Subject Title: Relational Database Management System **Subject Code:** 12065

1. Attempt any TEN of the following. 20 Marks

a) State any two advantages of data base management system over file processing system.
[1 Mark for each advantage, Any 2 advantages]

Ans: 1. Reduction in Redundancy: Duplication of records is reduced.
2. Avoiding Inconsistency: As the redundancy is reduced inconsistency is avoided.
3. Maintaining Integrity: Accuracy is maintained.
4. Sharing of data: Sharing of data is possible.
5. Enforcement of Security: Security can be enforced.
6. Transaction support.

b) Define the term 'Integrity Constraint'.
[2 Marks for correct definition]

Ans: Integrity constraints are used to ensure accuracy and consistency of data in a relational database.

c) What is 'Authorization'?
[2 Marks for correct explanation]

Ans: The authorization is the process of allowing the different privileges to the user to access, add, modify or delete the data.

d) List out any two features of Hierarchical Model.
[1 Mark for each feature. Any 2 features]

Ans: 1. It is a record based model.
2. It is based on Tree like structure.
3. It follows One to Many relationship.

e) What is functional dependencies?
[Explanation 2 Marks]

Ans: In the given relation R, attribute 'Y' of R is functionally dependent on attribute ' 'X' if and only if whenever two tuples of R are having same 'X' value then 'Y' value will be same. It is denoted as X→Y which means 'Y' functionally dependent on 'X'.

OR

A functional dependency occurs when one attribute in a relation uniquely determine another attribute.
e.g. emp_id→ename(ename functionally dependent on emp_id)

f) What is query processing?
[Explanation 2 Marks]

Ans: Query processing involves multiple activities to extract the data from database. This mainly involves the conversion of high level language to low level language like relational algebra, optimization and actual evaluation.

g) What is deadlock?
[Explanation 2 Marks]

Ans: A situation in which all transactions are waiting for each other to release the data item and cannot get executed is known as deadlock.

h) Define the term 'Transaction' in SQL.
[2 Marks for correct definition]

Ans: A transaction can be defined as a unit of program execution that access and updates data items.

OR

A sequence of operations with the database which represents one meaningful activity is called 'Transaction'.

OR

A single logical unit of work which is the collection of several operations is called a Transaction.

i) State any two functions of database administrator.
[1 Marks for each function. Any 2 functions]

Ans: 1. Schema Definition: The DBA creates database schema by executing DDL statements.
2. Storage structure and access method: The storage of the data structure and how to access data from it is handled by DBA.
3. Granting of authorization for data access: The DBA will grant and restrict access to the user.
4. Regular Maintenance of Database: The DBA performs some of the routine maintenance activities like periodic backups, ensuring enough disk space is available.

j) State the meaning of following components of E-R diagram.
[1 Mark for each component]]

Ans: **i) Rectangle:** It represents Entity set

ii) Line: It links attributes to entity sets and entity sets to relationship sets.

iii) Double Ellipse: It represents Multivalued attributes

iv) Diamond: It represents relationship set

k) Define the term 'VIEW'.
[2 Marks for correct definition]

Ans: Views are the temporary relations which are created either for the convenience or to hide some details of that particular relation from other users.

l) State the meaning of weak entity set and strong entity set.
[1 Mark for weak entity and 1 Mark for strong entity]

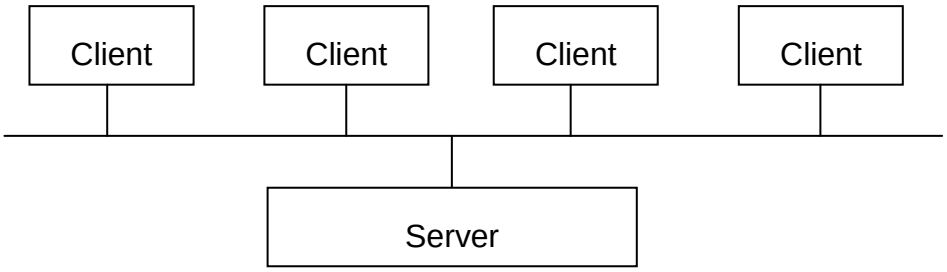
Ans: **Weak Entity:** A Entity set that does not have a Primary key is referred as a weak entity set. **OR**
An entity set that does not have sufficient attribute to form a primary key is called as Weak Entity Set

Strong Entity Set: An entity set that have sufficient attributes to form a primary key is called as strong entity set. **OR**
An entity set that have a primary key of its own is referred as Strong entity set.

2. Attempt any FOUR of the following. 16 Marks

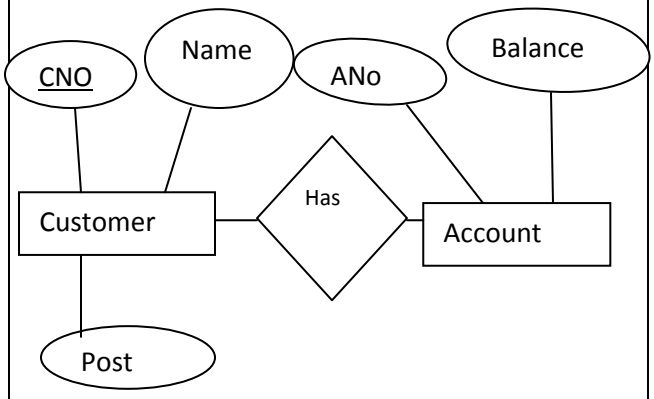
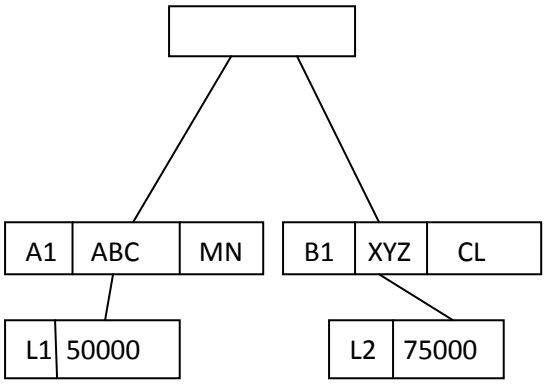
a) State the meaning of client server architecture. State the role of server.
[Meaning of client server architecture 2 Marks, Role of server 1 Mark each]

Ans: Computer networking allows some task to be executed on a server system and some tasks on client system. This leads to development of client server architecture. The clients are the machines which requests for the service to the server. Server is the machine which serves to the clients. There are different types of client/server architecture such as two tier, three tier architecture.

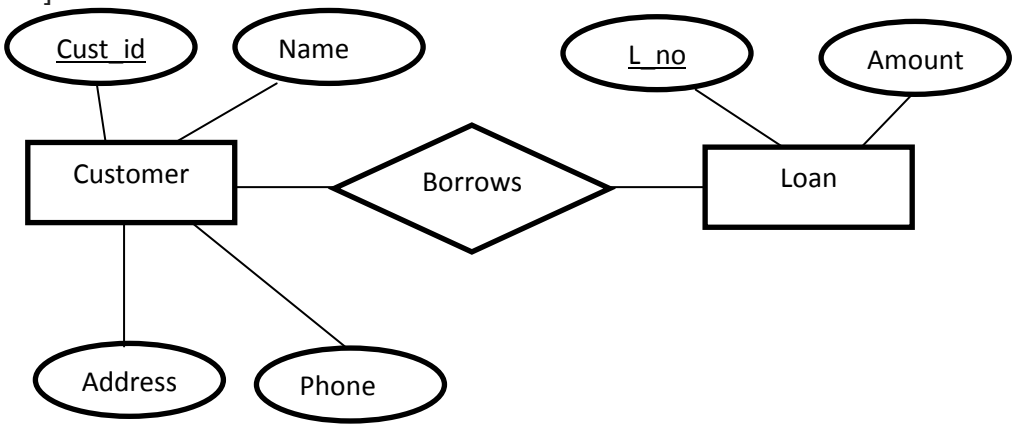


Role of Server: The server is the machine that can provide services to the client machine such as file access, printing, and database access. It is used to manage the database tables optimally among multiple clients who concurrently request the server for the same data.

b) Compare E-R model with Hierarchical model.
[1 Mark for each point, 4 points should be mentioned]

E-R Model	Hierarchical Model
1. It is object based logical model	1.It is record based model
2. Information is represented as E-R diagram having different symbols like rectangle for entity, diamond for relationship set ,ellipse for attributes etc.	2. It is a tree like structure. It consists of a collection of *records* that are connected to each other through *links*.
3. It has many to many relationships.	3. It has one to many relationships.
4. It is used by database designer to communicate the design to the end user.	4. It is mainly used in mainframe computers.
5. 	5. 

c) Sketch E-R diagram for customer and loan. Assume suitable attributes.
[Correct shape 1mark, correct entity 1 Mark, correct attributes 1 Mark, proper relationship set 1 Mark]



d) List out query processing components and state their functions.
[Listing of query processing component 2 Marks, Functions 2 Marks]

Ans: 1. DDL Interpreter: It interprets DDL statements and records them in a set of tables containing metadata or data dictionary.
2. DML Compiler: It translates DML statements of high level query language into low level instructions that query evaluation engine understands.
3. Embedded DML PreCompiler: It converts DML statements embedded in application program to normal procedural calls in host language. The precompiler must interact with the DML compiler to generate the appropriate code.
4. Query Evaluation Engine: It executes low level instructions generated by DML compiler and DDL interpreter.

e) Explain disadvantages file processing over DBMS?
[Any 4 correct disadvantages 1Mark each]

Ans: 1. Data Redundancy and Inconsistency: As each application has its separate file, the same data is duplicated in many files. The duplication is known as redundancy.
Due to this redundancy, data inconsistency can occur which means that changes in the data may be done in one file but not in another.
2. Integrity Problem: Lack of integrity occurs as a result of inconsistency. We cannot assure accuracy in the data due to redundancy and inconsistency.
3. Sharing of data: Sharing of data is not possible.
4. Data Isolation: As the data is scattered in various files and files may be in different format, it is difficult to write any new application program.
5. Difficulty in accessing data: It is difficult to access different data according to different user's requirements.

6. Security Problem: It is difficult to keep security for retrieving particular data.
7. Atomicity: In many applications it is difficult to ensure that once a failure has occurred the data is stored to the consistent state that existed prior to the failure.

f) Explain single valued and multivalued attribute of E-R model. What is null attributes?
[1 ½Mark each for single valued and multivalued attribute, Null attribute 1 Mark]

Ans: Single valued attributes: A attribute that hold a single value is known as single valued attribute. E.g. a person can have only one age, roll_no so these are single valued attributes.

Multivalued attribute: Attribute that hold multiple values are known as multivalued attribute. E.g. a person may have multiple phone numbers, email_ids. So phone_no, email_id are the multivalued attribute.

Null attribute: The attribute value that indicates an unknown or missing value is called as Null attribute.

e.g. If an employee is not assigned any project then the "project_id" attribute will be Null

3. Attempt any **Four** of the following. 16 Marks

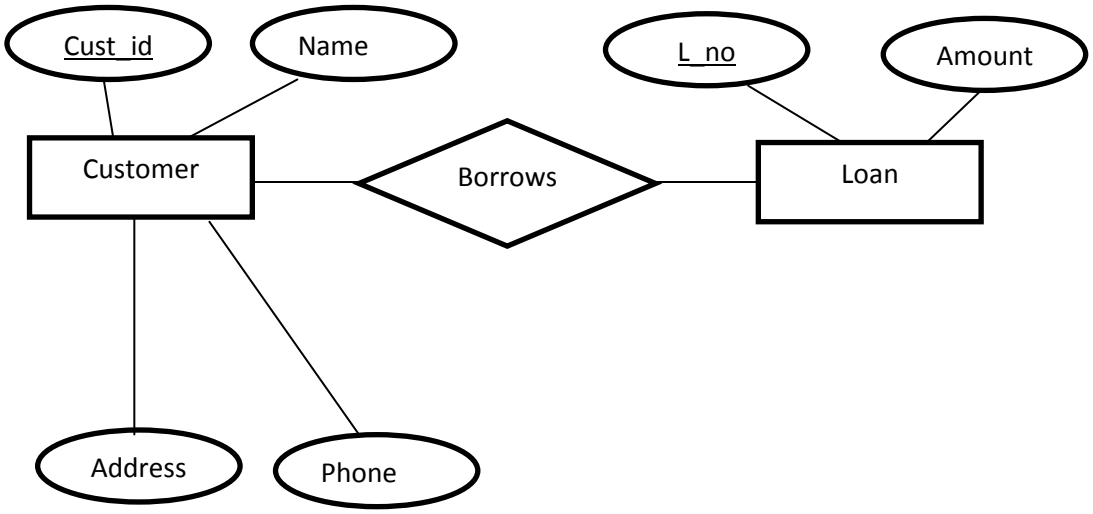
a) State the meaning of relationship set and combined tables of E-R model.
[(Meaning of relationship set 1½ Marks, Diagram 1½ Marks)]
[combined table 1 Mark **(Remark: Any other relevant explanation 1 Mark)**]

Ans: 1) A relationship set is a mathematical relation among $n \geq 2$ entities, each taken from entity sets $\{(e_1, e_2, \ldots e_n) \mid e_1 \in E_1, e_2 \in E_2, \ldots, e_n \in E_n\}$
where (e1, e2, …, en) is a relationship.

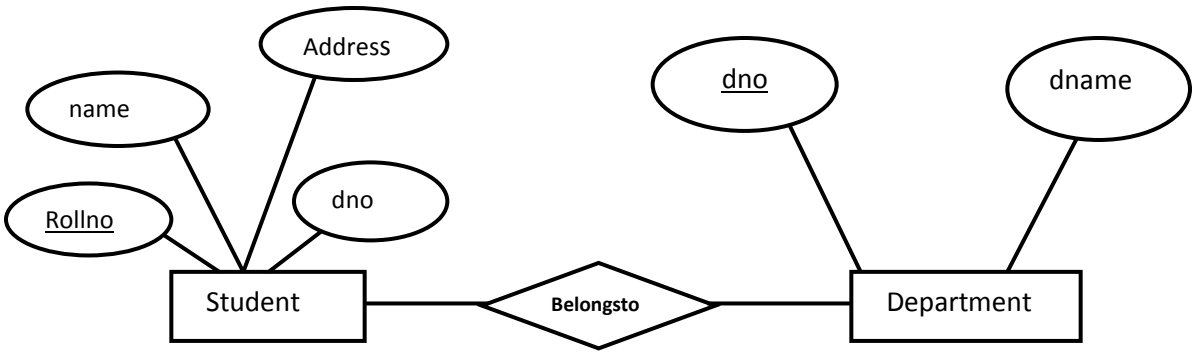
OR

A relationship is an association among several entities.

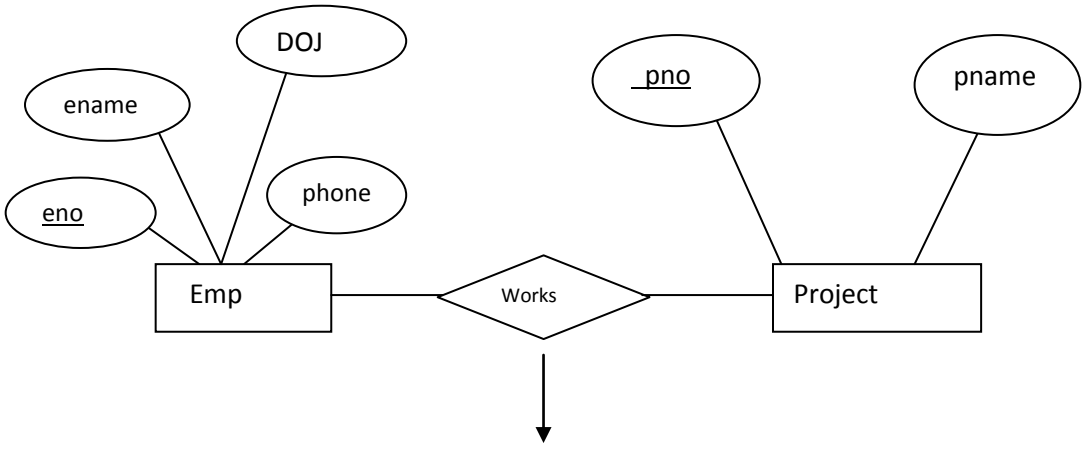
2) Same entity set can participate in different relationship sets.
3) Relationship sets that involve two entity sets are binary (or degree two). Generally, most relationship sets in a database system are binary.
4) e.g. Customer and loan E-R model.



Combined Table:



In the above diagram there are two entities: Student and department
Student{rollno, name, address, dno}
Department {dno, dname}
Dno of student entity is the foreign key which takes the values from dno of department entity. If we want to see the dname of each student then Student and Department entities will be combined by using foreign key(dno).



eno	pno
E1	P1
E2	P2

To design combined table for relationship "works " the attributes eno from emp and pno from project can be used.

b) Explain Tuple Relational Calculus of E-R model
[Tuple Relational Calculus Explanation 2 Marks, Any correct Example 2 Marks]

Ans: The Tuple Relational Calculus:

The tuple relational calculus is a nonprocedural DML statement.
In Tuple relational calculus formal description of the information is desired.
A query in the tuple relational calculus is expressed as

$$\{t \mid P(t)\}$$

i.e. the set of tuples t for which predicate P is true.
We also use the notation
$t[a]$ to indicate the value of tuple t on attribute a.
$t \in r$ to show that tuple t is in relation r.
For example, to find the branch-name, loan number, customer name and amount for loans over \$1200:

$$\{t \mid t \in borrow \wedge t[amount] > 1200\}$$

This gives us all attributes, but supposes we only want the customer names. We need to write an expression for a relation on scheme *(cname)*.

$$\{t \mid \exists s \in borrow\ (t[cname] = s[cname] \wedge s[amount] > 1200)\}$$

c) What is Relational Algebra? Explain select and project operations.
[Relational Algebra Explanation 2 Marks, Select operation 1 Mark, Project operation 1Mark]

Ans: Relational algebra is a set of operations on relational database that allows retrieval of data.

Relational Algebra is a procedural query language. It consists of a set of operations that take one or two relations as input and produces a new relation as their result.

1) SELECT (σ)

SELECT is used to obtain a subset of the tuples of a relation that satisfy a *select condition*.

For example, find all employees having salary more than 10000

$\sigma_{\text{salary} > 10000}(\text{employee})$

2) PROJECT(π)

The PROJECT operation is used to select a subset of the attributes of a relation by specifying the names of the required attributes.

For example, to get a list of all employees surnames and employee numbers:

$\pi_{\text{surname, empno}}(\text{employee})$

d) List out parts of SQL and state their functions.
[Listing 2 Marks, Functions 2 Marks]
[Remark: Any relevant answer should be given marks]

Ans1: List of SQL parts: DDL, DML, DCL, and TCL

1) DDL: Data Definition Language is a standard subset of SQL that is used to define tables. There are many other statements, but those are the ones most commonly used.
CREATE TABLE to create a table in the database
DROP TABLE to remove a table from the database
ALTER TABLE to add or remove columns from a table in the database

2) DML: This is a standard subset of SQL that is used for data manipulation
SELECT to select rows of data from a table
INSERT to insert rows of data into a table
UPDATE to change rows of data in a table
DELETE to remove rows of data from a table

3) DCL: The Data Control Language (DCL) This component of the SQL language is used to create privileges to allow users access to, and manipulation of, the database. There are two main commands:
GRANT to grant a privilege to a user
REVOKE to revoke (remove) a privilege from a user

4) TCL: Transaction Control (TCL) statements are used to manage the changes made by DML statements.
COMMIT - To make the changes permanent in the database
SAVEPOINT - saving point in a transaction to which you can later roll back
ROLLBACK - Restore database to original since the last COMMIT

OR

Ans2: The SQL is subdivided into following parts:

- **Clauses**: which are constituent components of statements and queries. E.g. select clause, where clause, order by clause, group by clause etc.
- **Expressions**: which can produce either scalar values or tables consisting of columns and rows of data.
  e.g. select name, salary*12 from emp; (In this salary*12 is the expression)
- **Predicates**: It specifies conditions. E.g. select * from emp where salary>5000 and dept= 'Marketing';
- **Insignificant whitespace**: generally ignored in SQL statements and queries, making it easier to format SQL code for readability.

e) State any four built in aggregate functions of SQL and explain it in brief.
[Listing of 4 Aggregate functions 2 Marks i.e. ½ Mark each function and any relevant explanation of functions 2 Marks ½ Mark each function's explanation]

Ans: 1) Following are four aggregate functions:
min
max
avg
sum
count

2)

min	**returns the smallest value in a given column** Syntax : SELECT min(column_name) FROM table_name; Example: SELECT min(ssc_per) from stud_info;
max	**returns the largest value in a given column** Syntax : SELECT max(column_name) FROM table_name; Example: SELECT max(ssc_per) from stud_info;

sum	**returns the sum of the numeric values in a given column** Syntax : SELECT sum(column_name) FROM table_name; Example: SELECT sum(fees) from branch_details;
avg	**returns the average value of a given column** Syntax : SELECT avg(column_name) FROM table_name; Example: SELECT avg(ssc_per) from stud_info;
count	**returns the total number of values in a given column** Syntax : SELECT count(column_name) FROM table_name; Example: SELECT count(stud_id) from stud_info;
count(*)	**returns the number of rows in a table** Syntax : SELECT count(*) FROM table_name; Example: SELECT count(*) from stud_info;

f) Explain in brief concept of 'Trigger'.
[Trigger Definition 1 Mark, Syntax 1 Mark, any correct Explanation with example 2 Marks]

Ans: 1) Trigger Definition:

A trigger is a pl/sql block structure which is fired when a DML statements like Insert, Delete, Update is executed on a database table. A trigger is triggered automatically when an associated DML statement is executed.

2) Syntax:

The Syntax for creating a trigger is:
CREATE [OR REPLACE] TRIGGER trigger_name
{BEFORE | AFTER | INSTEAD OF }
{INSERT [OR] | UPDATE [OR] | DELETE}
ON table_name
[FOR EACH ROW/For Each statement]

```
WHEN (condition)
BEGIN
 --- sql statements
END;
```

3) For Example:

Let's create a table 'product_check' which we can use to store messages when triggers are fired.

```
CREATE TABLE product_check
(Message varchar2(50),
Current_Date  date);
```

1) BEFORE UPDATE, Statement Level: This trigger will insert a record into the table 'product_check' before a sql update statement is executed, at the statement level.

```
CREATE or REPLACE TRIGGER Before_Update_Stat_product
BEFORE
UPDATE ON product
Begin
INSERT INTO product_check
Values('Before update, statement level',sysdate);
END;
/
```

4. Attempt any **FOUR** of the Following: 16 Marks

a) Explain Join concept of SQL. State types of join
[Join concept 1 Mark, Types 3 Marks]

1) JOIN: SQL Joins are used to relate information in different tables. A Join condition is a part of the sql query that retrieves rows from two or more tables. A SQL Join condition is used in the SQL WHERE Clause of select.

OR

Join is used to combine the data spread across tables. A join is performed by the where clause which combines the specified rows of the tables.
SQL Join types are as follows:

1) INNER JOIN or EQUI JOIN:

A join which is based on equalities is called equi join. In equi join comparison operator "=" is used to perform a Join.

Syntax:

```
SELECT tablename.column1_name,tablename.column1_name
FROM table_name1,table_name2
where table_name1.column_name=table_name2.column_name;
```

Example:

```
Select stud_info.stud_name,stud_info.branch_code,branch_details.location
```

```
From stud_info,branch_details
Where
Stud_info.branch_code=branch_details.branch_code;
```

2) SELF JOIN:

The SQL **SELF JOIN** is used to join a table to itself, as if the table were two tables, temporarily renaming at least one table in the SQL statement.

Syntax:

```
SELECT a.column_name, b.column_name...
FROM table1 a, table1 b
WHERE a.common_filed = b.common_field;
```

Example:

```
Select x.stud_name, y.stud_name from stud_infox,stud_info y
Where x.leader= y.stud_id;
```

3) LEFT OUTER JOIN:

A left outer join retains all of the rows of the “left” table, regardless of whether there is a row that matches on the “right” table.

Syntax:

```
Select column1name,column2name
from
table1name any_alias1 ,table2name any_alias2
on
any_alias1.columnname(+) = any_alias2.columnname;
```

OR

```
Select column1name,column2name
from
table1name left outer join table2name on
table1name.columnname= table2name.columnname;
```

Example:

```
select
last_name,department_name
from
employees e,departments d
on
e.department_id(+) = d.department_id;
```

OR

```
select
last_name,department_name
from
employees  left outer join departments
on
employees.department_id = departments.department_id;
```

4) RIGHT OUTER JOIN

A right outer join retains all of the rows of the "right" table, regardless of whether there is a row that matches on the "left" table.

Syntax:

```
Select column1name,column2name
from table1name any_alias1 ,table2name any_alias2
on
any_alias1.columnname =any_alias2.columnname (+);
```

OR

```
Select column1name,column2name
from  table1name any_alias1  right outer join table2name any_alias2
on
any_alias1.columnname =any_alias2.columnname;
```

Example:

```
Select last_name,department_name
from
employees e,departments d
on
e.department_id = d.department_id(+);
```

OR

```
Select last_name,department_name
from
employees e right outer join departments d
on
e.department_id = d.department_id;
```

5) NON EQUI JOIN:

Non equi joins is used to return result from two or more tables where exact join is not possible.

Syntax:

Select aliasname.column1name, aliasname.column2name from tablename alias where <condition using range>;

For example:

we have emp table and salgrade table. The salgrade table contains grade and their low salary and high salary. Suppose you want to find the grade of employees based on their salaries then you can use NON EQUI join.
Select e.empno, e.ename, e.sal, s.grade from emp e, salgrade s where e.sal between s.lowsal and s.hisal;

b) With example explain 'select' clause and 'where' clause.
[Select: example 1 Mark and explanation 1 Mark, where: example 1 Mark and explanation 1 Mark]

Ans: 1) The SELECT statement is used to select data from a database. The result is stored in a result table.

SQL SELECT Syntax:

i)Select columnname(s) from tablename;
This syntax will display only specified columns in the list.
ii) select * from tablename;
This syntax will display all the columns in the table.

SQL SELECT Example:

i)Select stud_id,stud_name from stud_info;
ii) select * from stud_info;

2)The WHERE Clause

The WHERE clause is used to extract only those records that fulfill a specified condition.

SQL WHERE Syntax:

```
SELECT column_name(s)
FROM table_name
WHERE column_name=rowvalue;
```

Example:

```
SELECT * FROM stud_info
WHERE stud_name='Ameya';
```

c) Explain block structure of PL-SQL
[Syntax of block structure of PL-SQL: 1 Mark, Explanation of Declare 1 Mark, Begin-End 1Mark, Exception 1 Mark]

Ans:

```
DECLARE
Variable declaration
BEGIN
Program Execution
EXCEPTION
Exception handling
END;
```

Each PL/SQL program consists of SQL and PL/SQL statements which from a PL/SQL block.
A PL/SQL Block consists of four sections:
The Declaration section :
Declaration of memory variables used later in begin section.
The Begin..End section :
SQL executable statements for manipulating table data should be in BEGIN....END block.
The Exception section :
SQL and/or PL-SQL code to handle errors that may crop up during execution of the above code block.

d) List out any four statements of PL-SQL
[Listing of any 4 statements 1 Mark each for each]

Ans: 1) IF-THEN
2) IF-THEN-ELSE
3) IF-THEN-ELSIF
4) LOOP
5) EXIT-WHEN
6) NULL
7) dbms_output.put_line
8) Any correct PL-SQL/SQL statement which can be used in BEGIN….END block.

e) Describe in brief control structure of PL-SQL.
[2 Marks for each control structure with any relevant example, Any 2 Control structure of the following]

1) IF-THEN Statement
The simplest form of IF statement associates a condition with a sequence of statements enclosed by the keywords THEN and END IF (not ENDIF), as follows:
IF condition THEN
sequence_of_statements
END IF;

The sequence of statements is executed only if the condition is true. If the condition is false or null, the IF statement does nothing. In either case, control passes to the next statement. An example follows:

```
IF sales > quota THEN
compute_bonus(empid);
 UPDATE payroll SET pay = pay + bonus WHERE empno = emp_id;
END IF;
```

2) IF-THEN-ELSE Statement

The second form of IF statement adds the keyword ELSE followed by an alternative sequence of statements, as follows:

```
IF condition THEN
  sequence_of_statements1
ELSE
  sequence_of_statements2
END IF;
```

The sequence of statements in the ELSE clause is executed only if the condition is false or null. Thus, the ELSE clause ensures that a sequence of statements is executed. In the following example, the first UPDATE statement is executed when the condition is true, but the second UPDATE statement is executed when the condition is false or null:

```
IF trans_type = 'CR' THEN
  UPDATE accounts SET balance = balance + credit
ELSE
  UPDATE accounts SET balance = balance - debit
END IF;
```

3) IF-THEN-ELSIF Statement

Sometimes you want to select an action from several mutually exclusive alternatives. The third form of IF statement uses the keyword ELSIF (not ELSEIF) to introduce additional conditions, as follows:

```
IF condition1 THEN
  sequence_of_statements1
ELSIF condition2 THEN
  sequence_of_statements2
ELSE
  sequence_of_statements3
END IF;
```

If the first condition is false or null, the ELSIF clause tests another condition. An IF statement can have any number of ELSIF clauses; the final ELSE clause is optional. Conditions are evaluated one by one from top to bottom. If any condition is true, its associated sequence of statements is executed and control passes to the next statement. If all conditions are false or null, the sequence in the ELSE clause is executed. Consider the following example:

```
BEGIN
  ...
  IF sales > 50000 THEN
bonus := 1500;
  ELSIF sales > 35000 THEN
bonus := 500;
  ELSE
bonus := 100;
  END IF;
  INSERT INTO payroll VALUES (emp_id, bonus, ...);
END;
```

There are three types of loops in PL/SQL:

- Simple Loop
- While Loop
- For Loop

1. A Simple Loop is used when a set of statements is to be executed at least once before the loop terminates. An EXIT condition must be specified in the loop, otherwise the loop will get into an infinite number of iterations. When the EXIT condition is satisfied the process exits from the loop.

   The General Syntax to write a Simple Loop is:

   ```
   LOOP
   Statements;
     EXIT;
     {or EXIT WHEN condition;}
   END LOOP;
   ```

2. While Loop

   A WHILE LOOP is used when a set of statements has to be executed as long as a condition is true. The condition is evaluated at the beginning of each iteration. The iteration continues until the condition becomes false.

   The General Syntax to write a WHILE LOOP is:

   ```
   WHILE <condition>
   ```

```
 LOOP statements;
END LOOP;
```

3. FOR Loop

A FOR LOOP is used to execute a set of statements for a predetermined number of times. Iteration occurs between the start and end integer values given. The counter is always incremented by 1. The loop exits when the counter reachs the value of the end integer.

The General Syntax to write a FOR LOOP is:

```
FOR counter IN val1..val2
  LOOP statements;
END LOOP;
```

val1 - Start integer value.

val2 - End integer value.

f) Explain Generalisation of E-R model.

[Explanation of Generalisation 2 Marks, Diagram 1 Mark]

Ans: 1) A generalization hierarchy is a form of abstraction that specifies that two or more entities that share common attributes can be generalized into a higher-level entity type called a super type or generic entity.

2) The lower level of entities becomes the subtypes, or categories, to the super type. Subtypes are dependent entities.

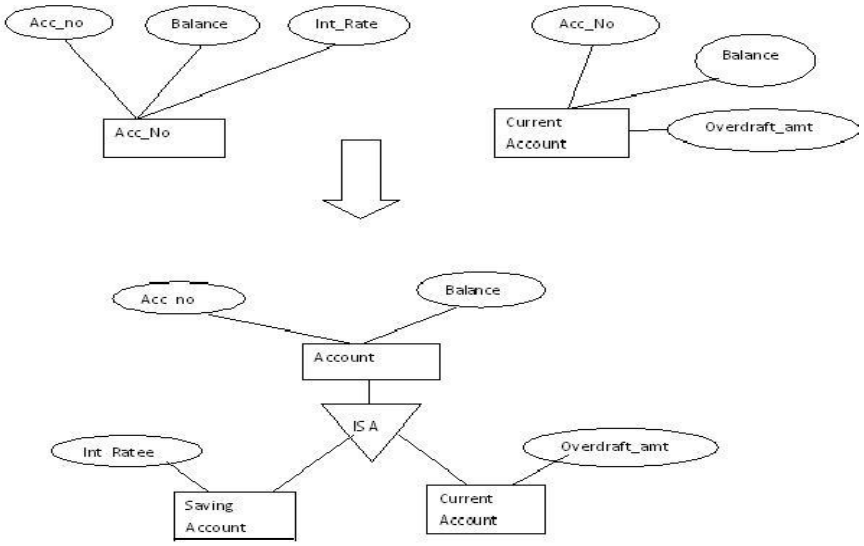
3) Generalization is used to emphasize the similarities among lower-level entity set and to hide differences.

4) Generalization is bottom-up design process.

5) It makes ER diagram simpler because shared attributes are not repeated. Generalization is denoted through a triangular component labeled 'IS A" as shown in fig.

6) In fig, Account is the higher-level entity set and saving account and current account are lower-level entity sets.

The attributes account_no and balance are common for saving and current account which can be mentioned only in account entity.



5. Attempt any FOUR of the following. 16 Marks

a) What is decomposition? Which are desirable properties of decomposition.
[Explanation decomposition – 3 Marks, properties – 1 Mark]

Ans: Decomposition replaces an "unnormalized" relation by a set of normalized relations
For relation R(x,y,z) there can be 2 subsets: R1(x,z) and R2(y,z)
If we union R1 and R2, we get R
R = R1 U R2
If **R** is a relation scheme then **{R_1, R_2, ..., R_n}** is a decomposition if
$R = R_1 \cup R_2 \cup ... \cup R_n$

Properties of decomposition

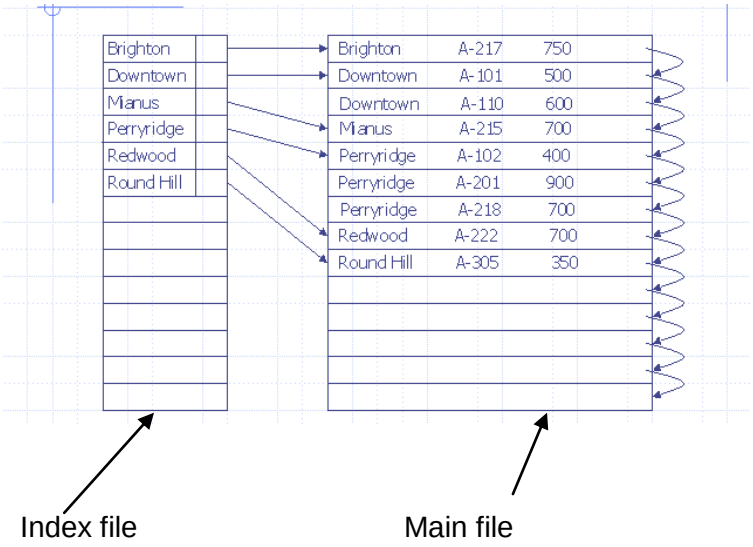
1. Lossless join decomposition
2. Dependency preservation
3. Attribute must be in some relation
4. Minimum repetition of information

A decomposition of a relation **R** is a set of relations **{R_1,R_2,...,R_n}** such that each **R_i** is a subset of **R** and the union of all of the **R_i** is **R**.

b) How to access records using sequential index access method.
[Explanation 4 Marks]

Ans: 1) Sequential file that is indexed is called as index sequential file.
2) The records are stored sequentially by primary key values and there is an index built over the primary key field.

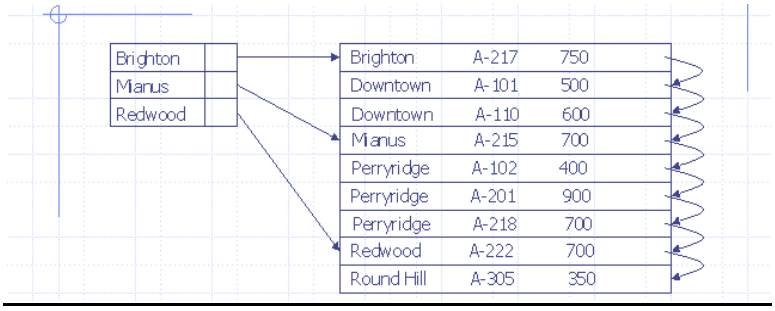
3) An index is a set of <key,pointer>pairs where the pointer is the position in the data file of the record with the given key.
4) The key search proceeds as follows
   The search key is compared with the index(value) to find the highest index key preceding the search one and a linear search is performed from the record.
5) The index key points onwards until the search key is match or until the record pointed by the next index entry is reached.
6) In spite of the double file access(index+data) needed by this kind of search the decrease in access time with respect to a sequential file is significant.
7) The index allows for random access to records, while the sequential storage of the records of the file provides easy access to the sequential records.



c) Explain the concept of dense and sparse indexing.
[Explanation of Dense 2 Marks, sparse – 2 Marks, Any two valid points 1 Mark each]

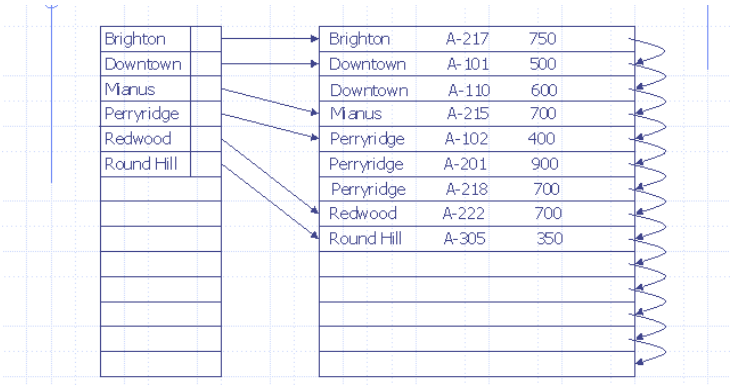
Ans: Sparse or Non-Dense Primary Index:

1) Sparse indices are those indices that do not include all the available values of a field
2) To keep the size of the index smaller, instead of pointing to each and every record in the main table, the index points to the particular records in the main table.
3) An index groups the records as per the index values.



Dense index:

1) A dense index contains one index entry for every value of the indexing attributes,
2) For large tables the Dense Index itself begins to grow in size. the number of entries in the index table is the same as the number of entries in the main table.
3) The index record contains the search key value and a pointer to the first data record with that search key value.



d) State properties of Boycee Codd Normal form.
[Any Relevant/valid point regarding BCNF 1 Mark each]

Ans: 1) BCNF A relation R is in Boyce-Codd normal form (BCNF) if and only if every determinant is a candidate key.
2) In BCNF non-trivial functional dependency is preserved for super key.
3) BCNF gives more consistency.
4) The following relation is in BCNF.
SUPPLIERS (supplier_no, supplier_name, city, zip)
We assume that each supplier has a unique supplier_name, so that supplier_no and supplier_name are both candidate keys.

e) What is Hashing? State meaning of static and dynamic hashing.
[Explanation of hashing 3 Marks, Static and dynamic hashing 1Mark]

Ans: Hashing: 1) in hashing a bucket is unit of storage containing one or more records.It is typically a disc block.

2) In hashing the hash function is used to locate record for access , insertion as well as deletion.
3) Hashing is also used as an internal search structure within a program whenever a group of records is accessed exclusively by using the value of one field.
4) Hashing affords an extremely fast method for direct retrieval, but offers no support for range searches.
5) There are two basic forms of hash methods: static and dynamic.

Static Hashing:

The set of all possible search key value is large,the set{k1,k2,…,kn} of search key clause actually stored in database is much smaller than k. Number of buckets are fixed.

Dynamic Hashing

It is used to modify the hash function dynamically in order to accommodate the growth or shrinkage. Number of buckets are not fixed, but grows or shrinks as needed.

f) Explain string, date and time functions of SQL.
[Any 4 string functions syntax/ example 2 Marks, Any 2 functions Date and time syntax /example 2 Marks)

Ans: Character functions

1) Initcap (char): Returns the input string with initial letter capitalized and all other character in lowercase.
Select initcap('hello') from dual;
2) Lower(char): Returns the input string with all letters in lowercase
Select lower(stud_name) from stud_info;
3) Upper(char): Returns the input string with all letters in uppercase
Select upper(stud_name) from stud_info;
4) Ltrim(char,set): It trims from the left of character string
Select ltrim('Sachin Tendulkar','Sachin') from dual;
5) Rtrim(char,set): It trims from the right of character string
Select rtrim('Sachin Tendulkar','Tendulkar') from dual;
6) Translate(char,from string,to string): It returns expr with all occurrences of each character in from_string replaced by its corresponding character in to_string.
Select translate(Hickory,'H','D') from dual;
7) Replace(char,searchstring,[restring]): It returns character string with each occurrences of searchstring replaced with [restring]

Select replace('Tick and Tock','T','Cl') from dual;

8) Substr(char,m,n) : It returns substring of character string that stack at m character and is of length n

Select substr(Triangle'4,5) from dual;

9) Lpad(char1,length,char2) : It returns char1, left-padded to give length with the sequences of characters in char2

Select lpad('SKY',8,'=') from dual;

10) Rpad(char1,length,char2) : It returns char1, right-padded to give length with the sequences of characters in char2

Select rpad('SKY',8,'=') from dual;

11) Length(char) : It returns the length of character string

Select length('Hello') from dual;

12) Concatenation(||): It merges two or more strings or a string and a data value together.

Date function:

1) Months_between(d1,d2) : Used to find number of months between d1 and d2.
Select months_between('05-MAY-1996','05-JAN-1996') "Months" from dual;

2) Add_months(d,n) : returns date after adding the number of months specified with the function. Select add_months(sysdate,2) from dual;

3) Next_day(d,char) : Returns the date of the first weekday named 'char' that is after the date named by date.
Select next_day('01-FEB-2006','Wednesday') "next_day" from dual;

4) Last_day(d) : Returns the last day of the month that contains date 'd'.
Select last_day(sysdate) "last" from dual;

5) Round(date,[fmt]) : retuns date rounded to the unit specified by the format model 'fmt'.
Select round(sysdate,'day') "round_day" from dual;

6) Trunc(date([fmt]) : Returns date with the time portion of the day truncated to the unit specified by the format model fmt.
Select trunc(sysdate,'day') "trunc_day" from dual;

7) New_time(date,'d','n') : Returns the time and date of a date column or literal in other given time zones.
Select new_time(sysdate,'est','pst') from dual;
Where est is eastern standard time & pst is pacific standard time

6. Attempt any **TWO** of the following. 16 Marks

a) State strategies of general query processing.
[Any 4 basic strategies 2 Marks each]

Ans: The main aim of the query processing is to minimize cost of each query execution. The cost may be in the form of time or space complexity.
Thus query processing to reduce the size of the intermediate & final results as well as processing cost. The general strategies are as follows:

1) Perform selection as early as possible :
Selection reduces the cardinality of the relation & reduces the subsequent processing time.

2) Combine number of unary operations : Both the selection & projection can be done on the tuple or relation R simultaneously ,requiring a single pass over the tuples,thus providing singular access to them.
For example π_x (σ_y(R)) i.e Both selection and projection operations can be done simultaneously. Similarly σ_{c1} (σ_{c2}(R))= $\sigma_{c1 \wedge c2}$(R)
Where c1 and c2 are certain conditions.

3) Conversion of Cartesian into join:
Convert the Cartesian with a certain subsequent selection into a natural join.
e.g. σ_y (R x S) = R$\bowtie$ S

4) Compute common expression only once:
A common expression that appears more than once in query may be computed once, stored & then reused.

5) Preprocess the relation:
Before performing an operation such as a join we can preprocess the relation. The preprocessing includes sorting & index creation on the join attributes.

b) How deadlock can be detected? And write procedure to recover from it.
[Deadlock detection 4 Marks, recovery 2 Marks for each recovery technique]

Ans: Deadlock is a situation in which two or more transactions are in a wait state, each of them waiting for one of the others to release a lock before it can proceed.

Deadlock detection
If there is a cycle in wait for graph then deadlock is detected.

Procedure to draw wait for graph:
1) Wait for graph consist of a pair G=(V,E) where V are Vertices and E are edges.
2) If Ti request a data item currently being held by Tj , then the edge $T_i \rightarrow T_j$ is inserted in the wait for graph.
3) This edge is removed only when T_j is no longer holding a data item needed by Ti.

4) The system is in deadlock if and only if the wait for graph has a cycle e.g in the above Fig.2 all the transactions are waiting for each other to release the data item thus it forms the cycle in the wait for graph so there is a deadlock.

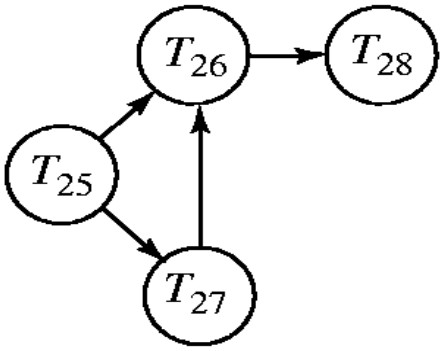


Fig.1 Wait-for graph without a cycle

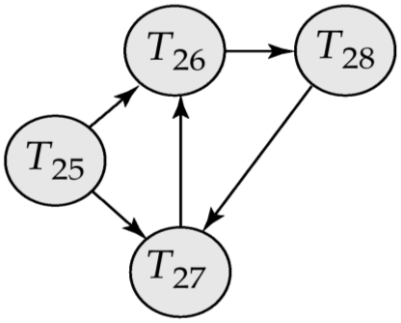


Fig.2Wait-for graph with a cycle

Recovery from Deadlock

Once it is determined that a deadlock exits, the system needs to recover from the deadlock. For this, one or more transactions are rolled back to break the deadlock

a. Selection of a victim :

In the situation of a deadlock, you first need to determine the transaction (or transactions) that should be rolled back to break the deadlock. Such a transaction is called the victim transaction. The transaction that will lead to minimum loss, in terms of cost, should be chosen for rollback.

The following factors determine the cost of a rollback :

1)How much execution has the transaction completed and for how much more time the transaction will execute to complete its task?

2) How many data items were used by the transaction?

3) How many more data items does the transaction need to complete?

4) How many transaction will be included in the rollback?

b. Rollback:

After determining the transaction to be rolled back, you need to determine how far the transaction is to be rolled back. The easiest answer to this problem is to do a total rollback, which means that the transaction will be aborted and restarted. However, it is better to roll back the transaction only till the point where the deadlock can be broken. This method requires the DBMS to maintain information about all current transaction.

c. Starvation:

When the selection of a victim is based on cost factors, it might happen that the same transaction is selected as a victim every time a deadlock occurs. Due to this, the transaction might not be able to complete its task. Such a situation is called starvation. To avoid starvation, you need to ensure that a transaction is picked as a victim for only a fixed number of times. To ensure this, you can select a victim based on the number of rollbacks along with the cost factor.

c) List out lock based protocol and explain in brief any two protocol.
[Listening 2 Marks, explanation 3 Marks each]

Ans: 1.Two phase protocol
2. Timestamp based protocol
3. Validation based protocol
4. Multiple Granularities protocol

A lock is a mechanism to control concurrent access to a data item
Data items can be locked in two modes :
1. exclusive (X) mode. Data item can be both read as well as written. X-lock is requested using lock-X instruction.
2. shared (S) mode. Data item can only be read. S-lock is requested using lock-S instruction.
Lock requests are made to concurrency-control manager. Transaction can proceed only after request is granted.
Lock-compatibility matrix:

	Shared lock	**Exclusive lock**
Shared lock	**True**	**False**
Exclusive lock	**False**	**False**

Types of lock based protocol

1) Two-Phase Locking

1) Two-phase locking defines how transactions acquire and relinquish (or revoke) locks.
1. Growing phase – acquires all the required locks without unlocking any data. Once all locks have been acquired, the transaction is in its locked point.
2. Shrinking phase – releases all locks and cannot obtain any new lock.

2) In the example for two-phase locking protocol , the transaction acquires all the locks it needs (two locks are required) until it reaches its locked point.

When the locked point is reached, the data are modified to conform to the transaction requirements. The transaction is completed as it released all of the locks it acquired in the first phase.

3) Two-phase locking does not ensure freedom from deadlocks.

Cascading roll-back is possible under two-phase locking.

Example of two phase lock

T_5	T_6	T_7
lock-X(A)		
read(A)		
lock-S(B)		
read(B)		
write(A)		
unlock(A)		
	lock-X(A)	
	read(A)	
	write(A)	
	unlock(A)	
		lock-S(A)
		read(A)

There are many protocols derived from 2PL :

Strict two-phase locking. Here a transaction must hold all its exclusive locks till it commits.

Rigorous two-phase locking is even stricter: here all locks (shared and exclusive) are held till commit. In this protocol transactions can be serialized in the order in which they commit.

2)Timestamp-Based Protocols

- Each transaction is issued a timestamp when it enters the system. If an old transaction T_i has time-stamp TS(T_i), a new transaction T_j is assigned time-stamp TS(T_j) such that TS(T_i) < TS(T_j).
- The protocol manages concurrent execution such that the time-stamps determine the serializability order.
- In order to assure such behavior, the protocol maintains for each data *Q* two timestamp values:
  - **W-timestamp**(*Q*) is the largest time-stamp of any transaction that executed write(*Q*) successfully.
  - **R-timestamp**(*Q*) is the largest time-stamp of any transaction that executed read(*Q*) successfully.
- The timestamp ordering protocol ensures that any conflicting read and write operations are executed in timestamp order.
- Suppose a transaction T_i issues a **read**(*Q*)

1. If TS(T_i) ≤**W-timestamp**(Q), then T_i needs to read a value of Q that was already overwritten. Hence, the read operation is rejected, and T_i is rolled back.
2. If TS(T_i) ≥**W-timestamp**(Q), then the read operation is executed, and **R-timestamp**(Q) is set to the maximum of **R-timestamp**(Q) and TS(T_i).

➢ Suppose that transaction T_i issues **write**(Q).
1. If TS(T_i) < **R-timestamp**(Q), then the value of Q that T_i is producing was needed previously, and the system assumed that value would never be produced. Hence, the write operation is rejected, and T_i is rolled back.
2. If TS(T_i) < **W-timestamp**(Q), then T_i is attempting to write an obsolete value of Q. Hence, this write operation is rejected, and Ti is rolled back.
3. Otherwise, the **write** operation is executed, and **W-timestamp**(Q) is set to TS(T_i).

3) Validation-based protocol

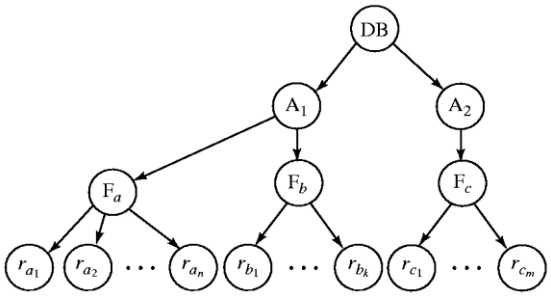
1) A **validation-based protocol** that assumes most database operations do not conflict.
2) No requirement on locking or time stamping techniques.
3) Transaction executed without restrictions until committed and fully in the hope that all will go well during validation.
4) Two or three Phases:

■ **Read (and Execution) Phase** – the transaction reads the database, executes the needed computations, and makes the updates to a private copy of the database values.
■ **Validation Phase** – the transaction is validated to ensure that the changes made will not affect the integrity and consistency of the database.
■ **Write Phase** – the changes are permanently applied to the database.
5)The optimistic approach is acceptable for mostly read or query database system that require very few update transactions.
6)Each transaction T_i has 3 timestamps
■ **Start**(T_i) : the time when T_i started its execution
■ **Validation**(T_i): the time when T_i entered its validation phase
■ **Finish**(T_i): the time when T_i finished its write phase
7) Serializability order is determined by timestamp given at validation time, to increase concurrency. Thus TS(T_i) is given the value of **Validation**(T_i).
8) This protocol is useful and gives greater degree of concurrency if probability of conflicts is low.
9) That is because the serializability order is not pre-decided and relatively less transactions will have to be rolled back.

4) Multiple Granularities

1. Allow data items to be of various sizes and define a hierarchy of data granularities, where the small granularities are nested within larger ones.
2. Can be represented graphically as a tree. When a transaction locks a node in the tree *explicitly*, it *implicitly* locks all the node's descendents in the same mode.
3. Granularity of locking (level in tree where locking is done):
- *fine granularity* (lower in tree): high concurrency, high locking overhead

- *coarse granularity* (higher in tree): low locking overhead, low concurrency



The highest level in the example hierarchy is the entire database.
The levels below are of type *area*, *file* and *record* in that order.

Following are the modes of lock used in multiple granularity schemes:

1. Shared mode lock (S).
2. Exclusive mode lock (X).
3. Intension – Shared mode lock (IS).
4. Intension – Exclusive mode lock (IX).
5. Shared and intension – exclusive mode lock (SIX).

Intension Shared mode lock (IS): If a node is locked in intension shared mode then explicit locking is done at lower level of tree but with only shared mode lock.

Intension Exclusive mode lock (IX): If a node is locked in intension exclusive mode then explicit locking is done at a lower level with exclusive mode or shared mode lock.

Shared and intension – exclusive mode lock (SIX): If a node is locked in intension exclusive mode, the sub tree routed is locked explicitly in shared mode and that explicit locking is been done at lower level with exclusive mode lock.