Ans. (c)

22. Consider the relation enrolled (student, course) in which (student, course) is the primary key, and the relation paid (student, amount) where student is the primary key. Assume no null values and no foreign keys or integrity constraints. Given the following four queries:

Query 1: `SELECT student FROM enrolled WHERE student in (SELECT student FROM paid)`

Query 2: `SELECT student FROM paid WHERE student in (SELECT student FROM enrolled)`

Query 3: `SELECT E.student FROM enrolled E, paid P WHERE E.student = P.student`

Query 4: `SELECT student FROM paid WHERE exists`

`(SELECT * FROM enrolled WHERE enrolled.student = paid.student)`

Which one of the following statements is correct?

(a) All queries return identical row sets for any database
(b) Query 2 and Query 4 return identical row sets for all databases but there exist databases for which Query 1 and Query 2 return different row sets.
(c) There exist databases for which Query 3 returns strictly fewer rows than Query 2
(d) There exist databases for which Query 4 will encounter an integrity violation at runtime.

(GATE 2006: 2 Marks)

Solution: The output of Query 2, Query 3 and Query 4 will be identical. Query 1 produces duplicate rows. But row set produced by all the queries will be same. For example,

Table: Enrolled

Student	Course
A	C_1
B	C_1
A	C_2
C	C_1

Table: Enrolled

Student	Amount
A	100
B	100
D	200

Output of Query 1: A B

Output of Query 2: A B

Output of Query 3: A B

Output of Query 4: A B

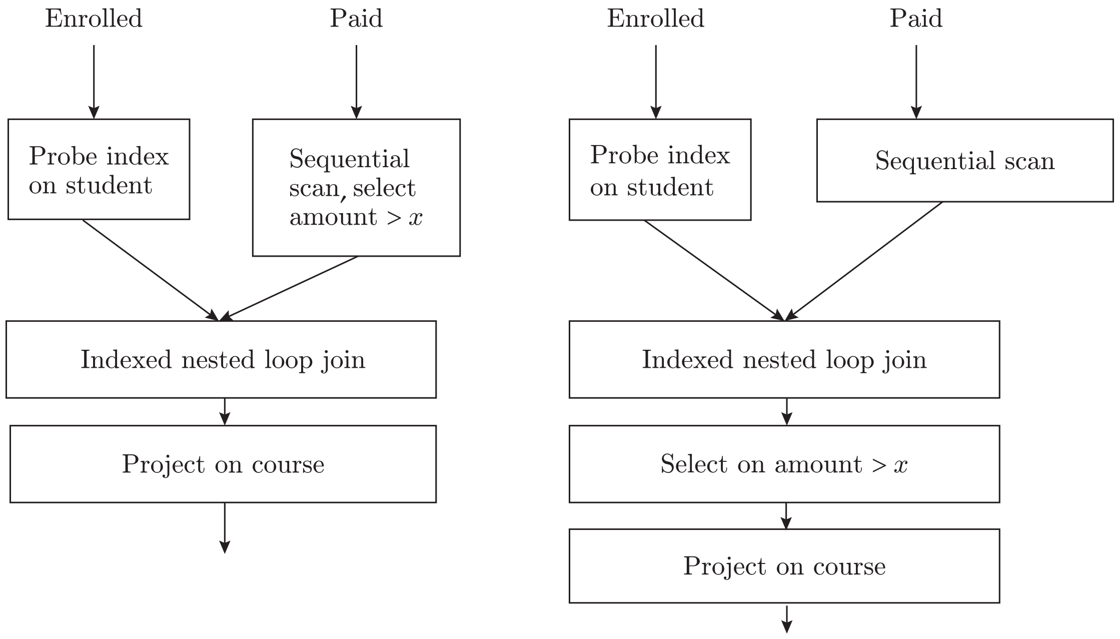
Ans. (a)

23. Consider the relation enrolled (student, course) in which (student, course) is the primary key, and the relation paid (student, amount) in which student is the primary key. Assume no null values and no foreign keys or integrity constraints. Assume that amounts 6000, 7000, 8000, 9000 and 10000 were each paid by 20% of the students. Consider these query plans (Plan 1 on left, Plan 2 on right) to "list all courses taken by students who have paid more than x".

A disk seek takes 4 ms, disk data transfer bandwidth is 300 MB/s and checking a tuple to see if amount is greater than x takes 10 µs. Which of the following statements is correct?

(a) Plan 1 and Plan 2 will not output identical row sets for all databases.
(b) A course may be listed more than once in the output of Plan 1 for some databases.
(c) For $x = 5000$, Plan 1 executes faster than Plan 2 for all databases.
(d) For $x = 9000$, Plan 1 executes slower than Plan 2 for all databases.

(GATE 2006: 2 Marks)



Solution: Plans need to load both tables' courses and enrolled. So, disk access time is the same for both plans.

Plan 2 does lesser number of comparisons compared to Plan 1.

So, join operation will require more comparisons as the second table will have more rows in Plan 2 compared to Plan 1.

The joined table of two tables will have more rows, so, more comparisons are needed to find amounts greater than x.

Plan 1 executes faster than Plan 2. First tuples are filtered then joined, whereas in plan 2 first tuples are joined and then filtered, which takes more time.

Ans. (c)

24. The following functional dependencies are given:

$AB \to CD$, $AF \to D$, $DE \to F$, $C \to G$, $F \to E$, $G \to A$

Which one of the following options is false?

(a) $\{CF\}+ = \{ACDEFG\}$
(b) $\{BG\}+ = \{ABCDG\}$
(c) $\{AF\}+ = \{ACDEFG\}$
(d) $\{AB\}+ = \{ABCDFG\}$

(GATE 2006: 2 Marks)

Solution: Closure of AF or $AF+ = \{ADEF\}$, closure of AF does not contain C and G.

The closure of {AF}+ is {A F D E}. It cannot drive all the members of set given in option (c). So, option (c) is false.

Ans. (c)

25. Information about a collection of students is given by the relation studinfo(studId, name, sex). The relation enroll(studId, courseId) gives which student has enrolled for (or taken) what course(s). Assume that every course is taken by at least one male and at least one female student. What does the following relational algebra expression represent?

$$\Pi_{\text{courseId}}(\Pi_{\text{studId}}(\sigma_{\text{sex="female"}}(\text{studInfo})) \times \Pi_{\text{courseId}}(\text{enroll}))$$

(a) Courses in which all the female students are enrolled.
(b) Courses in which a proper subset of female students are enrolled.
(c) Courses in which only male students are enrolled.
(d) None of the above

(GATE 2007: 2 Marks)

Solution:

(i) Select studId of all female students and select all courseId of all courses.
(ii) The query performs a Cartesian product of the above select two columns in Step 1.
(iii) It subtracts enroll table from the result of Step 2.

This will remove all the (studId, courseId) pairs, which are present in enrol table.

If all female students have registered in courses, then this course will not be there in the subtracted result.

So, the complete expression returns courses in which a proper subset of female students is enrolled.

Ans. (b)

26. Consider the relation employee (name, sex, supervisorName) with name as the key. `supervisorName` gives the name of the supervisor of the employee under consideration. What does the following Tuple Relational Calculus query produce?

```
{e.name|employee(e) ∧{(∀x)
[¬employee(x) ∨ x.supervisorName ≠
e.name ∨ x.sex = "male")]}/
```

inserted on the left of expression

(a) Names of employees with a male supervisor.
(b) Names of employees with no immediate male subordinates.
(c) Names of employees with no immediate female subordinates.
(d) Names of employees with a female supervisor.

(GATE 2007: 2 Marks)

Solution: The query selects all those employees whose immediate subordinate is "male".

Ans. (b)

27. Consider the table employee (empId, name, department, salary) and the two queries Q_1, Q_2 below. Assuming that department 5 has more than one employee, and we want to find the employees who get higher salary than anyone in the department 5, which one of the statements is TRUE for any arbitrary employee table?

Q_1:
```
SELECT e.empId 1
FROM employee e
WHERE not exists (SELECT * FROM
employee s WHERE s.department = "5"
and s.salary >= e.salary)
```

Q_2:
```
SELECT e.empId
FROM employee e
WHERE e.salary > Any
(SELECT distinct salary FROM em-
ployee s WHERE s.department = "5")
```

(a) Q_1 is the correct query.
(b) Q_2 is the correct query.
(c) Both Q_1 and Q_2 produce the same answer.
(d) Neither Q_1 nor Q_2 is the correct query.

(GATE 2007: 2 Marks)

Solution: Q_1 selects an employee but does not compute the result as after the 'Where not exists', it does not have statement to produce the result.

Q_2 selects an employee who gets higher salary than anyone in the department 5. It correctly finds employees who get higher salary than anyone in the department 5.

Ans. (b)

28. Which one of the following statements is FALSE?

(a) Any relation with two attributes is in BCNF.
(b) A relation in which every key has only one attribute is in 2NF.
(c) A prime attribute can be transitively dependent on a key in a 3NF relation.
(d) A prime attribute can be transitively dependent on a key in a BCNF relation.

(GATE 2007: 2 Marks)

Solution: According to the definition of 3NF, a prime attribute can be transitively dependent on a key. Option (d) is incorrect because this does not satisfy the condition of BCNF.

Ans. (d)

29. The order of a leaf node in a B+ tree is the maximum number of (value, data record pointer) pairs it can hold. Given that the block size is 1 KB, data record pointer is 7 bytes long, the value field is 9 bytes long and a block pointer is 6 bytes long, what is the order of the leaf node?

(a) 63 (b) 64
(c) 67 (d) 68

(GATE 2007: 2 Marks)

Solution: Let the order of the leaf node be n.

Block size = 1 KB = 1024 bits

$$6 + 7n + (n - 1)9 = 1024$$
$$n = 64$$

Ans. (b)

30. Consider the following schedules involving two transactions. Which one of the following statements is TRUE?

S_1: $r_1(X)$; $r_1(Y)$; $r_2(X)$; $r_2(Y)$; $w_2(Y)$; $w_1(X)$

S_2: $r_1(X)$; $r_2(X)$; $r_2(Y)$; $w_2(Y)$; $r_1(Y)$; $w_1(X)$

(a) Both S_1 and S_2 are conflict serializable.
(b) S_1 is conflict serializable and S_2 is not conflict serializable.
(c) S_1 is not conflict serializable and S_2 is conflict serializable.
(d) Both S_1 and S_2 are not conflict serializable.

(GATE 2007: 2 Marks)

Solution:

Schedule S_1

T_1	T_2
$r_1(X)$	
$r_1(Y)$	
	$r_2(X)$
	$r_2(Y)$
	$w_2(Y)$
$w_1(X)$	

The schedule is not conflict serializable.

Schedule S_2

T_1	T_2
$r_1(X)$	
	$r_2(X)$
	$r_2(Y)$
	$w_2(Y)$
$r_1(Y)$	
$w_1(X)$	

The schedule is conflict serializable to $T_2 T_1$.

Ans. (c)

31. A clustering index is defined on the fields which are of type

(a) Non-key and ordering
(b) Non-key and non-ordering
(c) Key and ordering
(d) Key and non-ordering

(GATE 2008: 1 Mark)

Solution: Clustering index is defined on the fields. If the records of the file are physically ordered on a non-key field, it will not have a distinct value for each record. So, the clustering index is defined on the fields of type non-key and ordering.

Ans. (a)

32. Which of the following tuple relational calculus expression(s) is/are equivalent to $\forall t \in r\ (P(t))$?

I. $\neg\exists t \in r\ (P(t))$

II. $\exists t \notin r\ (P(t))$

III. $\neg\exists t \in r(\neg P(t))$

IV. $\exists t \notin r\ (\neg P(t))$

(a) I only (b) II only
(c) III only (d) III and IV only

(GATE 2008: 1 Mark)

Solution: Using negation theorem, we find option (c) is true.

Ans. (c)

33. Let R and S be two relations with the following schema:
$R(\underline{P, Q}, R_1, R_2, R_3)$
$S(\underline{P, Q}, S_1, S_2)$
where $\{P, Q\}$ is the key for both schemas. Which of the following queries are equivalent?

I. $\Pi_P(R \bowtie S)$

II. $\Pi_P(R) \bowtie \Pi_P(S)$

III. $\Pi_P\left(\Pi_{P,Q}(R) \cap \Pi_{P,Q}(S)\right)$.

IV. $\Pi_P\left(\Pi_{P,Q}(R) - \left(\Pi_{P,Q}(R) - \Pi_{P,Q}(S)\right)\right)$

(a) Only I and II (b) Only I and III
(c) Only I, II and III (d) Only I, III and IV

(GATE 2008: 2 Marks)

Solution:
In I, P from natural join of R and S are selected.
In III, all P from intersection of (P, Q) pairs present in R and S.
IV is also equivalent to III because $(R - (R - S)) = R \cap S$.
II is not equivalent as it may also include P, where Q is not same in R and S.
Queries I, II and III performs the same operations, that is, they select attribute P.

Ans. (c)

34. Consider the following relational schemes for a library database:
Book(Title, Author, Catalog_no, Publisher, Year, Price)
Collection(Title, Author, Catalog_no)
within the following functional dependencies:

I. Title Author → Catalog_no
II. Catalog_no → Title Author Publisher Year
III. Publisher Title Year → Price

Assume {Author, Title} is the key for both schemes. Which of the following statements is true?

(a) Both Book and Collection are in BCNF.
(b) Both Book and Collection are in 3NF only.
(c) Book is in 2NF and Collection is in 3NF.
(d) Both Book and Collection are in 2NF only.

(GATE 2008: 2 Marks)

Solution: Collection is in BCNF as there is only one functional dependency Title Author → Catalog_no and {Author, Title} is key for collection.

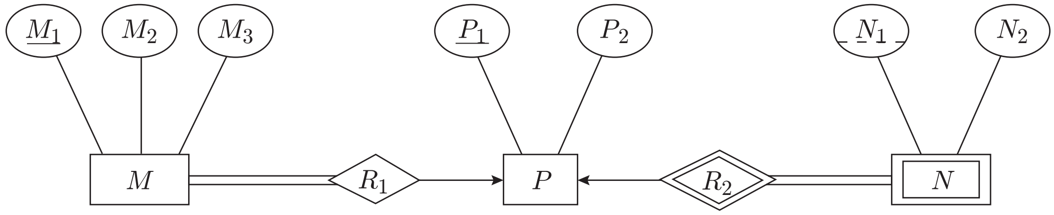
Book is not in BCNF because Catalog_no is not a key and there is a functional dependency Catalog_no → Title Author Publisher Year.

Book is not in 3NF because non-prime attributes (Publisher Year) are transitively dependent on key [Title, Author].

Book is in 2NF because every non-prime attribute of the table is dependent on either the key [Title, Author] or another non-prime attribute.

Ans. (c)

Linked Answer Questions 35 and 36: Consider the following ER diagram:



35. The minimum number of tables needed to represent M, N, P, R_1, R_2 is

(a) 2 (b) 3
(c) 4 (d) 5

(GATE 2008: 2 Marks)

Solution: Many-to-one and one-to-many relationship sets that are total on the many-side can be represented by adding an extra attribute to the "many" side, containing the primary key of the "one" side.

Since R_1 is many to one and participation of M is total, M and R_1 can be combined to form the table $\{M_1, M_2, M_3, P_1\}$. N is a week entity set, so it can be combined with P.

Ans. (a)

36. Which of the following is a correct attribute set for one of the tables for the correct answer to the above question?

(a) $\{M_1, M_2, M_3, P_1\}$
(b) $\{M_1, P_1, N_1, N_2\}$
(c) $\{M_1, P_1, N_1\}$
(d) $\{M_1, P_1\}$

(GATE 2008: 2 Marks)

Solution:
For R_1 correct attribute set: M_1, M_2, M_3, P_1

For R_2 correct attribute set: N_1, N_2, P_1, P_2 with P_1 as primary key and N_1 as weak entity set.

Ans. (a)

37. Consider two transactions T_1 and T_2, and four schedules S_1, S_2, S_3, S_4 of T_1 and T_2 as given below:

T_1: $R_1[x]\,W_1[x]\,W_1[y]$

T_2: $R_2[x]\,R_2[y]\,W_2[y]$

S_1: $R_1[x]\,R_2[x]\,R_2[y]\,W_1[x]\,W_1[y]\,W_2[y]$

S_2: $R_1[x]\,R_2[x]\,R_2[y]\,W_1[x]\,W_2[y]\,W_1[y]$

S_3: $R_1[x]\,W_1[x]\,R_2[x]\,W_1[y]\,R_2[y]\,W_2[y]$

S_4: $R_2[x]\,R_2[y]\,R_1[x]\,W_1[x]\,W_1[y]\,W_2[y]$

Which of the above schedules are conflict-serializable?

(a) S_1 and S_2 (b) S_2 and S_3
(c) S_3 only (d) S_4 only

(GATE 2009: 2 Marks)

Solution: The serial schedule T_1T_2 has the following sequence of operations:

$$R_1[x]\,W_1[x]\,W_1[y]\,R_2[x]\,R_2[y]\,W_2[y]$$

And the schedule T_2T_1 has the following sequence of operations:

$$R_2[x]\,R_2[y]\,W_2[y]\,R_1[x]\,W_1[x]\,W_1[y]$$

The schedule S_2 is conflict-equivalent to T_2T_1 and S_3 is conflict-equivalent to T_1T_2.
Only S_2 and S_3 are conflict serialzable schedules. To test for conflict serializability, make the wait for graph. If there is a cycle in graph, it means not conflict serializable.

Ans. (b)

38. The following key values are inserted into a B+ tree in which order of the internal nodes is 3, and that of the leaf nodes is 2, in the sequence given below. The order of internal nodes is the maximum number of tree pointers in each node, and the order of leaf nodes is the maximum number of data items that can be stored in it. The B+ tree is initially empty.

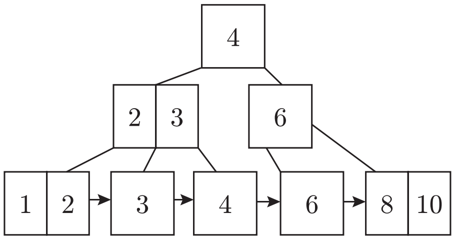
10, 3, 6, 8, 4, 2, 1

The maximum number of times leaf nodes would get split up as a result of these insertions is

(a) 2 (b) 3
(c) 4 (d) 5

(GATE 2009: 2 Marks)

Solution: Total number of splits here are 5.



Ans. (a)

39. Let R and S be relational schemes such that $R = \{a, b, c\}$ and $S = \{c\}$. Now consider the following queries on the database:

I. $\Pi_{R-S}(r) - \Pi_{R-S}(\Pi_{R-S}(r) \times s - \Pi_{R-S,S}(r))$

II. $\{t \mid t \in \Pi_{R-S}(r) \wedge \forall u \in s(\exists v \in r(u = v[s] \wedge t = v[R-S]))\}$

III. $\{t \mid t \in \Pi_{R-S}(r) \wedge \forall v \in r(\exists u \in s(u = v[s] \wedge t = v[R-S]))\}$

IV. `SELECT R.a, R.b FROM R, S WHERE R.c = S.c`

Which of the above queries are equivalent?

(a) I and II (b) I and III
(c) II and IV (d) III and IV

(GATE 2009: 2 Marks)

Solution: I and II describe the division operator in relational algebra and tuple relational calculus, respectively.

Ans. (a)

Common Data Questions 40 and 41: Consider the following relational schema:

```
Suppliers(sid: integer, sname: string,
city: string, street: string)

Parts(pid: integer, pname: string,
color: string)

Catalog(sid: integer, pid: integer,
cost: real)
```

40. Consider the following relational query on the above database:

```
SELECT S.sname
FROM Suppliers S
WHERE S.sid NOT IN (SELECT C.sid
          FROM Catalog C
          WHERE C.pid NOT IN (SELECT P.pid
                  FROM Parts P
                  WHERE P.color <> ‘blue’))
```

Assume that relations corresponding to the above schema are not empty. Which one of the following is the correct interpretation of the above query?

(a) Find the names of all suppliers who have supplied a non-blue part.
(b) Find the names of all suppliers who have not supplied a non-blue part.
(c) Find the names of all suppliers who have supplied only blue parts.
(d) Find the names of all suppliers who have not supplied only blue parts.

(GATE 2009: 2 Marks)

Solution: The sub-query “`SELECT P.pid FROM Parts P WHERE P.color <> ‘blue’`” returns pids of parts, which are not blue.

The sub-query “`SELECT C.sid FROM Catalog C WHERE C.pid NOT IN (SELECT P.pid FROM Parts P WHERE P.color <> ‘blue’)`” returns sid of all those suppliers who have supplied blue parts.

The complete query returns the names of all suppliers who have supplied a non-blue part.

Inner query `SELECT P. pid FROM Parts P WHERE P. color <> ‘blue’` will select pid of non-blue parts. Outer query of this will produce sid of those suppliers whose pid is not in the result of inner query. The query will result the supplier names who have supplied non-blue parts

Ans. (a)

41. Assume that, in the suppliers relation above, each supplier and each street within a city has a unique name, and (sname, city) forms a candidate key. No other functional dependencies are implied other than those implied by primary and candidate keys. Which one of the following is TRUE about the above schema?

(a) The schema is in BCNF.
(b) The schema is in 3NF but not in BCNF.
(c) The schema is in 2NF but not in 3NF.
(d) The schema is not in 2NF.

(GATE 2009: 2 Marks)

Solution: Candidate key: sname, city
Primary key: sname
Alternate key: sid, sname

city → street
sname → city
Transitivity: sname → city → street
So, it is in 3NF.

Ans. (b)

42. Consider a B+ tree in which the maximum number of keys in a node is 5. What is the minimum number of keys in any non-root node?

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

(GATE 2010: 1 Mark)

Solution: In B+ tree, root node has minimum two block pointers and maximum p block pointer, where p = order and key = order − 1.

In the non-root node, the minimum number of keys $= p/2 - 1$

Here, key = 5, order = 6

So, minimum number of keys in non-root node $= 6/2 - 1 = 2$

Ans. (b)

43. A relational schema for a train reservation database is given below:

```
Passenger(pid, pname, age)
Reservation(pid, class, tid)
```

Table: Passenger

pid	pname	Age
0	'Sachin'	65
1	'Rahul'	66
2	'Saurav'	67
3	'Anil'	69

Table: Reservation

pid	Class	Tid
0	'AC'	8200
1	'AC'	8201
2	'SC'	8201
5	'AC'	8203
1	'SC'	8204
3	'AC'	8202

What pids are returned by the following SQL query for the above instance of the tables?

```
SELECT pid
FROM Reservation WHERE class = 'AC' AND
EXISTS(SELECT * FROM Passenger
  WHERE age > 65 AND Passenger.pid =
        Reservation.pid)
```

(a) 1, 0 (b) 1, 2
(c) 1, 3 (d) 1, 5

(GATE 2010: 1 Marks)

Solution: The outer query selects four entries (with pid as 0, 1, 5, 3) from Reservation table. Out of these selected entries, the sub-query returns non-null values only for 1 and 3.

Ans. (c)

44. Which of the following concurrency control protocols ensure both conflict serializability and freedom from deadlock?

I. Two-phase locking
II. Timestamp ordering

(a) I only (b) II only
(c) Both I and II (d) Neither I nor II

(GATE 2010: 1 Mark)

Solution: Two-phase locking is a concurrency control method, which guarantees serializability. The protocol utilizes locks, applied by a transaction to data, which may block other transactions from accessing the same data during the transaction's life. 2PL may be lead to deadlocks which result from the mutual blocking of two or more transactions.

Timestamp ordering algorithm is a non-lock concurrency control method. In Timestamp based method, deadlock cannot occur as no transaction ever waits.

Ans. (b)

45. Consider the following schedule for transactions T_1, T_2 and T_3:

T_1	T_2	T_3
Read(X)		
	Read(Y)	
		Read(Y)
	Write(Y)	
Write(X)		
		Write(X)
	Read(X)	
	Write(X)	

Which one of the schedules below is the correct serialization of the above?

(a) $T_1 \rightarrow T_3 \rightarrow T_2$ (b) $T_2 \rightarrow T_1 \rightarrow T_3$
(c) $T_2 \rightarrow T_3 \rightarrow T_1$ (d) $T_3 \rightarrow T_1 \rightarrow T_2$

(GATE 2010: 2 Marks)

Solution: T_1 can complete before T_2 and T_3 as there is no conflict between Write(X) of T_1 and the operations in T_2 and T_3 which occur before Write(X) of T_1.

T_3 can complete before T_2 as the Read(Y) of T_3 does not conflict with Read(Y) of T_2. Similarly, Write(X) of T_3 does not conflict with Read(Y) and Write(Y) operations of T_2.

After topologically sorting, the sequence is $T_1 \rightarrow T_3 \rightarrow T_2$.

Ans. (a)

46. The following functional dependencies hold for relations $R(A, B, C)$ and $S(B, D, E)$

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

The relation R contains 200 tuples and the relation S contains 100 tuples. What is the maximum number of tuples possible in the natural join $R \bowtie S$?

(a) 100 (b) 200
(c) 300 (d) 2000

(GATE 2010: 2 Marks)

Solution: B is a candidate key of R.

So, all 200 values of B must be unique in R.

There is no functional dependency given for S.

For the maximum number of tuples in output, there are two ways.

(i) All 100 values of B in S are same and there is an entry in R, which matches with this value. So, we are having 100 tuples in output.
(ii) All 100 values of B in S are different and these values are present in R also. So, we are having 100 tuples.

Ans. (a)

47. Consider a relational table with a single record for each registered student with the following attributes:

I. Registration_Number: Unique registration number for each registered student.
II. UID: Unique Identity number, unique at the national level for each citizen.
III. BankAccount_Number: Unique account number at the bank. A student can have multiple accounts or joint accounts. This attribute stores the primary account number.
IV. Name: Name of the Student.
V. Hostel_Room: Room number of the hostel.

Which of the following options is INCORRECT?

(a) BankAccount_Number is a candidate key.
(b) Registration_Number can be a primary key.
(c) UID is a candidate key if all students are from the same country.
(d) If S is a superkey such that $S \cap$ UID is NULL then $S \cup$ UID is also a superkey.

(GATE 2011: 1 Mark)

Solution: When two students hold joint account in that case BankAccount_Num will not uniquely determine other attributes.

Ans. (a)

48. Database table by name Loan_Records is given below:

Borrower	Bank_Manager	Loan_Amount
Ramesh	Sunderajan	10000.00
Suresh	Ramgopal	5000.00
Mahesh	Sunderajan	7000.00

What is the output of the following SQL query?

```
SELECT count(*) FROM(
   (SELECT Borrower, Bank_Manager FROM
   Loan_Records) AS S NATURAL JOIN
   (SELECT Bank_Manager, Loan_Amount
   FROM Loan_Records) AS T );
```

(a) 3 (b) 9
(c) 5 (d) 6

(GATE 2011: 2 Marks)

Solution: Output as Table S:

Borrower	Bank_Manager
Ramesh	Sunderajan
Suresh	Ramgopal
Mahesh	Sunderajan

Output as Table T:

Bank_Manager	Loan_Amount
Sunderajan	10000.00
Ramgopal	5000.00
Sunderajan	7000.00

Natural Join of S and T:

Borrower	Bank_Manager	Loan_Amount
Ramesh	Sunderajan	10000.00
Ramesh	Sunderajan	7000.00
Suresh	Ramgopal	5000.00
Mahesh	Sunderajan	10000.00
Mahesh	Sunderajan	7000.00

Ans. (c)

49. Consider a database table T containing two columns X and Y each of type integer.

After the creation of the table, one record ($X = 1$, $Y = l$) is inserted in the table.

Let MX and MY denote the respective maximum values of X and Y among all records in the table at any point in time. Using MX and MY, new records are inserted in the table 128 times with X and Y values being $MX + 1$, $2 * MY + 1$, respectively. It may be noted that each time after the insertion, values of MX and MY change.

What will be the output of the following SQL query after the steps mentioned above are carried out?

```
SELECT Y FROM T WHERE X = 7;
```

(a) 127 (b) 255
(c) 129 (d) 257

(GATE 2011: 2 Marks)

Solution:
The value of X is calculated using MX + 1 and Y is calculated using 2 × MY + 1

For example, X = 1 and Y = 1

at the next step, X = MX + 1 = 1 + 1 = 2, Y = 2 × MY + 1 = 2 × 1 + 1 = 3

X	*Y*
1	1
2	3
3	7
4	15
5	31
6	63
7	127

Ans. (a)

50. Which of the following statements are **TRUE** about an SQL query?

I. An SQL query can contain a HAVING clause even if it does not have a GROUP BY clause.
II. An SQL query can contain a HAVING clause only if it has GROUP BY clause.
III. All attributes used in the GROUP BY clause must appear in the SELECT clause.
IV. Not all attributes used in the GROUP BY clause need to appear in the SELECT clause.

(a) I and III (b) I and IV
(c) II and III (d) II and IV

(GATE 2012: 1 Mark)

Solution:

I. HAVING clause can also be used with aggregate function. If we use a HAVING clause without a GROUP BY clause, the HAVING condition applies to all rows that satisfy the search condition.
II. To verify S,

```
CREATE TABLE temp (id INT, name
VARCHAR(100));
INSERT INTO temp VALUES (1, "abc");
INSERT INTO temp VALUES (2, "abc");
INSERT INTO temp VALUES (3, "bcd");
INSERT INTO temp VALUES (4, "cde");
SELECT Count(*) FROM temp GROUP BY name;
```

Result:

```
count(*)
2
1
1
```

Ans. (b)

51. Given the basic ER and relational models, which of the following is **INCORRECT**?

(a) An attribute of an entity can have more than one value.
(b) An attribute of an entity can be composite.
(c) In a row of a relational table, an attribute can have more than one value.
(d) In a row of a relational table, an attribute can have exactly one value or a NULL value.

(GATE 2012: 1 Mark)

Solution: In relational table, more than one value for an attribute will violate the condition of first normal form. So, option (c) is incorrect.

Ans. (c)

52. Which of the following is **TRUE**?

(a) Every relation is 3NF is also in BCNF.
(b) A relation R is in 3NF if every non-prime attribute of R is fully functionally dependent on every key of R.
(c) Every relation in BCNF is also in 3NF.
(d) No relation can be in both BCNF and 3NF. (GATE 2012, 1 mark)

Solution: Any relation that is in 'x'NF, will automatically be in 'x-1' NF. Now, let us see the order of NF as we discussed in the text, it is – 1NF, 2NF, 3NF, BCNF, 4NF and 5NF. Therefore, option (c) is correct.

Ans. (c)

53. Suppose $R_1(\underline{A}, B)$ and $R_2(\underline{C}, D)$ are two relation schemas. Let r_1 and r_2 be the corresponding relation instances. B is a foreign key that refers to C in R_2. If data in r_1 and r_2 satisfy referential integrity constrains, which of the following is **ALWAYS TRUE**?

(a) $\Pi_B(r_1) - \Pi_C(r_2) = \varnothing$
(b) $\Pi_C(r_2) - \Pi_B(r_1) = \varnothing$
(c) $\Pi_B(r_1) - \Pi_C(r_2)$
(d) $\Pi_B(r_1) - \Pi_C(r_2) \neq \varnothing$

(GATE 2012: 2 Marks)

Solution: B is a foreign key in r_1, which refers to C in r_2.

r_1 and r_2 satisfy referential integrity constraints.

So, every value that exists in column B of r_1 must also exist in column C of r_2.

Ans. (a)

54. Consider the following transactions with data items P and Q initialized to zero:

```
T1: read (P);
    read (Q);
    if P=0 then Q:=Q+1;
    write (Q).
T2: read (Q);
    read (P)
    if Q=0 then P:=P+1;
    write (P).
```

Any non-serial interleaving of T_1 and T_2 for concurrent execution leads to

(a) a serializable schedule
(b) a schedule that is not conflict-serializable
(c) a conflict-serializable schedule
(d) a schedule for which precedence graph cannot be drawn

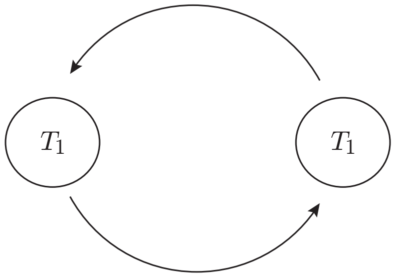
(GATE 2012: 2 Marks)

Solution: Two schedules are conflict-serializable if the actions belong to different transactions, one of the actions is a write operation, the actions access the same object (read or write).

Two schedules are conflict-equivalent if both schedules involve the same set of transactions, and the order of each pair of conflicting actions in both schedules is the same.

Take the following schedule:

S: $r_1(P)$; $r_1(Q)$; $r_2(Q)$; $r_2(P)$; $w_1(Q)$;$w_2(P)$



Cycle exists between two transactions, so T_1 and T_2 are not conflict serializable.

A schedule is conflict-serializable when the schedule is conflict-equivalent to one or more serial schedules. There are two possible serial schedules—T_1 followed by T_2 and T_2 followed by T_1. In both, one of the transactions reads the value written by another transaction as a first step.

Ans. (b)

Common Data Questions 55 and 56: Consider the following relations A, B and C:

A

Id	Name	Age
12	Arun	60
15	Shreya	24
99	Rohit	11

B

Id	Name	Age
15	Shreya	24
25	Hari	40
98	Rohit	20
99	Rohit	11

C

Id	Phone	Area
10	220	02
99	2100	01

55. How many tuples does the result of the following relational algebra expression contain? Assume that the schema of $A \cup B$ is the same as that of A.

$$(A \cup B) \bowtie_{\text{A.Id}>40 \bowtie \text{ C.Id}<15} \text{C}$$

(a) 7 (b) 4
(c) 5 (d) 9

(GATE 2012: 2 Marks)

Solution:

$A \cup B$

Id	Name	Age
12	Arun	60
15	Shreya	24
99	Rohit	11
25	Hari	40
98	Rohit	20

$(A \cup B) \bowtie_{\text{A.Id}>40 \bowtie \text{ C.Id}<15} \text{C}$

Id	Name	Age	Id	Phone	Area
12	Arun	60	10	2200	02
15	Shreya	24	10	2200	02
99	Rohit	11	10	2200	02
25	Hari	40	10	2200	02
98	Rohit	20	10	2200	02
99	Rohit	11	99	2100	01
98	Rohit	20	99	2100	01

Ans. (a)

56. How many tuples does the result of the following SQL query contain?

```
SELECT A.Id
FROM A
WHERE A.Age > ALL (SELECT B.Age
                   FROM B
                   WHERE B.Name = 'Arun')
```

(a) 4 (b) 3 (c) 0 (d) 1

(GATE 2012: 2 Marks)

Solution: ALL returns the values to be greater than all the values returned by the sub-query.

There is no entry with the name Arun in Table B. So, the sub-query will return NULL. If a sub-query returns NULL, then the condition becomes true for all rows of A.

Ans. (b)

57. An index is clustered, if

(a) It is on a set of fields that form a candidate key.
(b) It is on a set of fields that include the primary key.
(c) The data records of the file are organized in the same order as the data entries of the index.
(d) The data records of the file are organized not in the same order as the data entries of the index.

(GATE 2013: 1 Mark)

Solution: In clustered index, the leaf nodes contain the data page of the table concerned. Each index row contains a key value and a pointer to either the intermediate level (other than root and leaf nodes) or data row in leaf level. Therefore, they are organised in the same order.

Ans. (c)

58. Consider the following relational schema:

```
Students(rollno: integer, sname: string)
Courses(courseno: integer, cname: string)
Registration(rollno: integer, courseno;
integer, percent: real)
```

Which of the following queries are equivalent to this query in English?

"Find the distinct names of all students who score more than 90% in the course numbered 107"

I. `SELECT DISTINCT S.sname FROM Students as S, Registration as R WHERE R.rollno=S.rollno AND R.Courseno=107 AND R.percent>90`

II. $\Pi_{\text{sname}}(\sigma_{\text{courseno}=107} \Lambda \, \text{percent} >$
$90(\text{Registration students}))$

III. $\{T | \exists S \in \text{ Student}, \exists \text{R} \in \text{ Registration } (\text{S.rollno} = \text{R.rollno} \wedge \text{R.courseno} = 107 \wedge R.\text{percent} > 90 \wedge T. \text{ sname} = S.\text{name})\}$

$\{<\text{S}_N> | \exists S_R \exists R_P(<S_R, S_N> \in \text{ Student} \wedge < S_R, 107, R_P> \in R_P > 90)\}$

(a) I, II, III and IV (b) I, II and III only
(c) I, II and IV only (d) II, III and IV only

(GATE 2013: 2 Marks)

Solution: All of the query statements are equivalent to the given query.

Ans. (a)

Linked Answer Questions 59 and 60: Relation R has eight attributes $ABCDEFGH$. Fields of R contain only atomic values. $F = \{CH \rightarrow G, A \rightarrow BC, B \rightarrow CFH, E \rightarrow A, F \rightarrow EG\}$ is a set of functional dependencies (FDs) so that $F+$ is exactly the set of FDs that hold for R.

59. How many candidate keys does the relation R have?

(a) 3 (b) 4
(c) 5 (d) 6

(GATE 2013: 2 Marks)

Solution: Candidate keys AD, BD, ED and FD

Ans. (b)

60. The relation R is

(a) In 1NF, but not in 2NF
(b) In 2NF, but not in 3NF
(c) In 3NF, but not in BCNF
(d) In BCNF

(GATE 2013: 2 Marks)

Solution: $A \rightarrow BC$, $B \rightarrow CFH$ and $F \rightarrow EG$ are partial dependencies.

So, they are in 1NF but not in 2NF.

Ans. (a)

PRACTICE EXERCISES

Set 1

1. A relational database can be defined as

(a) a place to store relational information.
(b) a database that relates to different operations.
(c) a database to relate the relationship of different databases.
(d) a database related to storing of information about humans.

2. The minimal superkeys are called

(a) minimum keys. (b) candidate keys.
(c) domain keys. (d) optimal keys.

3. Which of the following combinations represents a valid data model?

(a) E-R model, network data model, normal form data model
(b) Relational data model, hierarchical data model, SQL data model
(c) Object-based data model, table data model
(d) E-R model, relational data model, object-based data model

4. Which of the following is not a database model?

(a) Network database model
(b) Relational database model
(c) Object-oriented database model
(d) Normal form data model

5. Which one of the following allows accessing or maintaining data in a database?

(a) DCL (b) DML
(c) DDL (d) None of the above

6. Which of the following is a binary operation?

(a) Select (b) Project
(c) Rename (d) Union

7. Which of the following is a unary operation?

(a) Join (b) Project
(c) Intersection (d) Cartesian product

8. What is the least and ceiling number of rows in the join selection of two tables having m rows and n rows, respectively?

(a) 1 and $m \times n$ (b) 0 and $m + n$
(c) 1 and infinite (d) 0 and mn

9. An attribute of one table having the same value of a primary key of another table is termed as

(a) foreign key. (b) alternate key.
(c) candidate key. (d) transitive key.

10. A database has five attributes defined as $\{A, B, C, D, E\}$. The functional dependency is defined by

$\{A \rightarrow BC, E \rightarrow FG\}$

The candidate key of this table is

(a) A. (b) AE.
(c) E. (d) BC and FG.

11. Consider the following set of functional dependency

$\{AB \rightarrow CD, CD \rightarrow E, E \rightarrow A\}$ AB, CD are candidate keys.

It is in which normal form?

(a) 2NF (b) BCNF (c) 3NF (d) 4NF

12. The term physical data independence can be defined as

(a) If we modify the conceptual schema, then there is no need to modify the physical schema.
(b) If we modify the physical schema, then there is no need to modify the conceptual schema.

(c) If we modify the conceptual schema, then there is no need to modify the external schema.
(d) If we modify the middle level schema, then any one schema (internal/external) may be modified.

13. Relational algebra can be defined as

(a) Procedural. (b) Non-procedural.
(c) User dependent. (d) Domain dependent.

14. Which of the following is true?

(a) A relation in BCNF is always in 3NF.
(b) A relation in 3NF is always in BCNF.
(c) PJNF and 3NF are same.
(d) A relation in PJNF is not in 3NF.

15. RAD stands for ________.

(a) Read and destroy
(b) Read and develop
(c) Rapid application development
(d) Rapid application design

16. In DML, RECONNCT command cannot be used with

(a) CLEAN set. (b) FIXED set.
(c) OPTIONAL set. (d) USER set.

17. The UWA is used to communicate the contents of individual records between

(a) Host program and present record.
(b) Host program and host record.
(c) Host program and DBMS.
(d) Host program and host language.

18. Referential integrity is concerned with

(a) Primary key. (b) Foreign key.
(c) Alternate key. (d) Project_Join key.

19. Match the following:

List – I	**List – II**
I. DDL	A. LOCK TABLE
II. DML	B. COMMIT
III. TCL	C. Natural Difference
IV. BINARY Operation	D. REVOKE

Codes:

	I	II	III	IV
(a)	A	D	B	C
(b)	A	B	D	C
(c)	D	B	A	C
(d)	D	A	B	C

20. The value of particular field should be less than 50 if it is

(a) An integrity constraint.
(b) A referential constraint.
(c) A primary key constraint.
(d) A normalization constraint.

21. The ER model of a database includes

I. Different entity types.
II. Attributes for each entity type.
III. Relationships among entity types.
IV. Semantic integrity constraints are not defined.

Which of the above statements is correct?

(a) All of the above (b) Only I, II and III
(c) III and IV (d) II and III

22. The cardinality ratio and participation ________ of a table helps us to implement either the cross referencing design or mutual referencing design.

(a) Functionality (b) Constraints
(c) SQL queries (d) Domains

23. Consider the following schemas:

```
Branch = (Branch-name, Assets, Branch-city)
Customer = (Customer-name, Bank name,
            Customer-city)
Borrow = (Branch-name, loan number,
          customer account-number)
Deposit = (Branch-name, Accountnumber,
           Customer-name, Balance)
```

Using relational algebra, the query that finds customers having deposits have accountnumber > 23456 is

(a) $\Pi_{\text{customer-name}} (\sigma_{\text{account-number}} > 23456 \text{ (Deposit)}$
(b) $\sigma_{\text{customer-name}} (\sigma_{\text{account-number}} > 23456 \text{ (Deposit)}$
(c) $\Pi_{\text{customer-name}} (\sigma_{\text{account-number}} > 23456 \text{ (Borrow)}$
(d) $\sigma_{\text{customer-name}} (\Pi_{\text{account-number}} > 23456 \text{ (Borrow)}$

24. What deletes the entire file except the file structure?

(a) DELETE
(b) DELETE RECORDS
(c) ZAP
(d) RETAIN STRUCT

25. The function of a Transaction Manager is to

I. Maintain a log of transactions.
II. Maintain database images before and after a transaction.
III. Maintain appropriate concurrency control.
IV. Avoid deadlocks.

(a) I, II, III and IV (b) II and III
(c) I, III and IV (d) I, II and III

26. The term granularity relates to

(a) the size of table.
(b) the size of data item.
(c) the size of tuple.
(d) the size of primary key

27. Match the following:

I. OLAP	A. Back propagation
II. OLTP	B. Data warehouse
III. Decision tree	C. RDBMS
IV. Neural network	D. Classification

	I	**II**	**III**	**IV**
(a)	D	C	A	B
(b)	B	C	D	A
(c)	A	B	C	D
(d)	D	B	A	C

28. Which of the following statements is TRUE?

(a) A prime attribute can be transitively dependent on a key in 5NF relation.
(b) A prime attribute can be transitively dependent on a key in 3NF relation.
(c) A prime attribute can be transitively dependent on a key in PJNF relation.
(d) A prime attribute cannot be transitively dependent on a key in BCNF relation.

29. Which normal form is considered as adequate for a simple database design?

(a) 2NF (b) 3NF
(c) BCNF (d) PJNF

30. Which of the following is not a type of DBMS?

(a) Hierarchical (b) Network
(c) Relational (d) Graphical

31. Manager's salary details are to be hidden from Employee Table. This technique is called as

(a) internal level datahiding.
(b) physical level datahiding.
(c) external level datahiding.
(d) logical level datahiding.

32. A network schema is used to

(a) restrict to many-to-many relationship.
(b) permit many-to-many relationship.
(c) permit to store data in a database.
(d) help in storing one-to-many relationship.

33. Storing under unified scheme at a single site is called as

(a) data mining. (b) universal database.
(c) data warehousing. (d) advanced database.

34. The task of correcting and cleaning data is called as

(a) data organization. (b) pre-processing.
(c) data mining. (d) data structuring.

35. A clustering index consists of __________.

(a) declared and ordered primary key
(b) both grouped and ordered primary key and foreign key
(c) ordered foreign key
(d) grouped and ordered candidate key and alternate key

36. Consider the following schema:

```
Emp(Empcode, Name, Sex, Salary, Deptt)
```

A simple SQL query is executed as follows:

```
SELECT Deptt FROM Emp
WHERE sex = 'M'
GROUP by Dept
HAVING avg (Salary) > {select avg (Salary)
from Emp}
```

The output will be

(a) average salary of male employee is the average salary of the organization.
(b) average salary of male employee is less than the average salary of the organization.
(c) average salary of male employee is equal to the average salary of the organization.
(d) average salary of male employees in a department is more than the average salary of the organization.

37. Consider a relation $X(p, q, r)$, and the domains are represented by their atomic values. Further, the functional dependency $p \rightarrow q$, $q \rightarrow r$ is observed, the relation is in

(a) 1NF, not in 2NF
(b) 2NF, not in 3NF
(c) 3NF
(d) 1NF, not in 3NF

38. Match the following:

I. Foreign keys	A. Domain constraint
II. Private key	B. Referential integrity
III. Event control action model	C. Password
IV. Data security	D. Trigger

Codes:

	I	II	III	IV
(a)	C	B	A	D
(b)	B	A	D	C
(c)	C	D	A	B
(d)	A	B	C	D

Set 2

1. For a database relation $R(a, b, c, d)$, where the domains of a, b, c, d include only atomic values, only the following functional dependencies and those that can be inferred from them holds.

$a \rightarrow c$

$b \rightarrow d$

This relation is

(a) In First normal form but not in Second normal form.
(b) In Second normal form but not in Third normal form.
(c) In Third normal form.
(d) None of the above.

2. Let $R(a, b, c)$ and $S(d, e, f)$ be two relations in which d is the foreign key of S that refers to the primary key of R. Consider the following four operations on R and S:

I. Insert into R II. Insert into S
III. Delete from R IV. Delete from S

Which of the following is true about the referential integrity constraint above?

(a) None of I, II and IV can cause its violation.
(b) All of I, II, III and IV can cause its violation.
(c) Both I and IV can cause its violation.
(d) Both II and III can cause its violation.

3. There are five records in a database

Name	Age	Occupation	Category
Rama	27	CON	A
Abdul	22	ENG	A
Jenifer	28	DOC	B
Maya	32	SER	D
Dev	24	MUS	C

There is an index file associated with this and it contains the values 1, 3, 2, 5 and 4. Which one of the fields in the index built from?

(a) Age (b) Name
(c) Occupation (d) Category

4. Consider the following database relations containing the attributes

`Book_id`

`Subject_Category_of_books`

`Name_of_Author`

`Nationalilty_of_Author`

With Book_id as the primary key

(a) The highest normal form satisfied by this relation is — NF.
(b) Suppose the attributes Book_title and Author_address are added to the relation, and the primary key is changed to {name_of_Author, Book_title}, the highest normal form satisfied by the relation is — NF.

5. Consider the scheme $R = (S\ T\ U\ V)$ and the dependencies $S \rightarrow T$, $T \rightarrow U$, $U \rightarrow V$ and $V \rightarrow S$. Let $R = (R_1 \text{ and } R_2)$ be a decomposition such that $R_1 \cap R_2 = \phi$. The decomposition is

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

6. Consider a schema $R(A, B, C, D)$ and functional dependencies $A \rightarrow B$ and $C \rightarrow D$. Then the decomposition of R into $R_1(AB)$ and $R_2(CD)$ is

(a) Dependency preserving and lossless join.
(b) Lossless join but not dependency preserving.
(c) Dependency preserving but not lossless join.
(d) Not dependency preserving and not lossless join.

7. Relation R with an associated set of functional dependencies, F, is decomposed into BCNF. The redundancy (arising out of functional dependencies) in the resulting set of relation is

(a) Zero.
(b) More than zero but less than that of an equivalent 3NF decomposition.
(c) Proportional to the size of $F+$.
(d) Indeterminate.

8. From the following instance of a relation scheme $R(A, B, C)$, we can conclude that

A	*B*	*C*
1	1	1
1	1	0
2	3	2
2	3	2

(a) A functionally determines B and B functionally determines C.
(b) A functionally determines B and B does not functionally determine C.
(c) B does not functionally determine C.
(d) A does not functionally determine B and B does not functionally determine C.

9. Which of the following is/are correct?

(a) An SQL query automatically eliminates duplicates.
(b) An SQL query will not work if there are no indexes on the relations.
(c) SQL permits attribute names to be repeated in the same relation.
(d) None of the above.

10. Given relation $r(w, x)$ and $s(y, z)$, the result of select distinct w, x from r, s is guaranteed to be same as r, provided

(a) r has no duplicates and s is non-empty.
(b) r and s have no duplicates.
(c) s has no duplicates and r is non-empty.
(d) r and s have the same number of tuples.

11. Which of the following query transformations (i.e. replacing the LHS expression by the RHS expression) is incorrect? R_1 and R_2 are relations, C_1 and C_2 are selection conditions and A_1 and A_2 are attributes of R_1.

(a) $\sigma_{c_1}(\sigma_{c_2}(R_1)) \rightarrow \sigma_{c_2}(\sigma_{c_1}(R_1))$
(b) $\sigma_{c_1}(\Pi_{A_1}(R_1)) \rightarrow \Pi_{A_1}(\sigma_{c_1}(R_1))$
(c) $\sigma_{c_1}(R_1 \mathrm{E} R_2) \rightarrow \sigma_{c_1}(R_1) \mathrm{E} \sigma_{c_1}(R_1)$
(d) $\Pi_{A_2}(\sigma_{c_1}(R_1)) \rightarrow \sigma_{c_1}(\Pi_{A_2}(R_1))$

12. The relational algebra expression equivalent to the following tuple calculus expression:

$\{t | t \in r \wedge (t[A] = 10 \wedge t[B] = 20)\}$ is

(a) $\sigma_{(A = 10 \vee B = 20)}(r)$
(b) $\sigma_{(A = 10}(r)\ \mathrm{E}\ \sigma_{B = 20}(r)$
(c) $\sigma_{(A = 10}(r) \cap \sigma_{B = 20}(r)$
(d) $\sigma_{(A = 10}(r) - \sigma_{B = 20}(r)$

13. Which of the following relational calculus expressions is not safe?

(a) $\{t | \exists \in R_1(t[A] = u[A]) \wedge \neg \exists s \in R_2(t[A] = s[A])\}$
(b) $\{t | \forall u \in R_1(u[A] = \text{“}x\text{”} \Rightarrow \exists s \in R_2\ (t[A] = s[A] \wedge s[A] = u[A]))\}$
(c) $\{t | \neg\ (t \in R_1)$
(d) $\{t | \exists \in R_1(t[A] = u[A]) \wedge \exists s \in R_2(t[A] = s[A])\}$

14. With regard to the expressive power of the formal relational query languages, which of the following statement is true?

(a) Relational algebra is more powerful than relational calculus.
(b) Relational algebra has the same power as relational calculus.
(c) Relational algebra has the same power as safe relational calculus.
(d) None of the above.

15. For the schedule given below, which of the following is correct?

1 Read A
2 Read B
3 Write A
4 Read A
5 Write A
6 Write B
7 Read B
8 Write B

(a) This schedule is serializable and can occur in a scheme using 2PL protocol.
(b) This schedule is serialzable but cannot occur in a scheme using 2PL protocol.
(c) This schedule is not serializable but can occur in a scheme using 2PL protocol.
(d) This schedule is not serializable and cannot occur in a scheme using 2PL protocol.

16. Which of the following is correct?

(a) B trees are for storing data on disk and B+ trees are for main memory.
(b) Range queries are faster on B+ trees.
(c) B trees are for primary indexes and B+ trees are for secondary indexes.
(d) The height of a B+ tree is independent of the number of records.

17. B+ trees are preferred to binary trees in database because

(a) disk capacities are greater than memory capacities.

(b) disk access is much slower than memory access.
(c) disk data transfer rates are much less than memory data transfer rates.
(d) disks are more reliable than memory.

18. A B+ tree index is to be built on the name attribute of the relation STUDENT. Assume that all student names are of length 8 bytes, disk blocks are of size 512 bytes and index pointers are of size 4 bytes. Given this scenario, what would be the best choice of the degree (i.e., the number of pointers per node of the B+ tree?

ANSWERS TO PRACTICE EXERCISES

Set 1

1. (a)
2. (b)
3. (d)
4. (d)
5. (a)
6. (d)
7. (b)
8. (d)
9. (a)
10. (b)
11. (c)
12. (b)
13. (a)
14. (a)
15. (c)
16. (b)
17. (c)
18. (b)
19. (d)
20. (a)
21. (a)
22. (b)
23. (a)
24. (c)
25. (d)
26. (b)
27. (b)
28. (d)
29. (b)
30. (d)
31. (c)
32. (b)
33. (c)
34. (b)
35. (a)
36. (d)
37. (a)
38. (b)

Set 2

1. (a) The relation has the following functional dependencies:
$a \rightarrow c$
$b \rightarrow d$

$\{a\ b\}$ is key for the given relation.

Here prime attributes are a and b.

c and d are dependent on partial keys, this is not allowed in 2NF. So, the given relation is in 1 NF.

2. (d) Insertion into S and deletion from R can cause inconsistency. So, II and III can cause violation.

3. (c) The index is built from field occupation where column is sorted in alphabetic order CON, DOC, ENG, SER and MUS. Index is assigned according to alphabetic position.

4. (a) 2NF, (b) 3NF

5. (d) The relation schema $R = (S\ T\ U\ V)$ has following functional dependencies

$S \rightarrow T$

$T \rightarrow U$

$U \rightarrow V$

$V \rightarrow S$

Candidate keys for this relation are $\{S,\ T,\ U,\ V\}$
As given $R_1 \cap R_2 = \phi$
So, R_1 and R_2 might have two attributes. A relation with two attributes satisfies both 2NF and 3NF conditions.

6. (c) Dependency is preserved because there is no loss of functional dependencies.
Since $R_1(AB)$ and $R_2(CD)$ do not have any common attribute. So, it is lossy join.

7. (a) Redundancy is zero when we decompose into BCNF.

8. (b, c) F: $X \rightarrow Y$ implies that for any two tuples if $t_1[X] = t_2[X]$, they must have $t_1[Y] = t_2[Y]$.

9. (d) DISTINCT key word is used to remove duplicate rows along with SELECT keyword. SQL does not permit attribute names to be repeated in the same relation.

10. (a) Cross join with an empty relation produces the overall result as empty. Therefore, S is not empty. 'r' should not have any duplicates to have same rows as 'r'.

11. (a) $\sigma_{c_1}(\sigma_{c_1}(R))$ and $(\sigma_{c_2}(\sigma_{c_1}(R)))$ will not be equivalent if conditions c_1 and c_2 are not equivalent.

12. (c) Tuple calculus expression select tuples where $A = 10$ and $B = 20$ of relation R. So, the equivalent relational algebra expression is $\sigma_{(A\ =\ 10}(r) \cap \sigma_{B\ =\ 20}(r)$.

13. (c) The query $-t|\neg\ (t \in\ R_1)$ is syntactically correct. But it requests for all tuples that are not in R_1. That set of such t tuples is obviously infinite, in the context of such as the infinite domain set of all integers. Therefore, this is unsafe query.

14. (c) Both relational algebra and relational calculus has same expressive powers. Every query written in relation algebra can be expressed in relational calculus.

15. (d) Given schedule has $W_1(A)$, $R_2(A)$ and $W_2(B)$, $R_1(B)$ conflicting pairs. So the schedule is not serializable. Hence, it cannot occur in scheme using 2PL protocol.

16. (b) Most database systems use indexes built on some form of a B+ tree due to its many advantages, in particular its support for range queries. Leaf nodes are linked together in B+ trees, hence range queries are faster.

17. (b) In B+ trees, disk access is slow and numbers of hits are less. B+ trees have very high fan-out (typically on the order of 100 or more), which reduces the number of I/O operations required to find an element in the tree.

18. (43) Let n be the degree.

Given k key size (length of the name = 8 byte attribute of student)

Disk block size, $B = 512$ bytes

Index pointer size, $b = 4$ bytes

Degree of B+ tree can be calculated if we know the maximum number of key an internal node can have; the formula for that is

$(n - 1)k + n \times\ b =$ Block size

$(n - 1)8 + n \times 4 = 512$

$12n = 520$

$n = 520/12 = 43$