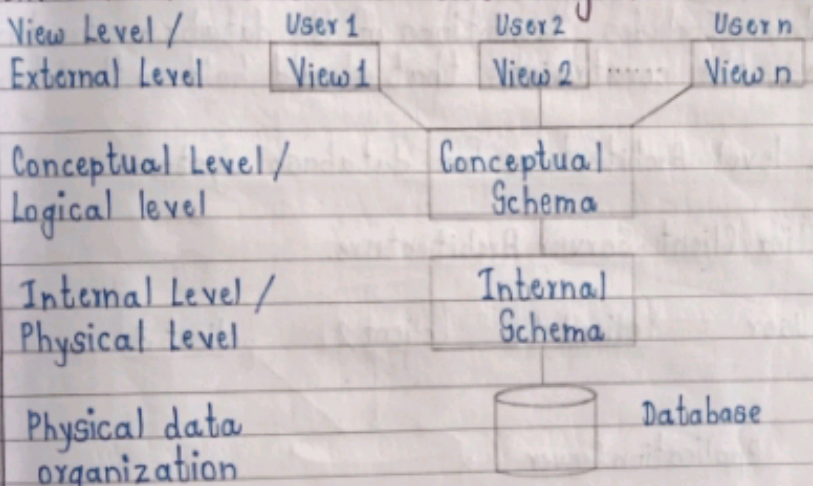
ASSIGNMENT - 1

1) Explain three levels of data abstraction with suitable diagram. What is data abstraction, instances and Schema. Explain three level architecture with diagram.

→



Data Abstraction: Only providing required information by hiding actual data.

View or External level: An application programs hide details of data types. It deals with the way the user see the data.

Logical or Conceptual Level: It describes data stored in database, and the relationships among the data. Describes what data is stored in database and relationships among the data. It includes authorization & validation procedures.

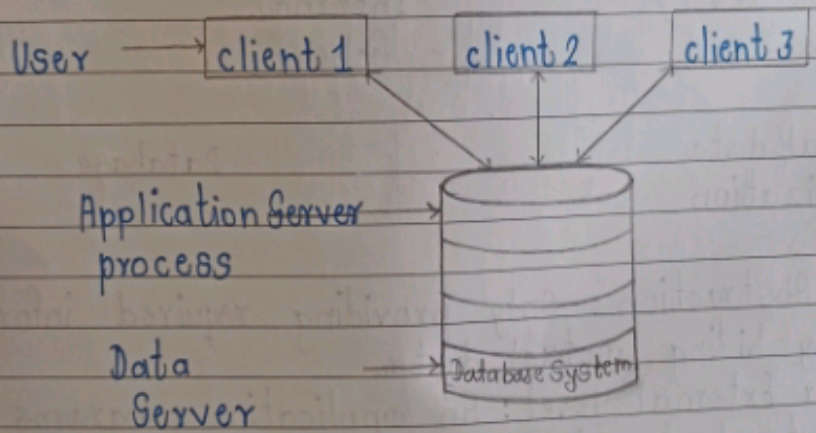
Physical or Internal Level : It describes how a record (e.g. customer) is stored.

Instance : The actual data stored in a database at a particular moment in time. Also called database state (or occurrence).

Schema : Overall design of database is called as database schema. Includes descriptions of the database structure and the constraints that should hold on the database.

Three level Architecture for database System

Two-Tier Client-Server Architecture.



User System - Provides user friendly layer for communication.

Processing management or Application : Includes process monitoring, development, implementation process.

Database management : Includes database & file services.

Three-Tier Client-S

client1 clie

Prog

Applicati

Data

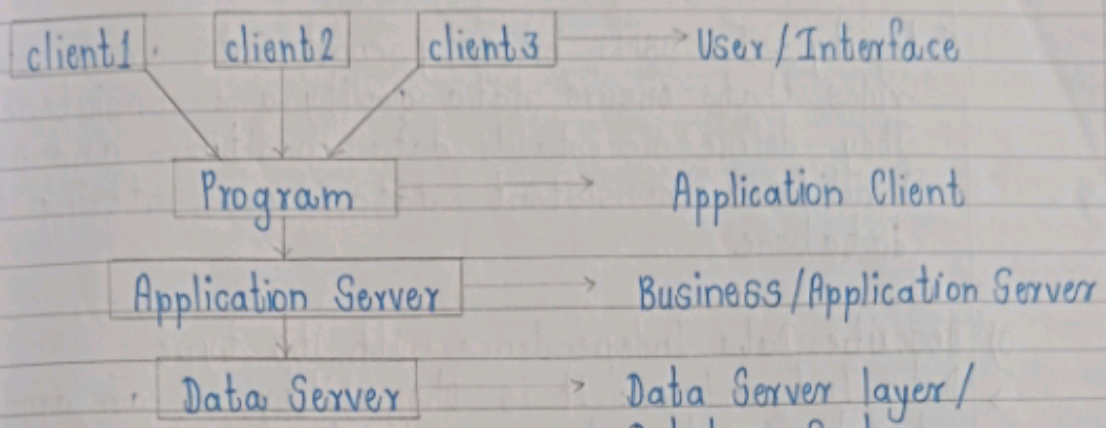
Additional feature
Encrypt the dat
Decrypt data
Thin client, requir
Application ma
Easier to modif

User : It provid
Application Client

Application Serve
Database Syste

2) Explain Data Red
→ Data Redundancy
is stored in r

Three-Tier Client-Server Architecture

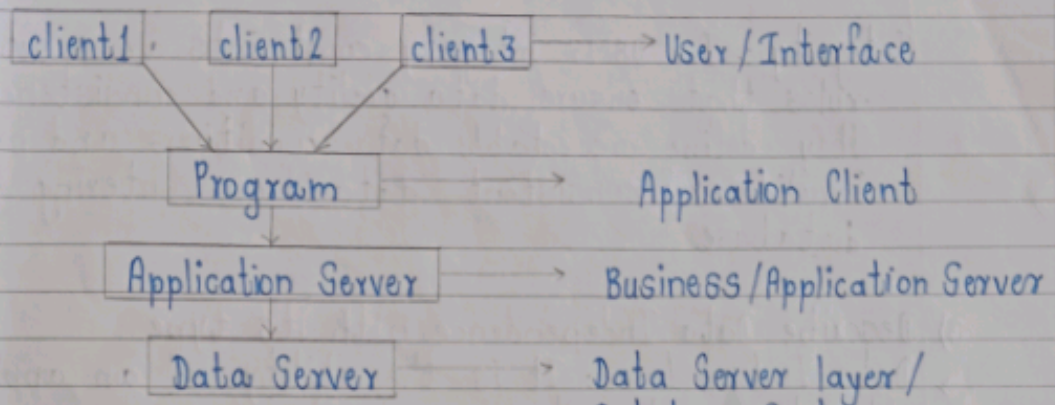


Additional features - Security:
Encrypt the data at the server before transmission
Decrypt data at the client
Thin client, requiring less expensive hardware.
Application maintenance centralized.
Easier to modify or replace one tier without affecting others.

User :- It provides layer for communication.
Application Client :- i) Business logic
ii) Data Managing
Application Server :- i) Database Storage
Database System :- i) Process management.
ii) Database file services.

2) Explain Data Redundancy and Integrity.
→ Data Redundancy :- Data redundancy is when the same data is stored in multiple places, such as within a database

Three-Tier Client-Server Architecture



Additional Features - Security:
Encrypt the data at the server before transmission
Decrypt data at the client
Thin client, requiring less expensive hardware.
Application maintenance centralized.
Easier to modify or replace one tier without affecting others.

User :- It provides layer for communication.
Application Client :- i) Business logic
ii) Data Managing
Application Server :- i) Database Storage
Database System :- i) Process management.
ii) Database file services.

2) Explain Data Redundancy and Integrity.

→ Data Redundancy :- Data redundancy is when the same data is stored in multiple places, such as within a database

or across different data systems.

Integrity :- In DBMS integrity constraints are a set of rules that ensure data quality and consistency. They define acceptable data conditions and prevent invalid or inconsistent data from entering the database.

3) Describe Data Independence with its type.

→ Data Independence: It is the ability of an application to change the storage structure & access strategy. This is a prime advantage of a database. In conventional systems applications are data-dependent.

It is divided into 2 types:

Logical Data Independence: The ability to change the conceptual schema without having to change the external schemas and their application programs. It is easy to achieve this.

Ex. New fields can be added to database without disturbing old records.

Physical Data Independence: The ability to change the internal schema without having to change the conceptual schema. It is difficult to achieve this.

Ex. To achieve this, attributes of different tables are considered & changes are done. Then those changes are reflected to old one.

4) Explain any four functions of DBMS.

→ 1. Data Structure and Management: The primary function of an DBMS is used to store data in a specified and formatted pattern.
Management includes insertion, create and deleting data.

2. Data Security and Integrity: DBMS provides a feature that only specified users can access the database integrity. Some rules are applied for by maintaining consistency rules.

3. Transaction Management: A DBMS supports transaction management to ensure that operations.

4. Concurrency Control: A DBMS allows multiple users to access the Database simultaneously without interfering with each other.

5) List and explain types of DBMS users. List four function of database administrator.

→ Database Administrators (DBAs): DBAs are responsible for managing and maintaining the database system. They ensure its optimal performance, availability, and security.

2. Database Designers: Database designers are responsible for designing the structure of the database. This includes defining the schema, tables, relationships, and constraints.

3. Application Programmers: Application programmers write software applications that interact with the database. They develop the programs that query the database and process the data.

4. End Users: End users are the individuals who interact with the database through applications or directly via query tools.

Four functions of Database Administrator
1. Database Installation and Configuration.
2. Backup and Recovery Management.
3. Performance Tuning.
4. Security Management.
5. Database planning
6. Logical Design.

6) State and explain four advantages of DBMS overfile processing system.

→ 1. Very less data redundancy and Inconsistency:
The duplication occurs is very less than thats why accuracy increases.

2. Provide better security:
It provide better security than file processing system.

3. Easy access data:
We can access data very easily over than file processing system.

4. It supports atomicity:
If failure occurs what work will automatically saved.

5. Provide security better:
It provides security better than file processing system.

7) State four differences between DBMS and RDBMS.

→

DBMS	RDBMS
1. It utilities less relationship.	1. It utilities more relationship.
2. Operation speed is low	2. Operation speed is high.
3. Less facilities & utilities are offered.	3. More facilities & utilities are offered.
4. DOS based platform	4. DOS, UNIX platform.
5. It uses concept of file.	5. It uses concept of table.
6. Ex Dbase, foxbase, MySQL, Foxpro, Microsoft SQL, etc.	6. Ex. Oracle, SQL server, etc.

8) Explain the overall structure of DBMS with suitable diagram.

→ Database User:

1. Naive : He don't have technical knowledge of DBMS. Just do data entry task.
2. Application programmer : He is a computer professional.
3. Sophisticated user : He only uses Query tools user.
4. DBA : He is supreme user who uses admin level tools.

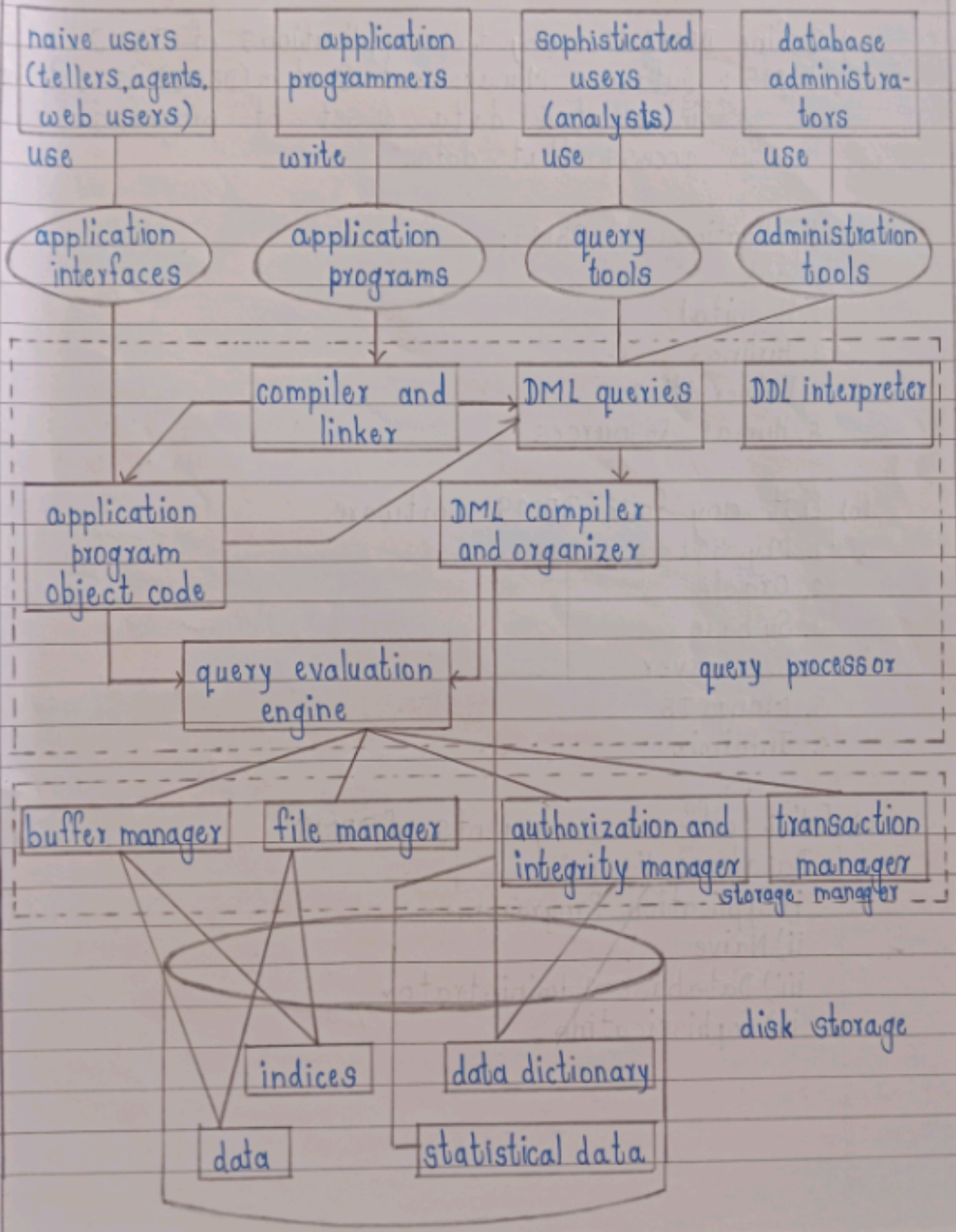
Query Processor:

1. DDL interpritors : It interpretes or resolve DDL statement.
2. Compiler and linker : It is used for compiler Query. It help to object to existuable.

3. Application program : It is conversion of source code to object code.
4. DML compiler : It converts DML statement into low level language.
5. DML Query : It is DML Query.
6. Query evaluation engine : It execute or evaluate DDL, DML, DCL queries.

Storage Manager :
1. Buffer manager : It provide temperory memory for execution.
2. file manager : It manages all the files required from storage structure.
3. Transaction manager : It ensures consistency avoid failure for a transition.
4. Authorication Integrity manager : It helps to check user Authority & security.

Disk Storage :
1. Data : It is helps to stored data.
2. Indices : It provides fast access.
3. Data dictionary : It stores meta data (about data).
4. Statistical data : It keeps all the information about log files.



(16)

9) Define DBMS. List any two applications of DBMS.

→ DBMS : Database Management System (DBMS) is set of interrelated data & set of programs to access that data.

Applications of DBMS:
1. Bank
2. Hospital
3. Airlines
4. Universities
5. Human Resources

10) List any four DBMS software.

→ 1. My SQL
2. Oracle
3. Sybase
4. SQL Server
5. Mango DB
6. Informix

11) Enlist different components of DBMS.

→ 1. Database User :
i) Application Programmer
ii) Naive
iii) Database Administrator
iv) Sophisticated

2. Way of Interaction:
   i) Application interfaces
   ii) Application program
   iii) Query tools
   iv) Admin tools

3. Query Processor:
   i) Application program object code.
   ii) Compiler / Linker
   iii) DML Interpretor
   iv) DML Compiler
   v) DML Queries
   vi) Query evaluation engine

4. Storage Manager:
   i) Buffer Manager
   ii) file Manager
   iii) Authorization & integrity Manager
   iv) Transaction Manager

5. Disk Storage:
   i) Data
   ii) Indices
   iii) Data dictionary
   iv) Statistical data

12) Describe Relational model with example.
→ · It was introduced by Dr. E.F. Codd in 1970.
· Data is represented with the help of tables (Person, Place,

Things, Events, etc.)
- Relational model is most commonly used database model.
- It is more flexible than hierarchical and network model.
- A relation represent a particular entity. It is used to store information about entity.
- Row is also known as tuple
- Column is also known as attributes.

Example:

Data Table 1: Project Table

Project Number	Description	Dept. Number
155	Payroll	257
498	Widgets	632
226	Sales manager	598

Data Table 2: Department Table

Dept. Number	Dept. Name	Manager SSN
257	Accounting	421-55-99993
632	Manufacturing	765-00-3192
598	Marketing	098-40-1370

Data Table 3: Manager Table

SSN	Last Name	First Name	Hire Date	Dept. Number
005-10-6321	Johns	Francine	10-7-65	257
549-77-1001	Buckley	Bill	2-17-79	650
098-40-1370	Fiske	Steven	1-5-85	598

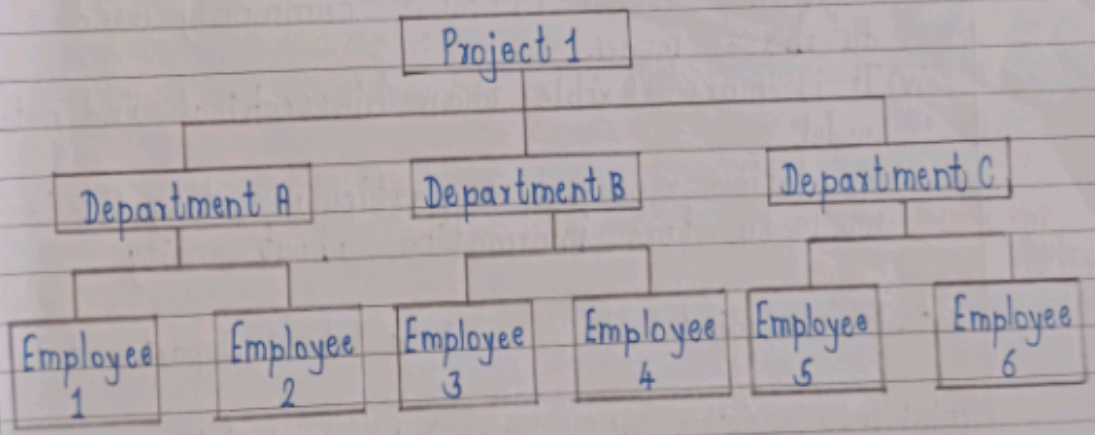
13) List & explain various Data models.

→ Types of Data models

1. Hierarchical Model
2. Network Model
3. Relational Model

1. Hierarchical Model

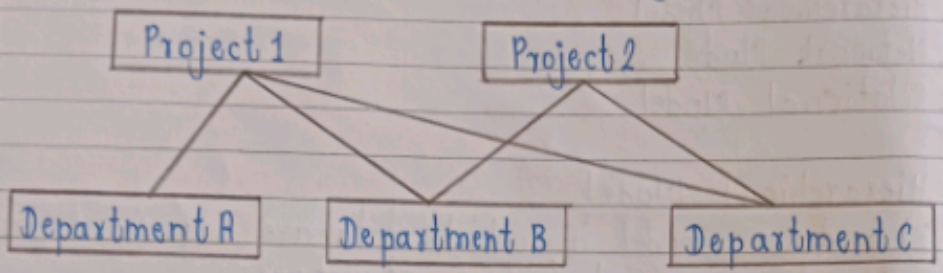
i) A data model in which data are organized in a top-down inverted tree structure.
ii) It represent parent child relationship structure.
iii) It organizes data in a tree structure.
iv) It allows 1:N Mapping between record types.



2. Network Model

i) An expansion of the hierarchical database model with an owner-member relationship in which a member may have many owners.
ii) It allows many to many relationship in data. Although 1:N or M:N relationship also permitted.
iii) It organizes data in the form of arbitrary graph.

iv) It consists of network diagram



3. Relational Model:

i) It was introduced by Dr. E.F. Codd in 1970.
ii) Data is represented with the help of tables.
iii) Relational model is most commonly used database model.
iv) It is more flexible than hierarchical and network model.
v) A relation represent a particular entity. It is used to store information about entity.

Data Table 1: Project Table

Project Number	Description	Dept Number
155	Payroll	257
498	Widgets	632
226	Sales manager	598

Data Table 2: Department Table

Dept Number	Dept. Name	Manager SSN
257	Accounting	421-55-9999
632	Manufacturing	765-00-3142
598	Marketing	098-40-1370

Data Table 3: Manager Table

SSN	Last Name	First Name	Hire Date	Dept. Number
005-10-6321	Johns	Francine	10-7-65	257
549-77-1001	Buckley	Bill	2-17-79	650
098-40-1370	Fiskex	Steven	1-5-85	598

14) List out query processing components and state their function.

→ 1. DDL Interpreter : It interpretes or resolve DDL statement.
2. Compiler & linker : It is used for compiler Query it help to object code/machine code.
3. DML Query : It is DML Query like insert, update, delete, select.
4. DML Compiler : It convert DML statement into low level language.
5. Application program object code : It is conversion of source code to object code / machine code.
6. Query evaluation engine : It is execute or evaluate DDL, DML, DCL & TCL Query.

15) List Codd rules. Explain any four.

→ 1. The information rule
2. The Guaranteed Access rule.
3. Systematic Treatment of NULL values.

4. Active Online Catalog Rule.
5. The Comprehensive Data Sublanguage rule.
6. The view updating rule.
7. High level insert, update and delete.
8. Physical data independence.
9. Logical data independence.
10. Integrity independence.
11. Distribution independence.
12. Non-Subversion independence.

Rule 1 : Information Rule :
The data stored in a database, may it be user data or metadata, must be a value of some table cell. Everything in a database must be stored in a table format.

Rule 6 : View Updating Rule:
All the views of a database, which can theoretically be updated, must also be updatable by the system.

Rule 8 : Physical Data Independence:
The data stored in a database must be independent of the applications that access the database. Any change in the physical structure of a database must not have any impact on how the data is being accessed by external applications.

Rule 12: Non-Subversion Rule:
If a system has a interface that provides access

to low-level records, then the interface must not be able to subvert the system and bypass security and integrity constraints.

16) Define Database. List any two properties of database.

→ Database : A database is an organized collection of data stored in a computer system and usually controlled by a Database Management System (DBMS).

Properties: 1. Data integrity
2. Data security
3. Concurrency control.

ASSIGNMENT - 2

1) What is the difference between weak entity set and strong entity set?

→

Weak entity set	Strong entity set
1. It depends on another strong entity.	1. It is independent in nature.
2. It does not have any primary key.	2. It has a primary key.
3. It is depicted by a dual rectangle.	3. It is depicted by a sole rectangle.
4. The connection between a weak and a strong entity is shown by a dual diamond.	4. The connection between two strong entities is shown by a sole diamond.

2) What are the types of attributes?

→ Types of attributes are

i. Simple Attributes
ii. Composite Attribute
iii. Single-Valued Attribute
iv. Multi-Valued Attribute
v. Derived Attribute
vi. Compiler Attribute

vii. Key Attribute

3) What is domain and entity?

→ Domain :

- A domain is the set of permitted values for one or more attributes.
- It defines the potential values that an attribute may hold.
- ~~for~~ It indicates area or field.
- It also indicates set of values of same data type.
- for e.g. If the age of the student in class is between 15 and 20, then we can define a set of values for the age attribute of student as the set of integers between 15 and 20.

Entity :

- An entity is a "thing" or "object" in the real world.
- An entity contains attributes, which describe that entity.
- So anything about which we store information is called an entity.
- for e.g. : A student, An employee, or bank a/c, etc. all are entities.

4) Define super key.

→ Super Key is defined as a set of attributes within a table that can uniquely identify each record within a table.

- Super key is a superset of Candidate key.

5) Define the term. i. Candidate key ii. Primary key

→ i. Candidate key:

- Candidate key in SQL is a set of attributes that uniquely identify tuples(rows) in a table.
- The candidate keys are as strong as the primary key.

ii. Primary key:

- Primary key in DBMS is a attribute or group of attributes (means column or group of columns) in a table that uniquely identify every row in the table.

6) Define attribute and entity.

→ Attribute:

An attribute is a property or characteristics of an entity it represents a specific detail or piece of information that describes an entity.

Entity:

In DBMS, an entity refers to real-world object or concept that can be distinctly identified & stored in a database. each entity has attributes that defines its properties or characteristics.

7) Explain single value and multivalue attribute of E-R model.

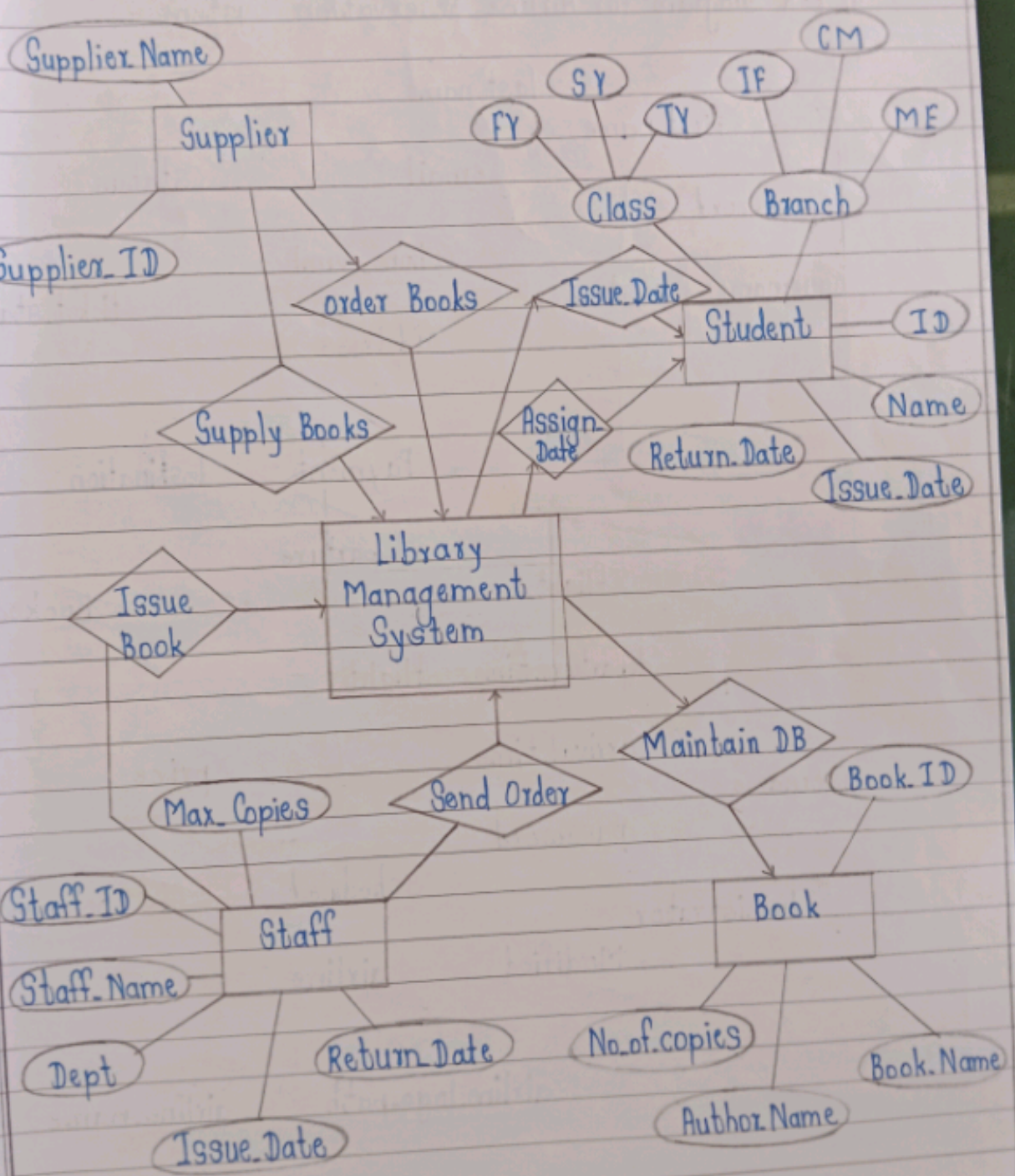
(21)

→ Single Value Attribute:
- A single value attribute is an attribute that holds a single value for each entity instance.
- e.g.: In a "Student" entity, the "Date of Birth" or "Student ID" attributes are single-value because each student can only have one birth date or ID.

Multi-value Attribute:
- A multi-value attributes is an attribute that can hold multiple values for a single entity instance.
- e.g.: In a "Student" entity, the "Phone Numbers" attribute could be multi-valued because a student might have more than one contact number.

8) Draw an E-R diagram of library management system.

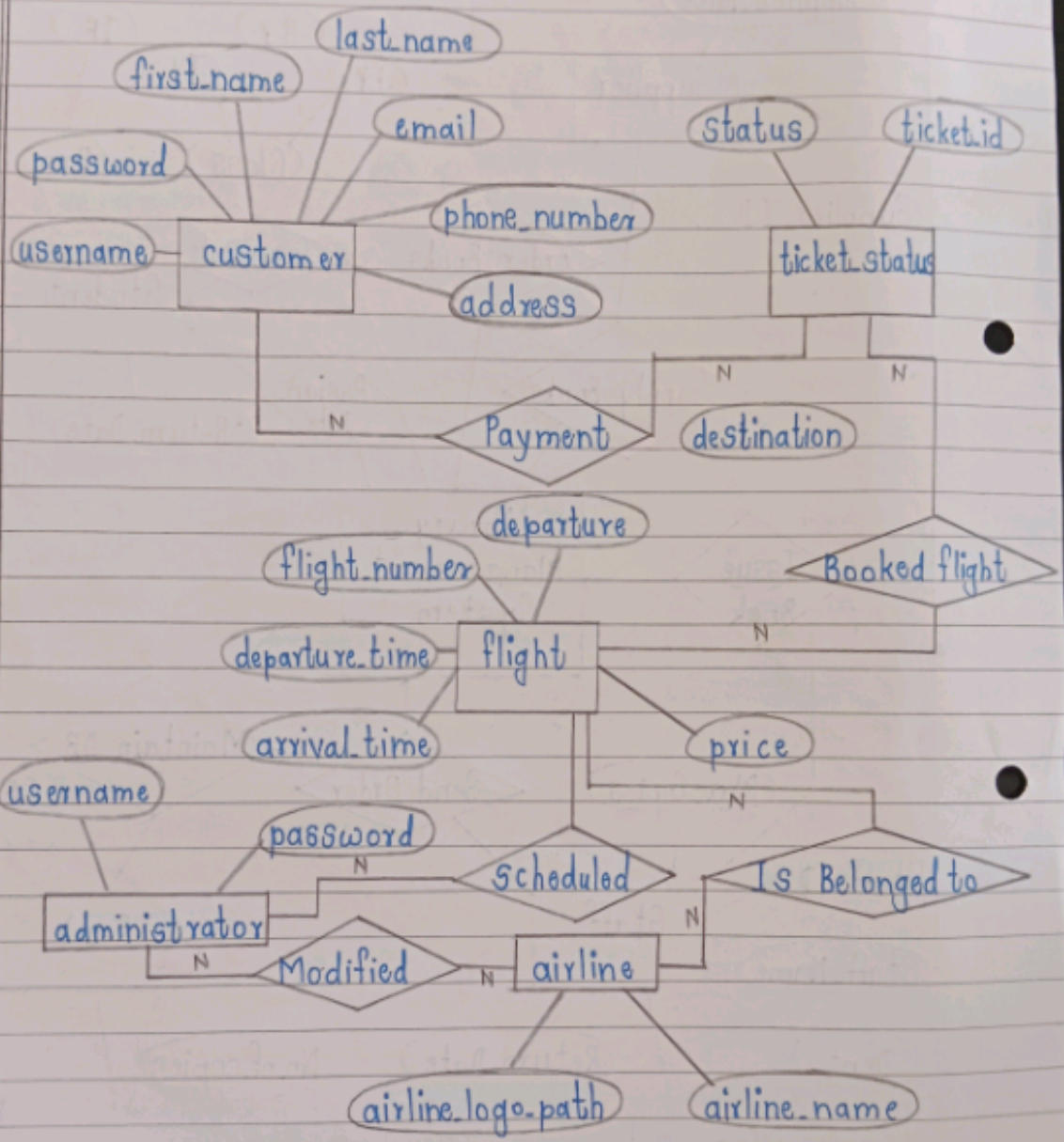
→



(23)

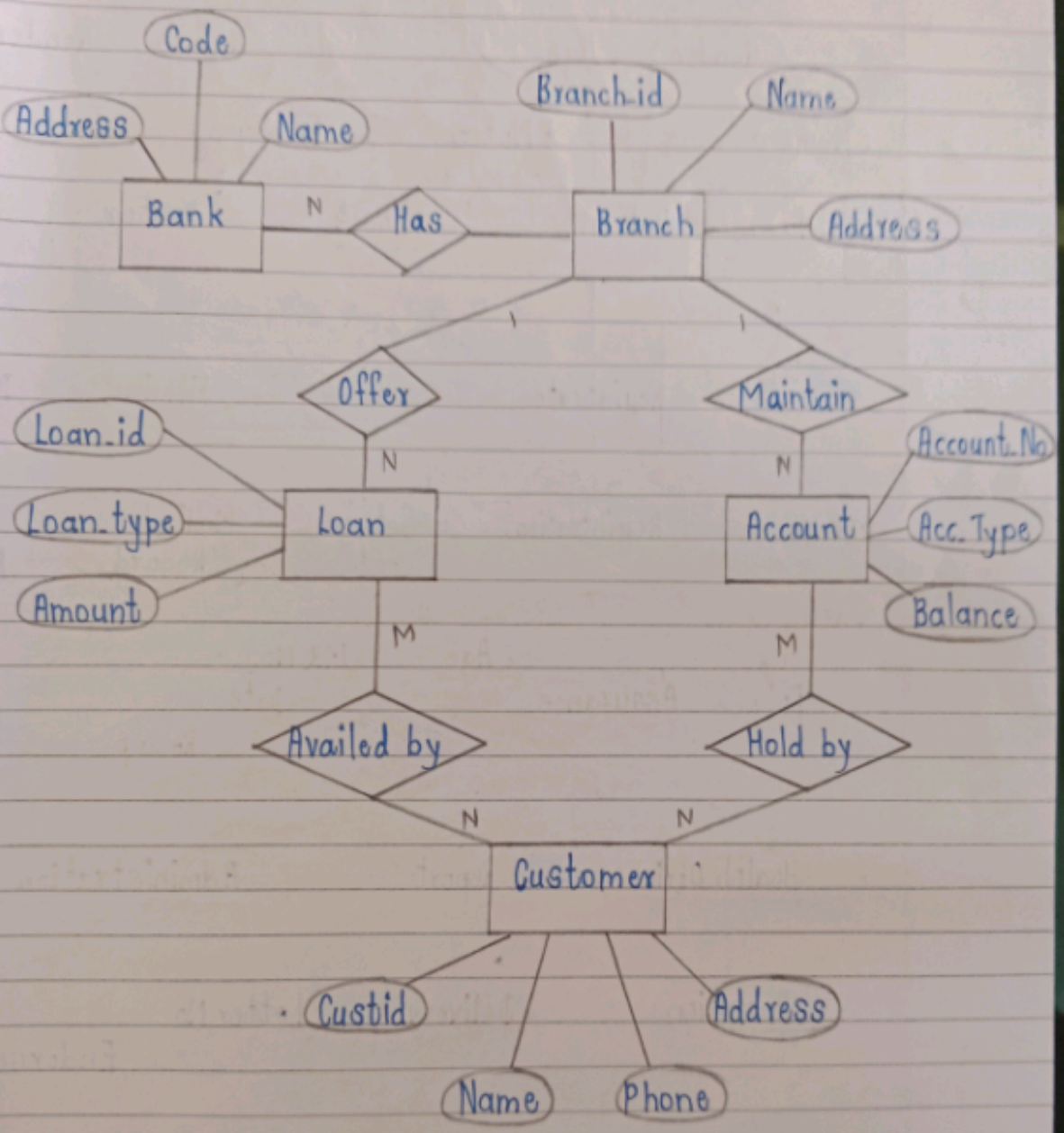
9) E-R Diagram for Airline Reservation System.

→



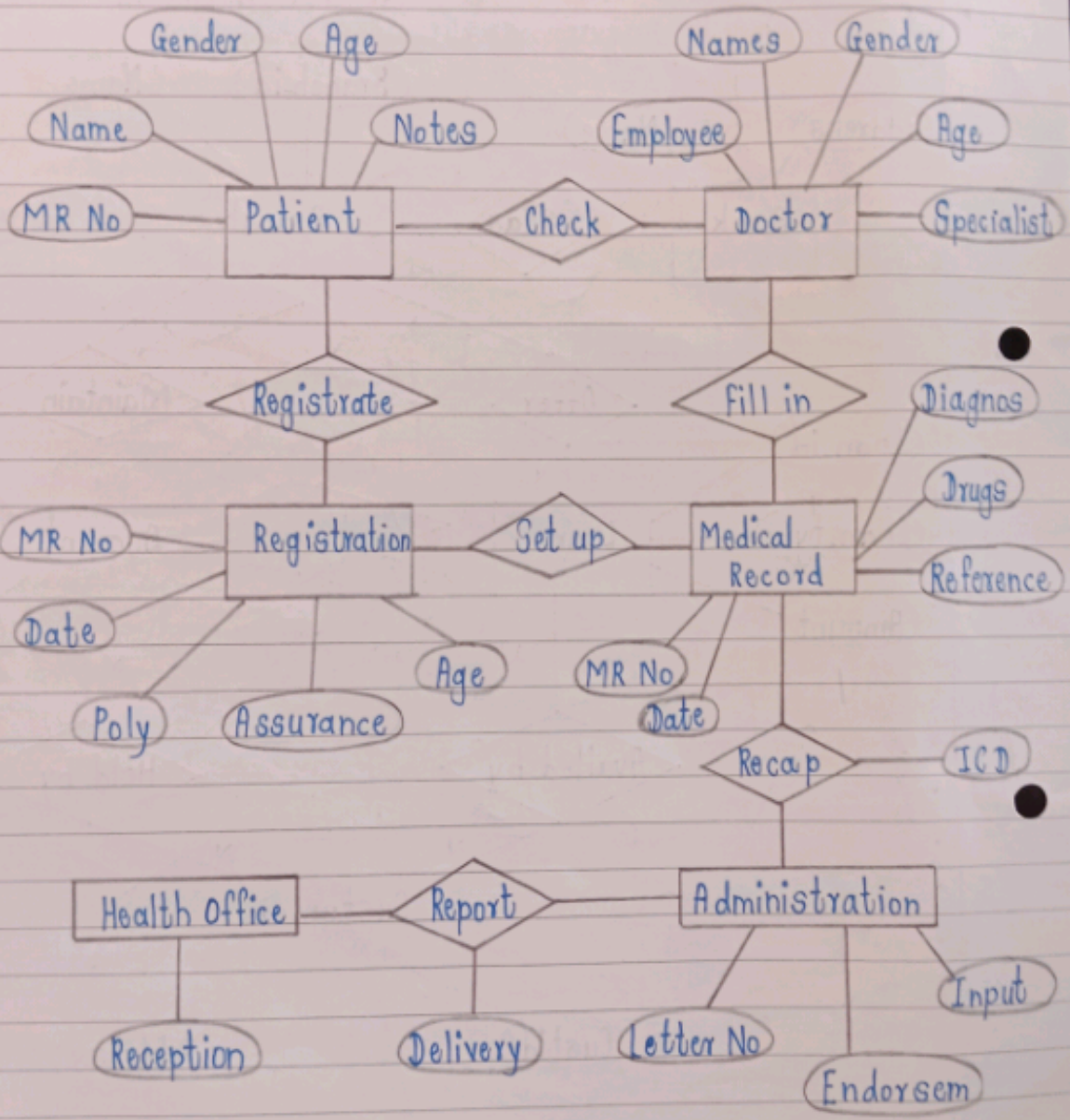
10) E-R Diagram for Customer & Loan System.

→



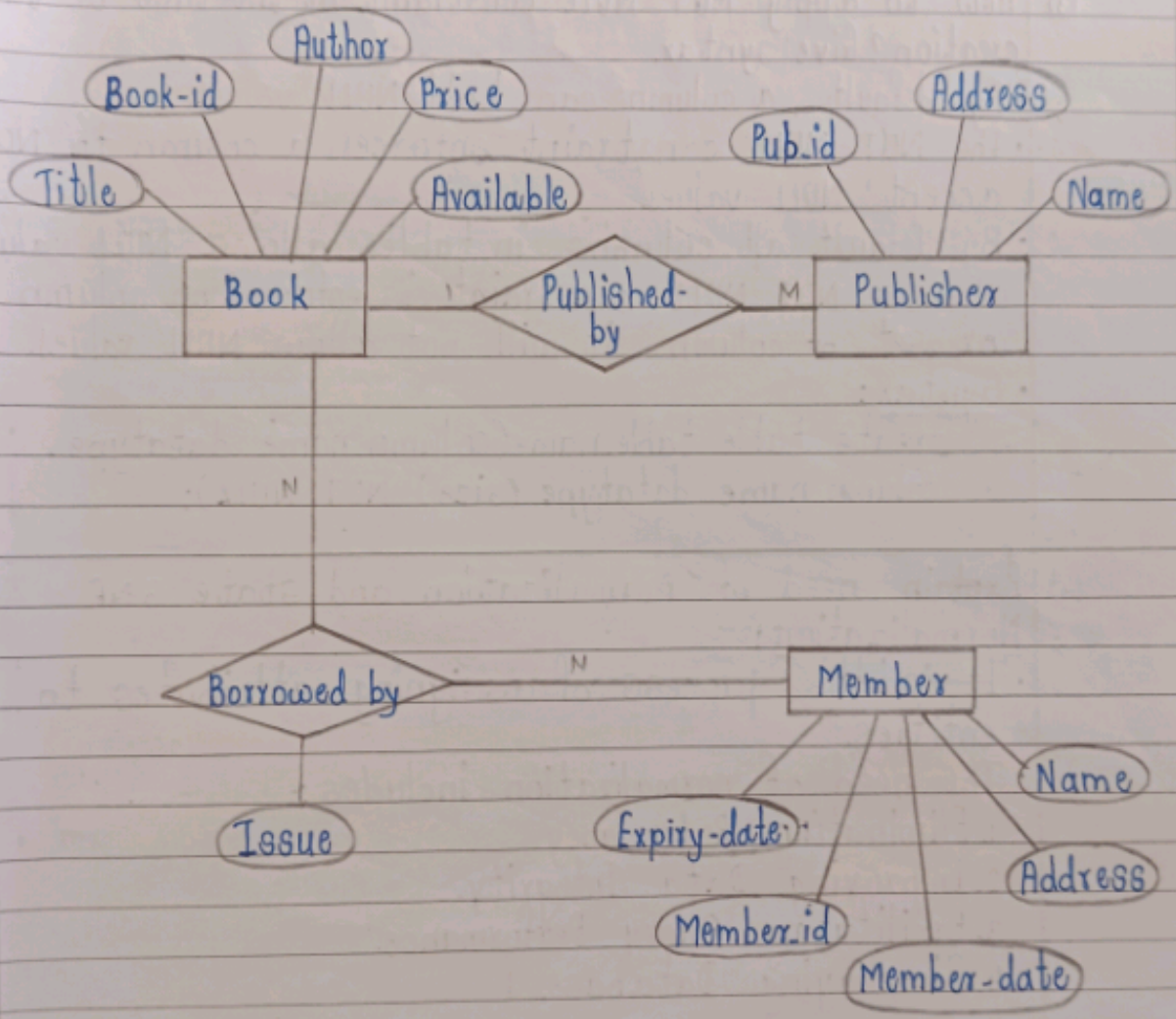
11) E-R Diagram for Hospital Management System

→



12) E-R Diagram for Library Management System.

→



13) How to apply NOT NULL constraint at the time of table creation? Give syntax.

→
- By default, a column can hold NULL values.
- The NOT NULL constraint enforces a column to NOT accept NULL values.
- By default all columns in tables allows NULL values when a NOT NULL constraint is enforced on column or set of column it will not allow NULL values.
- Syntax:

```
create table table_name(column_name datatype,
column_name datatype (size) NOT NULL);
```

14) Explain need of normalization and state 3NF.

→ Normalization:
- It is the process of assigning attributes to entities.
- The need of normalization includes:-
  1. Eliminating Redundancy
  2. Improving Data Integrity.
  3. facilitating Query Performance.
  4. Simplifying Database.

Three Normal form (3NF): it is a level of database normalization that builds upon the first too normal forms. A table is in 3NF if it meets the following criteria:-
1) it is in second normal form.
2) No transitive dependency.

15) State and explain 1NF and 2NF with example.

→ First Normal form (1NF)

- The term first normal form (1NF) describes the tabular format in which:
  - All the key attributes are defined.
  - There are no repeating groups in the table.
  - All attributes are dependent on the primary key.

First Normal form:

> This rule defines that all the attributes in a relation must have atomic domains.

Course	Content
Programming Web	Java, C++ HTML, PHP, ASP

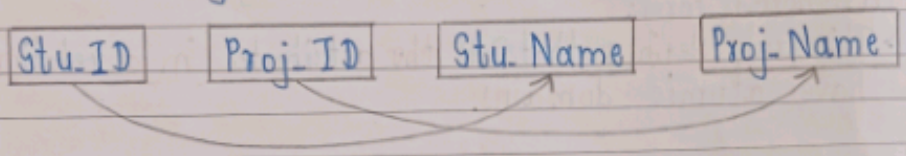
We re-arrange the relation (table) as below, to convert it to first Normal form

Course	Content
Programming	Java
Programming	C++
Web	HTML
Web	PHP
Web	ASP

Second Normal form (2NF)

- A table is in 2NF if:
  - It is in 1NF and
  - It includes no partial dependencies; that is, no attribute is dependent on only a portion of the primary key.

Student_Project



Student

Stu_ID	Stu_Name	Proj_ID

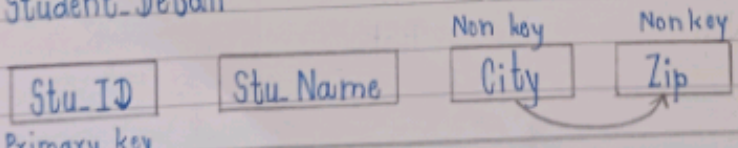
Project

Proj_ID	Proj_Name

16) Explain 3NF with example.

→ A table is in 3NF if:
- It is in 2NF and
- It contains no transitive dependencies.

Student_Detail



Relation not in 3NF

- We find that in above depicted Student_detail relation, Stu_ID is key and only prime key attribute.
- We find that City can be identified by Stu_ID as well as Zip itself. Neither Zip is a super key nor City is a prime attribute.
- Additionally, Stu_ID → City → Zip, so there exists transitive dependency.

Student_Detail

Stu_ID	Stu_Name	Zip

ZipCodes

Zip	City

We broke the relation as above depicted two relations to bring it into 3NF

17) Explain Normalization with example.

→
- Normalization is a process for assigning attributes to entities.
- It reduces data redundancies and helps eliminate the data anomalies.

Types of Normalization

1. First Normal Form (1NF)
2. Second Normal Form (2NF)

3. Third Normal form (3NF)

Example:
Student Table

Rno	Name	Branch	HOD	Telno
401	Akon	If	XYZ	53337
402	Bkon	If	XYZ	53337
403	Ckon	If	XYZ	53337
404	Dkon	If	XYZ	53337

- In the above table, we have data of 4 students.
- As we can see, data for the fields branch, hod (Head of Department) and Telno is repeated for the students who are in the same branch in the college, this is Data Redundancy.

18) Explain domain integrity constraint with example. (Not Null, Check)

→ Domain Constraint:
Domain constraint defines the domain or set of values for an attribute.
- There are 2 constraints which we can study under domain constraint.
  i. Not null constraint.
  ii. Check constraint.

i. Not Null Constraint:

- By default, a column can hold NULL values.
- The NOT NULL constraint enforces a column to NOT accept NULL values.
- By default all columns in tables allows null values. When a NOT NULL constraint is enforced on column or set of columns it will not allow null values.

Syntax:

create table table_name(column_name datatype, column_name datatype NOT NULL);

Example:

SQL> create table student (roll_no number (5), name varchar2 (20) NOT NULL);

ii. Check Constraint:

- The constraint define a condition that each row must satisfy.
- A single column can have multiple check condition.

Syntax:

create table table_name (column_name datatype, column_name datatype CONSTRAINT CONTRAINT_NAME CHECK < CONDITION>);

Example:

SQL> create table emp (id number(5), name varchar2 (10), sal number(10) constraint chk_sal check (sal > 10000));

19) List and explain the types of integrity constraints in detail.

→ Types of Integrity Constraints :

1. Primary Key Constraint
2. Unique Constraint
3. Foreign Key Constraint.
4. Check
5. NULL & NOT NULL

1. Primary Key Constraint :

It is use to avoid redundant / duplicate value entry within the row of specified column in table. It restricts null values too.

Syntax :

```
create table table_name (column_name datatype(size)
constraint constraint_name primary key, column_
name data_type(size));
```

2. Unique Constraint :

The UNIQUE constraint uniquely identifies each record in a database table. The UNIQUE and PRIMARY KEY constraints both provide a guarantee for uniqueness for a column or set of columns.

Syntax :

```
create table table_name(column_name datatype(size)
constraint constraint_name unique, column_name
datatype(size));
```

3. Foreign Key Constraint:

A foreign key (FK) is a column that is used to join a table to other tables to ensure referential integrity of the data. A foreign key constraint requires that a column contain only values from the primary key column on a specific dimension table.

Syntax:

create table table_name(column_name datatype(size) constraint constraint_name foreign key(Parent table column) references Parent-table name(Parent-table column));

4. Check:

- The constraint defines a condition that each row must satisfy.
- A single column can have multiple check condition.

Syntax:

create table table_name(column_name datatype, column_name datatype constraint constraint_name check <condition>);

5. NULL & NOT NULL:

- By default, a column can hold NULL values.
- The Not Null ~~colum~~ constraint enforces a column to Not accept Null values.
- By default all columns in tables allows null values. When a Not Null constraint is enforced on column or set of columns it will not allow null values.

Syntax:

```
create table table_name (column_name datatype,
column_name datatype NOT NULL);
```

ASSIGNMENT - 3

1) Explain Alter command. Give syntax of add and modify option.

→ By the use of Alter Table command we can modify our existing table.
It has three types -
- Adding New Columns
- Dropping a Column from the Table
- Modifying Existing Table.

Alter add:
It is used to add columns in an existing table.
Syntax:
alter table table_name add (new column_name, datatype (size));
Example:
alter table student add (age number (2), marks number (3));

Alter modify:
It is used to change / modify size and datatype of column.
Syntax:
alter table table_name modify (column_name new datatype (new size));
Example:
alter table employee modify (eid number (4));

2) List any four DDL commands.

→ 1. Create
2. Alter
3. Drop
4. Truncate
5. Rename

3) Describe Grant and Revoke commands.

→ Grant Command:

A DBA or user can grant access permission on owned database objects to other user or roles using GRANT command i.e. Owner of database allows user to do operation.

Syntax:

```
Grant [privilege]
ON [object]
TO {user|public|role} [with admin|grant option];
```

Example:

```
SQL> grant create session to u1;
```

Revoke Command:

The DCL command is used to revoke an existing privilege from a user. It can revoke a system privilege, object privilege or a role from a user. It take back the right offered to user by owner of database.

Syntax:

```
Revoke [privilege]
On [object]
```

From {user|public|role}

Example:

SQL> revoke select on T1 from Anil;

4) Describe Commit and Rollback with syntax.

→ Commit Command:

The commit command is used to save changes invoked by a transaction to the database. The commit command saves all the transactions to the database since the last commit or rollback command.

Syntax:

```
COMMIT;
```

Rollback Command:

The Rollback command is the transactional command used to undo transactions that have not already been saved to the database. This command can only be used to undo transactions since the last command commit or rollback was issued.

Syntax:

```
ROLLBACK;
```

5) Explain the use of truncate statement. Give example.

→ The truncate table statement is used to delete the data inside a table, but not the table itself.

Syntax:
truncate table tablename;
Example:
trancate table supplier;

6) Write syntax of insert command. Demonstrate with suitable example.

→ Syntax with example.

1. Insert into customers (ID, NAME, AGE, ADDRESS, SALARY) values (1, 'Ramesh', 32, 'Ahmedabad', 20000);
2. Insert into customers (ID, NAME, AGE, ADDRESS, SALARY) values (2, 'Kapil', 25, 'Delhi', 1500.00);

1] Specify both the columns names and the values to be inserted:
INSERT INTO table_name (column 1, column 2, column3,...)
values (value 1, value 2, value 3, ...);
example:
INSERT INTO CUSTOMER (ID, NAME, AGE, ADDRESS, SALARY)
values (1, 'Ramesh', 32, 'Ahmedabad', 2000.00);

2]
Syntax: INSERT INTO table name
values (value 1, value 2 ...);
Example: INSERT INTO CUSTOMERS
values (7, 'Arrow', 24, 'Nagar', 10000);

7) List and explain set operators in SQL with example.

→ Set Operators:

1. Union

2. Union All
3. Intersection
4. Minus

1. UNION:
UNION is used to combine the results of two or more select statements eliminating duplicated.
Syntax: select column_name from table1 union select column_name from table 2;
Example: select dept_no from emp union select deptno from dept;

2. UNION ALL:
UNION ALL is used to combine both table including duplicates which are present in both.
Syntax: select column_name from table1 : union all select column_name from table 2;
Example: select ename from emp1 union all select ename from emp2;

3. Intersection:
The intersection of two sets includes elements which are present in both.
Syntax: select column_name from table 1 intersect select column_name from table 2;
Example: select ename from emp1 intersect select ename from emp2;

4. Minus:
The minus of two sets includes elements from set1 minus elements of set2.

Syntax: select column_name from table1 minus
select column_name from table2;
Example: select ename from emp1 minus select
ename from emp2;

8) List and explain any 4 arithmetic operators in SQL with example.

→ Arithmetic Operators:

1. + (Addition)
2. - (Subtraction)
3. * (Multiplication)
4. / (Division)

Addition (+):

Adds values on either side of the operator.

Syntax: select <expression> [arithmetic operator] <expression> from [table_name] where [expression];

Example: select 10 + 5 AS additionresult;

Subtraction (-):

Subtracts right hand operand from left hand operand.

Syntax: select <expression> [arithmetic operator] <expression> from [table_name] where [expression];

Example: select 10 - 5 AS subtractionResult;

Multiplication (*):

Multiplies values on either side of the operator

Syntax: select <expression> [arithmetic operator]

<expression> from [table.name] where [expression];

Example: select 10 *5 AS multiplicationResult;

Division(/):

Divides left hand operand by right hand operand.

Syntax: select <expression> [arithmetic operator] <expression> from [table_name] where [expression];

Example: select 10 / 5 AS divisionResult;

9) Explain any four string functions with example.

→ String functions

1. Lower (str)
2. UPPER (str)
3. LENGTH (str)
4. LTRIM(str)

LOWER (string):

Returns the string str with all characters changed to lowercase according to the current character set mapping.

Example:

SQL > select lower ('QUADRATICALLY');

output: quadraitically.

UPPER (string):

Returns the string str with all characters changed to uppercase according to the current character set mapping.

Example:

SQL> select upper ('Hello World');

Output: HELLO WORLD

LENGTH(string):
Returns the length of the string str, measured in bytes.
Example:
SQL> select length('text');
Output: 4

LTRIM(string):
Returns the string str with leading space characters removed.
Example: Select LTRIM('123Vikas','123') from dual;
Output: Vikas

10) Describe date and time function.

→ 1. sysdate():
It is used to return system date.
Syntax / Example: select sysdate from dual;
Output: 21-SEP-24

2. next_day(date, day):
It is used to display date for specified day as per given day.
Syntax: select next_day(date, day) from dual;
Example: SQL> select nex_day(sysdate,'tuesday') from dual;
Output: 24-SEP-24

3. month_between(date1, date2):
It is used to return months between the given

two specied table.
Syntax: select months_between (date1, date2) from dual;
Example: SQL> select months_between ('22-SEP-24', '22-SEP-25')
from dual;
Output : 12

4. add_months (date, no_of_months):
It is used to add specific months to specific given date.
Syntax:- select add_months (date, no_of_months) from dual;
Example:- SQL> select add_months ("22-SEP-24', '2') from dual;
Output :- 22-NOV-24

5. last_day (date):
It is used to return the last day of given specified data.
Syntax: select last_day ('date') from dual;
Example: SQL> select last_day ('24-SEP-24') from dual;
Output : 30-SEP-24

11) Explain any four aggregate functions with their examples.
→ 1. AVG() - Returns the average value
Syntax: select Avg(colname) from table_name;
Example: Display the average salary from emp table.
SQL> select avg(sal) from emp;
2. COUNT() - Returns the number of rows.
Syntax: select count (colname) from table_name;

Example: Display the total number of employee working in the company.
SQL > select count(ename) from emp;

3. MAX() - Returns the largest value.
Syntax: select max(col_name) from table_name;
Example: Display the maximum salary from emp table.
SQL> select max(sal) from emp;

4. SUM() - Returns the sum.
Syntax: select sum(col_name) from table_name;
Example: Display the total salary paid to all employees.
SQL > select sum(sal) from emp;

12) What is sequence? What are various operations with respect to sequences?

→ A sequence is a user-defined schema bound object that generates a sequence of numeric values according to the specification with which the sequence was created.

Creating Sequence:
Sequence can be created using create sequence command.
Example: Create Sequence Seq1 As Start with 110 Increment by 5 Maxvalue 200 Cycle No cache

Using a Sequence:
To insert record in table using sequence.

Example: Insert into product values(Seq1.nextval, 'Laptop', 'HP', 89);

Altering Sequence:

Sequence can be Altered by using Alter Command.

Example: SQL> Alter Sequence Seq1 As Start with 110 increment by 2 maxvalue 300 cycle no cache;

Deleting Sequence:

Sequence can be Deleted by using Drop Command.

Example: SQL> Drop Sequence Seq1;

13) What are sequences? Create sequence for 'student' table.

→ A sequence is a user defined schema bound object that generates a sequence of numeric values according to the specification with which the sequence was created sequence.

```
SQL> create Sequence student_id_seq
     start with 1
     increment by 1
     no minvalue
     no maxvalue
     cache 1;
```

14) Describe following terms: i. unique indexes ii. composite indexes.

→ i. Unique Indexes:

Unique index applied on single column, not allows duplication.

Example: SQL> create index idx2 on order (product_id);

ii. Composite Indexes:

Composite index applied on multiple column.

Example: SQL> create index idx3 on order (supplier_name, unit_price);

15) Explain GROUP BY, ORDER BY clause of SQL with example.

→ GROUP BY CLAUSE:

The GROUP BY statement is used in conjunction with the aggregate functions to group the result-set by one or more columns.

Syntax: Select column_name, aggregate_function (column_name) from table_name where column_name operator value GROUP BY column_name;

Example: Display name of each department and maximum salary in the department.

SQL> select department, max(Salary) from employees GROUP BY department.

ORDER BY CLAUSE:

- The ORDER BY keyword is used to sort the result-set.
- The ORDER BY keyword sorts the records in ascending order by default.

- To sort the records in a descending order, you can use the Desc keyword.

Syntax: select column_name(s) from table_name order by column_name ASC / DESC;

Example: SQL> select * from custombers order by country;

16) Explain Having clause with example.

→ The HAVING clause was added to SQL because the WHERE keyword could not be used with aggregate functions.

Syntax: Select column_name, aggregate_function(column_name) from table_name WHERE column_name operator value GROUP BY column_name HAVING aggregate_function(column_name) operator value;

Example: Display only those departments where the minimum salary is less than 42,000.

SQL> Select department, Min(Salary) from employees Group by department Min (Salary) < 42000;

17) Explain word comparision operators:
i. IN and NOT IN ii. BETWEEN and NOT BETWEEN

→ i. IN and NOT IN

IN - it satisfy more than two conditions at a time.

Syntax: select column from table_name where column in(cond^n 1, cond^n 2, cond^n 3);

Example: SQL> select * from emp where dno in(10,20,30);

- NOT IN - This operator checks if a value does not exist within a specified list of values.

Syntax: select column from table_name where column not in (cond^{n}1, cond^{n}2, cond^{n}3);

Example: SQL> select * from emp where dno not in (10, 20, 30);

ii. BETWEEN AND NOT BETWEEN

- BETWEEN - The between operator searches record withing given range.

Syntax: select column from table_name where col_name between cond^{n}1 and cond^{n}2;

Example: SQL> select * from emp where esal between 10000 and 50000;

- NOT BETWEEN - Not between operator searches record outside the given range.

Syntax: select column_name from table_name where column_name not between cond^{n}1 and cond^{n}2;

Example: SQL> select * from emp where esal not between 1000 and 5000;

12) What are synonyms? How to create and drop synonym?

→ Synonym:

- It is an alternative name for a table, view, sequence, procedure, stored function, package, or another synonym.

- Synonyms provide both data independence and location transparency.

To create synonym:
SQL> create synonym syn_nm from table_name;
Example: Create synonym S1 from emp;

To drop synonym:
SQL> drop synonym S1;

19) State use of '%' character in string operations.
→ The '%' character is used as a wildcard in SQL for pattern matching in queries with the LIKE operator. here's how it is used:
1. Matches any sequence of characters (including none)
Examples:
- 'A%' matching any string starting with 'A'.

20) Explain the set operator with the help of example.
→ 1. Union: It combines a data from both relation but it does not take duplication.
Example: Select A from AB union select B from BA;

AB		BA		output:
A		B		1
1		1		2
2		3		3
3		5		4
4		8		5
5		9		6
				7

(5)

7	10	8
7	5	9
		10

2. Union All : It combines a data from both relation but it takes duplication.

Syntax/Example : select column1, column2 from table1 union all select column1, column2 from table2;

Example: SQL>select name from students union all select name from teachers;

3. Intersect : Returns only the rows that are common to both select queries.

Example : SQL>select name from students intersect select name from teachers;

4. Minus: Returns the rows from the first select query that are not in the result of the second select query.

Example : SQL> select name from students minus select name from teachers;

21) Explain view with example. How to create view? Give its syntax and explain its advantages.

→ Views :

- Views changes appearance of data
- A view is a named result of a query.
- A view is a snapshot relation.
- Views can be used in other queries and view definitions.

Creating a View

Database views are created using the create view statement.

Syntax: create view view_name AS select column_name(s) FROM table_name where condition;

Example: SQL> create view vw1 AS select product_id, product_name from product where product_id > 101;

Advantages:

1. A view is actually a composition of a table in the form of a predefined SQL query.
2. A view can contain all rows of a table or select rows from a table.
3. A view can be created from one or many tables.

22) What is OUTER JOIN? Explain in detail.

→ Outer join condition returns all rows from both tables which satisfy the join condition along with rows which do not satisfy the join condition from one of the tables.

Syntax: select col_name1, col_name2, col_name3, from table1 Outer Join Table2 where join condition;

Example: select eid from emp outer join stud where e.eid = S.eid;

Types of Outer Join:

1. Left Outer Join
2. Right Outer Join
3. Full Outer Join

23) Define index. State its type.

→ • These are schema objects that can speed access to table rows, and index-organized tables, which are tables stored in an index structure.
• An index is an optional structure, associated with a table or table cluster, that can sometimes speed data access.

Types of Index:

1. Simple Indexes: Applied on single column allows duplication.

```
SQL> create index idx1 ON order (product_name);
```

2. Unique Indexes: Applied on single column, not allows duplication.

```
SQL> create index idx2 ON order(product_id);
```

3. Composite Indexes: Applied on multiple column

```
SQL> create index idx3 ON order(supplier_name,
     unit_price);
```

24) Differentiate between view and index.

→

View	Index
1. virtual table based on query.	1. Data structure to improve search speed.
2. No physical storage, dynamically generated.	2. physically stored in the database.

3. Simplify complex queries.	3. Speed up data retrival operations.
4. Created using create view command.	4. Created using create index command.
5. Used for security, abstraction.	5. Used for quick search and sorting.

25) With the help of example, explain DROP VIEW command.

→ Obviously, where you have a view, you need a way to drop the view if it is no longer needed. The syntax is very simple and is given below.

Drop view view_name;

Following is an example to drop the customers view from the customers table.

Example: DROP VIEW CUSTOMERS VIEW;

ASSIGNMENT - 4

1) What are the various control structure statements used in PL/SQL?

→ Control Statements used in PL/SQL.
- Conditional Control.
- Iterative Control.
- Sequential Control.

- Conditional Control
  1. IF-THEN
  2. IF-THEN-ELSE
  3. IF-THEN-ELSE-IF
- Iterative Control
  1. LOOP
  2. WHILE LOOP
  3. FOR LOOP
- Sequential Control
  1. GOTO Statement.

2) Explain Conditional Control Structure in PL/SQL.

→ Conditional Control Statements are those which allow you to control the execution flow of the program depending on a condition.
- IF-THEN : Executes a sequence of statements if a condition is true.

Syntax: IF <condition> THEN...
statement
END IF;

- IF-THEN-ELSE: If condition is True then first block executed else condition false then Second Block is executed.

Syntax: IF <condition> THEN...
<statement list 1>
ELSE <condition 2>
<statement 2>
END IF;

- IF-THEN-ELSIF: If-then-elsif statement is test multiple condition, if any condition true then that condition Block is executed, if all condition are fail then finally else Block is executed.

Syntax: IF <condition 1> Then
<statement list 1>
ELSIF <condition 2> Then
<statement list 2>
ELSIF <condition n> Then
<statement list n>
ELSE
<statement list>
END IF;

3) Explain FOR LOOP in PL/SQL with example.

→ - In for loop we can Specify the Range of Integer Number.
- for loop Executed upto the Specified range.

- The General Syntax to write a FOR LOOP is:

Syntax:

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

- Example: To display 1 to 5 number using for Loop.

```
1. declare
2 a number := 1;
3 begin
4 for a in 1..5 loop
5 dbms_output.put_line(a);
6 end loop;
7 end;
```

4) Explain WHILE LOOP in PL/SQL with example.

→ - In While Loop Condition is Checked at the Begin of the loop.
- If Condition is false then it Comes out From loop, it Execute the Loop till Loop Condition is True.
- The General Syntax to write a WHILE LOOP is:

Syntax:

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

- Example: Display 1 to 5 number using while loop.

```
1 declare
2 a number:=1;
3 begin
```

```
4  while a<=5 loop
5  dbms_output.put_line(a);
6  a:=a+1;
7  end loop;
8  end;
```

5) What are exceptions? What are Predefined exceptions and User defined exceptions? Explain its types.

→ An exception is an event that occurs during the execution of a program that disrupts the normal flow of the program's instruction.

Types of Exception

1. Predefined Exception
2. User defined Exception

1. Predefined Exception: System exceptions are automatically raised by Oracle, when a program violates a RDBMS rule. There are some system exceptions which are raised frequently, so they are pre-defined and given a name in Oracle which are known as Named System Exceptions.

Example:-

Suppose a NO_DATA_FOUND exception is raised in a proc, we can write a code to handle the exception as given below.

```
begin
Execution section
Exception When no_data_found then
dbms_output.put_line('A SELECT... INTO did not return any row');
end;
```

2. User defined Exception: Apart from system exceptions we can explicitly define exceptions based on business rules. These are known as user-defined exceptions.

Steps to be followed to user-defined exceptions:
- They should be explicitly declared in the declaration section.
- They should be explicitly raised in the Execution Section.
- The should be handled by referencing the user-defined exception name in the exception section.

Example:

```
declare
exception name exception;
begin
If condition then
 RAISE exception_name;
end if;
EXCEPTION
 when exception_name Then
 statement;
end;
```

6) What is cursor? Explain Implicit and Explicit cursor.

→ - A cursor is a handle, or pointer, to the context area.
- Through the cursor, a pl/SQL program can control the context area and what happens to it as the

statement is processed.

- Two important features about the cursor are
  - Cursor allow you to fetch and process rows returned by a SELECT statement, one row at a time.
  - A cursor is named so that it can be referenced.

Types of Cursors

There are two types of cursors:

1. Implicit Cursor
2. Explicit Cursor

1. Implicit Cursor: An Implicit cursor is automatically declared by Oracle every time an SQL statement is executed. The user will not be aware of this happening and will not be able to control or process the information in an implicit cursor.

Cursor Attributes:

- %FOUND
- %NOTFOUND
- %ROWCOUNT

Example: Example of SQL%FOUND & SQL%NOTFOUND

```
declare
total_rows number(2);
begin
UPDATE customers
SET salary = salary + 500;
If sql%notfound then
dbms_output.put_line('no customers selected');
```

```
ELSIF sql%found then
  total_rows := sql%rowcount;
  dbms_output.put_line(total_rows||'customers
  selected');
end if;
end;
/
```

2. Explicit cursor: An explicit cursor is defined by the program for any query that returns more than one row of data. That means the programmer has declared the cursor within the PL/SQL code block.

Cursor Attribute:

1. ISOPEN

Operations: 1. Open Cursor
2. Fetch Records
3. Close Cursor

- Declaration of Cursor:

```
Cursor Cur-name is Select Statement;
```

- Opening a Cursor:

```
OPEN cursor-name;
```

- Fetching from Cursor

```
FETCH cursor_name INTO record_name;
```

- Closing cursor

```
CLOSE cursor_name;
```

Example: Explicit Cursor to access emp table.

```
declare
cursor c2 is Select empno, salary from emp
Where deptno = 10;
```

```
ecode emp.empno%Type;
sal emp.salary%Type;
begin
Open c2;
If c2%ISOPEN then
Loop
Fetch c2 into ecode, sal;
If c2%FOUND then
Update emp set salary = salary + 500;
   Else
   Exit;
End if;
End loop;
Close c2;
Else dbms_output.put_line('unable to open');
End if;
End;
```

7) Explain parameterized cursor with example.

→ - We can pass parameters into a cursor and use them in the query.
- We can only pass values to the cursor; and cannot pass values out of the cursor through parameters.
- Example:

```
declare
  cursor c(no number) is select * from emp_information
  where emp.no = no;
  tmp emp_information%rowtype;
```

```
begin
  OPEN c(4);
  FOR tmp IN c(4) LOOP
  dbms_output.put_line('EMP_No :' ||tmp.emp_no);
  dbms_output.put_line('EMP_Name:' ||tmp.emp_name);
  dbms_output.put_line('EMP_Salary:' ||tmp.emp_salary);
  END loop;
CLOSE c;
END;
```

8) Write step by step syntax to create, open and close cursor in PL/SQL block.

→ Step 1: Declaring a Cursor at Declare Section :

```
Cursor Cur-name is Select Statement;
```

Step 2: Opening a Cursor

```
OPEN cursor_name;
```

Step 3: fetching from Cursor

```
FETCH cursor_name INTO record_name;
```

Step 4: Closing cursor

```
CLOSE cursor_name;
```

9) Explain trigger with suitable example. List its types. How to create trigger? State any two advantages of trigger. Compare database triggers and procedures and explain the use of database trigger.

→

A trigger is a special type of stored procedure in a database that automatically executes or fires when certain events occur. Triggers are used to enforce business rules, validate input data, and maintain audit trails.

Example: The following statement creates a trigger for the Emp table.
The trigger is fired when DML operations (INSERT, UPDATE, and DELETE statements) are performed on the table.

```
CREATE OR REPLACE TRIGGER tr1
BEFORE DELETE OR INSERT OR
UPDATE ON Emp
FOR EACH ROW
WHEN (new.Empno >0)

DECLARE
sal_diff number;
BEGIN
sal_diff := new.sal - old.sal;
dbms_output.put_line ('old salary :' ||old.sal);
dbms_output.put_line('New salary :' ||new.sal);
dbms_output.put_line('Difference' ||sal_diff);
END;
/
```

Types of Triggers:
- DML Triggers
- DDL Triggers
- System / Database Event Triggers
- Instead-of Triggers
- Compound Triggers
- AFTER INSERT Trigger
- AFTER UPDATE Trigger
- AFTER DELETE Trigger
- BEFORE INSERT Trigger
- BEFORE UPDATE Trigger
- BEFORE DELETE Trigger

Create Trigger.

Syntax:
```
CREATE [OR REPLACE] TRIGGER trigger_name
BEFORE | AFTER
[INSERT, UPDATE, DELETE [COLUMN NAME.]
ON table_name
Referencing [OLD AS OLD | NEW AS NEW]
FOR EACH ROW | FOR EACH STATEMENT
[WHEN Condition];
```

Advantages of trigger.

1. Automatic Enforcement of Business Rules: Triggers can automatically enforce complex business rules, ensuring data integrity and consistency without requiring additional application logic.
2. Auditing and Logging: Triggers can automatically log

changes to the database, providing a history of modifications for auditing purposes.

Compare database triggers and procedures.

Database Trigger	Procedure
1. Triggers are fired when particular SQL command (DML) is executed.	1. Procedures are executed when they are called.
2. Triggers have event and actions.	2. Procedures do not have event and actions.
3. Triggers are called implicitly.	3. Procedures are called explicitly.

Use of database triggers:

1. Triggers can be used to prevent invalid transaction and to apply complex security authorization.
2. It can be used to various integrity constraint.
3. It can be used to check data modification.

10) Differentiate between function and procedure. List two advantages of each.

→

Aspect	Function	Procedure
Definition	A function is used for calculating result using given inputs.	A procedure is used to perform certain task in order.
Call	A function can be called by a procedure.	A procedure cannot be called by a function
DML	DML statements cannot be executed within a function.	DML statements can be executed within a procedure.
SQL Query	It can be called in a query.	It cannot be called in a query.
try-catch	It does not support try-catch blocks.	It supports try-catch blocks.
SELECT	There can be a function call in a SELECT statement.	There is no procedure call in a SELECT statement.

Advantages of functions
1. Reusability
2. Modularity.

Advantages of Procedures
1. Flexibility
2. Performance

11) Draw the block structure of PL/SQL. List advantages of PL/SQL.

→ PL/SQL Block Structure

```
DECLARE
    /* Declarative section: variables, types and local
       subprograms */
BEGIN
    /* Executable section: procedural and SQL statements
       go here. */
    /* This is the only section of the block that
       is required. */
EXCEPTION
    /* Exception handling section: error handling
       statements go here. */
    /* It is optional section */
END;
```

Advantages of PL/SQL
- Procedural language support.
- Reduces network traffic.

- Error handling
- Declare variable
- Intermediate Calculation
- Portable application.

12) What statement is used in PL/SQL to display the output?

→ To display output in PL/SQL, you can use the dbms_output.put_line statement.

In PL/SQL, the statement used to display the output is the dbms_output.put_line procedure. This procedure is part of the dbms_output package, which allows you to send messages from PL/SQL blocks, subprograms, and triggers to the standard output.

Example:

```
Set serveroutput on;
declare
v_message varchar2(50);
begin
v_message :='This is a test message.';
dbms_output.put_line(v_message);
end;
/
```

13) List out any four statements of PL/SQL.

→ .If Statement: Executes a sequence of statements if a condition is true.

- IF - THEN - ELSE statement : If condition is True then First block executed else condition false then Second Block is Executed.

- IF - THEN - ELSIF statement : If-then-elsif statement is test multiple condition, if any condition true then that condition Block is executed, if all condition are fail then finally else Block is executed.

- Iterative Control statement : An iterative control statement are used when we want to repeat the execution of one or more statement for specified number of times.
  following are types of LOOP : 1. Loop 2. While Loop 3. for Loop

- GOTO Statement : The GOTO statement immediately transfer program control unconditionally (Without checking Condition) to Label (Jump Location) Statement
  - GOTO statement cannot branch into IF statement, Loop statement.

14) Explain GOTO statement with example.

→ - The GOTO statement immediately transfer program control unconditionally (Without checking Condition) to Label (Jump Location) Statement.

- GOTO statement can not Branch into If statement, Loop statement.

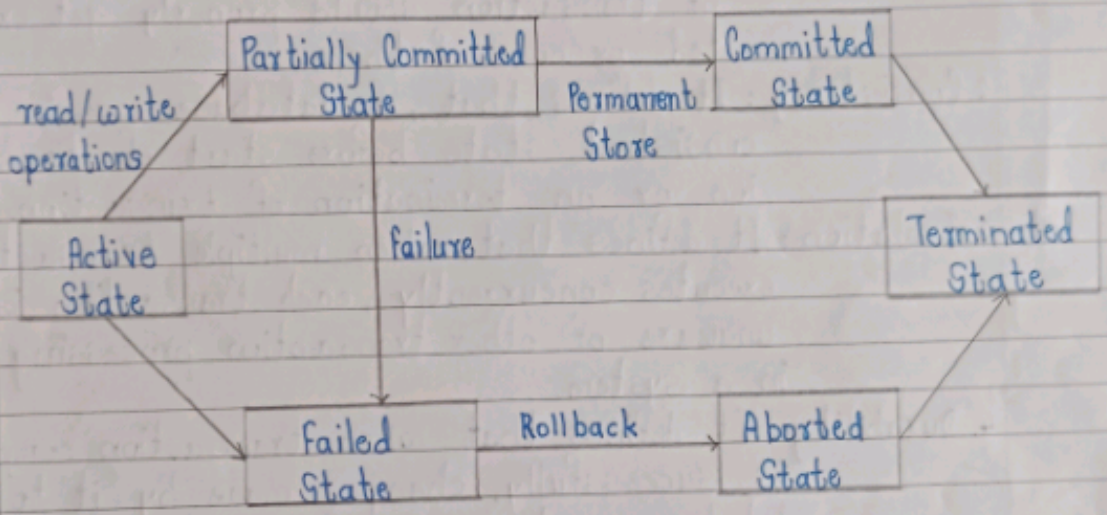
- Syntax:
```
declare
begin
GOTO Label;
<<Label>>
END;
```

- Example:
```
1. declare
2  subcos varchar2(20);
3  s_id number := 1;
4  begin
5  <<come_up>>
6  select sub into subcos from course where sid=s_id;
7  begin
8  dbms_output.put_line(subcos);
9  s_id := s_id + 1;
10 if s_id < 4 then
11 GOTO come_up
12 end if;
13 end;
14 /
```

ASSIGNMENT - 5

1) Explain states of transaction with neat diagram.

→



A transaction is a collection of operation of that performs single logic unit of work.
A transaction goes through various states :

1. Active : The transaction is executing its operative.
2. Partially Committed : All operations have been performed, but changes are not saved.
3. Committed : The transactions changes are saved to the database.
4. Failed : The transaction fails & no changes saved.
5. Aborted : The transaction is terminated due to failure and rolled back.

2) State and explain ACID properties.

→ ACID stands for Automicity, Consistency, Isolation, Durability.

1. Automicity: It defines that either all operation of transaction should properly get executed or none of them.
2. Consistency: It defines that, if database was in a consistent state before start of transaction or on termination of transaction.
3. Isolation: It defines that, when multiple transactions executes concurrently, each transaction is unaware of other transaction processing in a system.
4. Durability: It defines that when transaction completes successfully, changes made by it to the database must be saved inspite of any system failure.

3) What is Database Security? Explain its need.

→ Database Security :- This involves protecting database from unauthorized access, threats and misuse. security measures include access control encryption & auditing.

• Need of Database Security:

1. Data integrity: Prevents unauthorized modifications or curruption of data.
2. Confidentiality: Protects sensitive data from unauthorized access

3. Compliance : Helps meet legal and regulatory standards.
4. Availability : Ensures authorized users can access data when needed.

4) Enlist different database Users. Write function performed by each user.

→ 1. Application programmers
2. Specialized users
3. Sophisticated users
4. Naive users
5. Designers

1. Application programmers : These are computer professionals who interact with system through DML calls.
2. Specialized users : Write specialized database applications that do not fit into the traditional data processing framework.
ex. Expert system, CAD system etc.
3. Sophisticated users : form requests in a database query language.
4. Naive users : these are unsophisticated users invoke one of the permanent application programs that have been written previously. They not have any technical knowledge & provides simple commands provided in user interface.
ex. Data entry operator.
5. Database Designers : responsible for to define the content, structure, constraints, and functions or transactions against the database. Develops DB oriented softwares like MySQL, Server2003 etc.

5) Explain with suitable example how to create user.

→ To create a new user in SQL :

example : create user 'new_user' a 'localhost' identified by password;

This command creates a user named new_user with a specific password & restricts access to localhost. adjust privileges as needed with the GRAND command,

for Example: GRAND select, insert on Database_name to 'new_user' @ 'localhost';

6) Enlist & explain types of database privilege.

→ Database Privilege : A privilege is a right to execute a particular type of SQL statement or to access another user's object.

1. System privileges : A system privilege is the right to perform a particular action (Administrative Action), on a particular type of object.
   ex. the privileges to create tables and to delete.
2. Object privileges : An object privilege is a privilege or right to perform a particular action on a specific table, view, sequence, procedure, function, or package.
   ex. the privilege to delete rows from the table DEPT is an object privilege.

7) What is database backup? Write need of backup.

→ A database backup is the result of copying or archiving files and folders for the purpose of being able to restore them in case of data loss.

- Data loss can be caused by many things ranging from computer viruses to hardware failures to file corruption to fire, flood, or theft etc.
- A backup is a copy of data from your database that can be used to reconstruct that data. Database backup is the process of backing up the operational state, architecture and stored data of database software.

8) Explain types of database backup.

→ 1. Physical backup: Physical backups are copies of physical database files.
ex., a physical backup might copy database content from a local disk drive to another secure location.
A physical backup can be hot or cold.
a) Hot backup: Users can modify the database during a hot backup.
b) Cold backup: Users cannot modify the database during a cold backup, so the database and the backup copy are always synchronized.
2. Logical Backup: A logical backup copies data, but not physical files, from one location to another.
A logical backup is used to move or archieve a database, tables, or schemas and to verify database structures.

9) Differentiate between Physical backup & Logical backup.

→

Physical Backup	Logical Backup
1. Copies actual physical files of the database (data files, log files, etc)	1. extracts data in logical format (SQL statements data dumps).
2. Generally faster, works at the file system level.	2. Slower as it processes data row by row or object by object.
3. Not easily portable across different DBMSs.	3. More portable, can be restore on different systems or versions.
4. Quickers, restores the entire database at once.	4. Slower, especially for large datasets.
5. Requires DBMS specific tools and storage methods	5. Flexible and can be restored using generic SQL tools.
6. Suitable for large databases and full system backups.	6. Suitable for smaller databases or cross-platform restoration.

10) What is database recovery? State its need.

→ Database recovery is the process of restoring data that has been lost, accidentally deleted, corrupted or

made inaccessible
- The System Recovery Options menu contains several tools, such as Startup Repair, that can help you recover Windows from a serious error.

Need for database recovery:
Database recovery refers to the process of restoring a database to a consistent state after a failure or corruption. It involves bringing the database back to a point where data is accurate, transactions are committed, and system integrity is maintained.

11) Differentiate between Roll forward & Rollback recovery technique.

→

Roll forward Recovery	Rollback Recovery
1. Recovers the database by applying changes after the backup.	1. Reverts uncommitted or incomplete transactions.
2. Applies Committed transaction logs to bring the database upto date after a backup.	2. Undoes changes made by failed or incomplete transactions.
3. Uses redo logs to apply committed changes.	3. Uses undo logs to reverse incomplete transactions
4. Minimizes data loss by recovering all committed	4. Ensures data integrity by preventing incomplete

transactions.	transactions from being saved.

12) Write a short note on -
a. Data Warehouse
b. Data lakes.

→ a. Data Warehouse :

A data Warehouse is a centralized repository designed to store large volumes of structured data from multiple sources. It is optimized for query and analysis, often using Online Analytical Processing (OLAP) techniques.

b. Data lakes :

A data lake is a storage respository that holds vast amounts of row data in its native format until it is needed. Unlike a data warehouse, it can store structured, semi-structured, and unstructured data. Data lakes are ideal for big data analytics and support a wid variety of data types.

13) Write a short note on -
a. Data mining
b. Big data
c. Mongo DB
d. DynamoDB

→

a. Data mining:

Data mining is the process of discovering patterns and knowledge from large amounts of data using statistical and computational techniques. It involves methods from machine learning, statistics, and database system.

b. Big data:

Big data refers to extremely large datasets that cannot be managed or analyzed with traditional data processing applications. It encomposses the 3 vs: volume, velocity and variety.

c. Mongo DB:

Mongo DB is a NoSQL database that uses a document-oriented data model. It stores data in JSON-like documents, allowing for flexible schema design and easy scalability.

d. Dynamo DB:

DynamoDB is a fully managed NoSQL database service provided by Amazon web services (AWS). It offers key-value and document data structures, providing high performance and scalability with automatic sharding and replication.

ASSIGNMENT - 6

Part-A

Solve the following Queries

1) Consider following schema: Depositor (Acc_no, Name, PAN, Balance)
Create a view on depositor having attributes (Acc_no, PAN) where balance is greater than 100000.

→
```
SQL> create view Highbalance Depositor AS
     Select Acc_no , PAN
     from Depositor.
     where Balance > 100000 ;
```

2) Consider following schema : Employee(emp_no, emp_name, dept, designation, salary, Dept_location)
Solve following queries :
i. List all managers in Mumbai location.
ii. Set salary of all 'project leaders' to 70000/-.
iii. List employees with having alphabet 'A' as second letter in their name.
iv. Display details of those employees who work in Mumbai or Chennai.

→ i.
```
SQL> select *
     from Employee
     WHERE designation = 'Manager' AND Dept_location = 'Mumbai';
```

(P2)

ii. SQL> UPDATE Employee
SET Salary = 70000
WHERE designation = 'project leaders';

iii. SQL > SELECT *
FROM Employee
WHERE emp_name LIKE 'A-I.';

iv. SQL> SELECT *
FROM Employee
WHERE Dept_location IN ('Mumbai', 'Chennai');

3) Consider following database and solve queries : emp (empno, ename, ph, sal, dept_no, comm)
i. Change employee name 'Rahul' to 'Ramesh'.
ii. Give increment of 20% in salary to all employees.

→ i. SQL > UPDATE emp
SET ename = 'Ramesh'
WHERE ename = 'Rahul';

ii. SQL> UPDATE emp
SET sal = sal * 1.20;

4) Consider following schema : Employee (emp_no, emp_name, dept, designation, salary, Dept_location)
Solve following queries:

→ i. List Same as que. ②

5) Consider following schema: Depositor(cust_name, acc_no)
Borrower(cust_name, loan_no)

i. Find customer name having savings account as well as loan account.
ii. Find customer names having loan account but not the saving account.

→ i. SQL> select D.cust_name
FROM Depositor D
JOIN Borrower B ON D.cust_name = B.cust_name;

ii. SQL> SELECT B.cust_name
FROM Borrower B
LEFT JOIN Depositor D ON B.cust_name = D.cust_name
WHERE D.cust_name IS NULL;

6) Given - Employee (EMP_ID, FIRST_NAME, LAST_NAME, SALARY, JOINING_DATE, DEPARTMENT)
Write SQL queries for-
i. Get FIRST_NAME, LAST_NAME from employee table.
ii. Get unique DEPARTMENT from employee table.
iii. Get FIRST_NAME from employee table using alias name "Employee Name"
iv. Get FIRST_NAME from employee table after removing white spaces from left.

→ i. SQL> SELECT FIRST_NAME, LAST_NAME
FROM Employee;

ii. SQL> SELECT DISTINCT DEPARTMENT
From Employee;

iii. SQL> SELECT FIRST_NAME AS "Employee Name"
FROM Employee;

iv. SQL> SELECT LTRIM (FIRST_NAME) AS "FIRST NAME"
FROM Employee;

7) Consider the structure of student (Name, Mark, Age, Place, Phone, Birthdate).
Write SQL queries for the following:
i. To list name of student who do not have phone number.
ii. To list students from Mumbai and Pune.
iii. To change mark of 'Ajay' to 88 instead of 80.
iv. To list the students whose age is more than 12.

→ i. SQL> SELECT Name
FROM Student
WHERE Phone IS NULL;

ii. SQL> SELECT FROM Student WHERE Place IN ('Mumbai', 'Pune');

iii. SQL> UPDATE Student SET Mark = 88 WHERE name= 'Ajay'
AND Mark=80;

iv. SQL> SELECT *
FROM Student
WHERE Age > 12;

8) Consider the following database:
Employee (emp_id, emp_name, emp_city, emp_addr, emp_dept, join_date)
Solve the following query:
i. Display the names of employees in capital letters.
ii. Display the emp_id of employee who live in city Pune and Mumbai.
iii. Display the details of employees whose joining date is after '01-Apr-1997'.
iv. Display the total number of employees whose dept.no is '10'.

→ i. SQL> SELECT UPPER (emp_name) AS emp_name_upper
from Employee;

ii. SQL> SELECT emp_id FROM Employee
WHERE emp_city IN ('Pune', 'Mumbai');

iii. SQL> SELECT * FROM Employee Where join_date > '1997-04-01';

iv. SQL> SELECT COUNT(*) AS total_employees_dept_10
FROM Employee
WHERE emp_dept = 10;

9) Consider the structure: Book_master {book_id, book_name, subcode, author, no_of_copies, prices}.
Write SQL queries for the following:
i. Display total no. of book for subject 'DBM'.
ii. Get authorwise list of all books.

iii. Display all books whose prices are between Rs. 200 and Rs. 500.

→ i. SQL> SELECT COUNT(*) AS total_books
FROM Book_Master
Where subcode = 'DBM';

ii. SQL> SELECT author, book_name
FROM Book_Master
ORDER BY author;

iii. SQL> SELECT *
FROM Book_Master
WHERE price between 200 AND 500;

10) Consider the following database: Employee (emp_id, emp_name, emp_city, emp_addr, emp_dept, join_date)
Solve the following query:
i. Display the names of employees in capital letters.
ii. Display the emp_id of employee who live in city Pune and Mumbai.
iii. Display the details of employees whose joining date is after '01-Apr-1997'.
iv. Display the total number of employees whose dept. no is '10'.

→ i. SQL> SELECT UPPER (emp_name) AS emp_name upper
FROM Employee;

ii. SQL> Select emp_id from Employee where emp_city IN ('Pune', 'Mumbai');

iii. SQL> SELECT *
FROM Employee
WHERE join_date > '1997-04-01';

iv. SQL> SELECT COUNT(*) AS total_employees_dept_10
FROM Employee
WHERE emp_dept = '10';

ii) Consider the structure : Book_master = { bookid, bookname, subcode - author, no_of copies, price }
Write SQL queries for following:
i. Display total no. of books for subject 'DBM'.
ii. Get authorwise list of all books.
iii. Display all books whose prices are between Rs. 200 and Rs. 500.
iv. Display all books with details whose name start with 'S'.

→ i. SQL> Select COUNT(*) AS total_books
From Book_master
Where subcode = 'DBM';

iii. SQL> Select *
FROM Book_master where price between 200 AND 500;

ii. SQL> Select author, book_name
FROM Book_master ORDER By author;

iv. SQL> Select * from Book_master where book_name
Like 'S%';

Part B

Write following PL/SQL Programs-

1) Write a program to find largest of two numbers.

→
```
declare
num1 number := &num1;
num2 number := &num2;
largest number;
begin
if num1 > num2 then
largest := num1;
else
largest := num2;
end if;
dbms_output.put_line ('The largest number:' || largest);
end;
/
```

2) Write a PL/SQL program to print numbers from 1 to 15 using for loop.

→
```
begin
for i IN 1..15 loop
dbms_output.put_line(i);
end loop;
end;
/
```

3) Write a PL/SQL program to display square of any number.

→
```
declare
num number := &num;
square number;
begin
square := num * num;
dbms.output.put_line ('The square of ' || num || 'is' || square);
end;
/
```

4) Write a PL/SQL program which handles divide by zero exception.

→
```
declare
num1 number := &num1;
num2 number := &num2;
result number;
begin
result := num1 / num2;
dbms_output.put_line ('The result is :' || result);
exception
when zero_divide then
dbms_output.put_line ('Error : Cannot divide by zero');
end;
/
```

5) Write PL/SQL procedure to calculate factorial of a given number.

→
```
declare
```

~~num number~~
create or replace procedure calculate factorial
(num IN number) IS
factorial number := 1;
begin
for i IN 1.. num loop
factorial := factorial * i;
end loop;
dbms_output.put_line('factorial of ' || num || factorial);
end;
/

6) Write PL/SQL program to print even or odd numbers from given range.

→ declare
lower_limit number := &lower_limit;
upper_limit number := &upper_limit;
begin
for i IN lower limit.. upper_limit loop
if mod (i, 2) = 0 then
dbms_output.put_line (i || 'is even');
else
dbms_output.put_line (i || 'is odd');
end if;
end loop;
end;
/

7) Write a PL/SQL program to find the square of a number given by user WHILE... LOOP. (Accept the number from user dynamically).

→
```
declare
num number := &num;
square number;
begin
square := 8;
while square := 0 loop
square := num * num;
dbms_output.put_line ('The square of ' || num || 'is' || square);
end loop;
end;
/
```

8) Write a PL/SQL program using while loop to display n even numbers.

→
```
declare
n number := &n;
counter number := 1;
current_even number := 2;
begin
while counter <= n loop
dbms_output.put_line(current even);
current_even := current even 2;
end loop;
end;
/
```

a) Write a PL/SQL program to print numbers from 50 to 60 using for loop.

→
```
begin
for i IN 50.. 60 loop
dbms_output.put_line(i);
end loop;
end;
/
```

b) Write PL/SQL code to find sum and average of three numbers.

→
```
declare
num1 number := &num1;
num2 number := &num2;
num3 number := &num3;
sum_result number;
average_result number;
begin
sum_result := num1 + num2 + num3;
average_result := sum_result /3;
dbms_output.put_line ('Sum:' ||sum_result);
dbms_output.put_line('Average:' ||average_result);
end;
/
```

11) Write PL/SQL code to find area of circles with radius greater than 3 and less than equal to 7 and store the result in a table with attributes radius and

area.

→ First create a table to store the result:-

```
Create table circle_area(radius int, area int);

declare
radius number;
area number;
begin
for radius IN 4..7 loop
area := 3.14 * radius * radius;
insert into circle_area(radius, area) values(radius,
area);
end loop;
dbms_output.put_line('Area inserted successfully');
end;
/
```

12) Write PL/SQL code to find greatest of three numbers.

→

```
declare
num1 number := &num1;
num2 number := &num2;
num3 number := &num3;
greatest number;
begin
if num1 > num2 AND num1 > num3 then
greatest := num1;
else if num2 > num1 AND num2 > num3 then
greatest := num2;
```

```
else
greatest := num3;
end if;
dbms_output.put_line ('The greatest number is ' ||greatest);
end;
/
```

13) Write PL/SQL program to check weather given number is Prime or not.

→

```
declare
num number := &num;
is_prime Boolean := True;
begin
if num <= 1 then
is_prime := false;
else
for in 2.. floor (sqrt (sum)) loop
if mod (num,i):=0 then
is_prime := false;
exit;
end if;
end loop;
end if;
if is_prime then
dbms_output.put_line(num||'is a prime number');
else
dbms_output.put_line (num||' is not a prime number');
end if;
```

```
end;
/
```

14) Write PL/SQL program to generate Fibonacci series.

→
```
declare
num1 number := 0;
num2 number := 1;
temp number;
begin
dbms_output.put_line(num1);
dbms_output.put_line(num2);
for i IN 3..n loop
temp := num1 + num2;
dbms_output.put_line(temp);
num1 := num2;
num2 := temp;
end loop;
end;
/
```

15) Write PL/SQL program to return Sum of odd number between 1 to 100.

→
```
declare
sum_odd number := 0;
begin
for i IN 1..100 loop
if mod(i, 2) then
sum_odd := sum_odd + i;
end if;
```

```
end loop;
dbms_output.put_line('Sum of odd numbers between
1 to 100 :' || sum_odd);
end;
/
```

16) Write PL/SQL program for displaying top 10 employee details based on salary using cursors.

→

```
declare
cursor emp_cursor IS
select emp_id, emp_name, salary from employee ordered
by salary FETCH first 10 rows only;
emp_record emp_cursor %rowtype;
begin
open emp_cursor;
loop
fetch emp_cursor into emp_record;
exit when emp_cursor %NOT found;
dbms_output.put_line(' Employee ID ' ||emp_record,
emp_id ||emp_record.emp_name||'Salary:' || emp_record.
salary);
end loop;
close emp_cursor;
end;
/
```

17) Write PL/SQL program to handle a predefined exception.

→
```
declare
num1 number := &num1;
num2 number := &num2;
result number;
begin
result := num1 / num2;
dbms_output.put_line('Result' || result);
exception
when zero_divide then
dbms_output.put_line('ERROR: Division by zero is not
                      allowed');
end;
/
```

18) Write PL/SQL program to handle a user defined exception.

→
```
declare
insufficient_funds exception;
balance number:=500;
withdraw_amount number := &withdraw_amount;
begin
if withdraw_amount > balance then
raise insufficient_fund;
else
balance := balance_withdraw_amount;
dbms_output.put_line('withdrawal successful new
  balance:' ||balance);
end if;
exception
```

```
when insufficient_funds then
dbms_output.put_line('error: insufficient funds');
end;
/
```

1a) Write PL/SQL function that computes and returns the maximum of two values.

→
```
create or replace function max of two (num1
number, num2 number)
return number IS
begin
if num1 > num2 then
return num1;
else
return num2;
end if;
end;
/
```

2a) Write PL/SQL Procedure for creating a procedure for calculating sum of two numbers.

→
```
create or replace procedure sum_of_two (num1
IN number) IN number, result out number) IS
begin
result := num1 + num2;
end;          ... example call
declare
result number;
```

```
begin
sum_of_two(10,20,result);
dbms_output.put_line('Sum:' ||result);
end;
/
```