RDBMS (Diploma 3rd Computer Engineering)

Unit : 1

Introduction to Database System and SQL commands

Prepared by:

Mr. Uresh N. Parmar

Lecturer, Computer Engg. Dept.
C U Shah Govt. Polytechnic
Surendranagar

CO (Course Outcome)
&
UOs (Unit Outcomes)

CO (a)

Perform queries on datasets using SQL*Plus

UOs

1a. Differentiate the terms: Data, Information, Records, Fields, Metadata, Data warehouse, Data dictionary.

1b. DBMS Data types, Creating Tables (DDL), Managing Tables (DML) with SQL.

1c. Describe & practice Transaction Control Data Control Language.

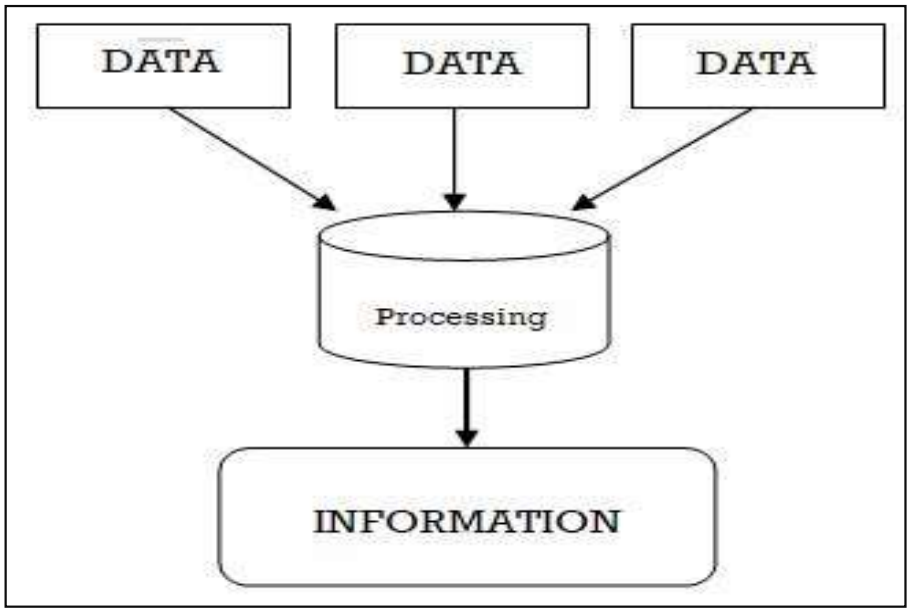
1.1 Basic concepts and definitions

❖ Data

- Plural form of *Datum.*
- Definition.
- Unorganized form.

❖ Information

- Definition.
- It is more precise form of data.



1.1 Basic concepts and definitions

❖ Database and Database system

❑ Database

- Evolved in 1960s.
- Kind of repository.
- Database is '*organized collection of interrelated data that is stored electronically in computer*'.
- It represents aspects of real world.
- Popular types of database.

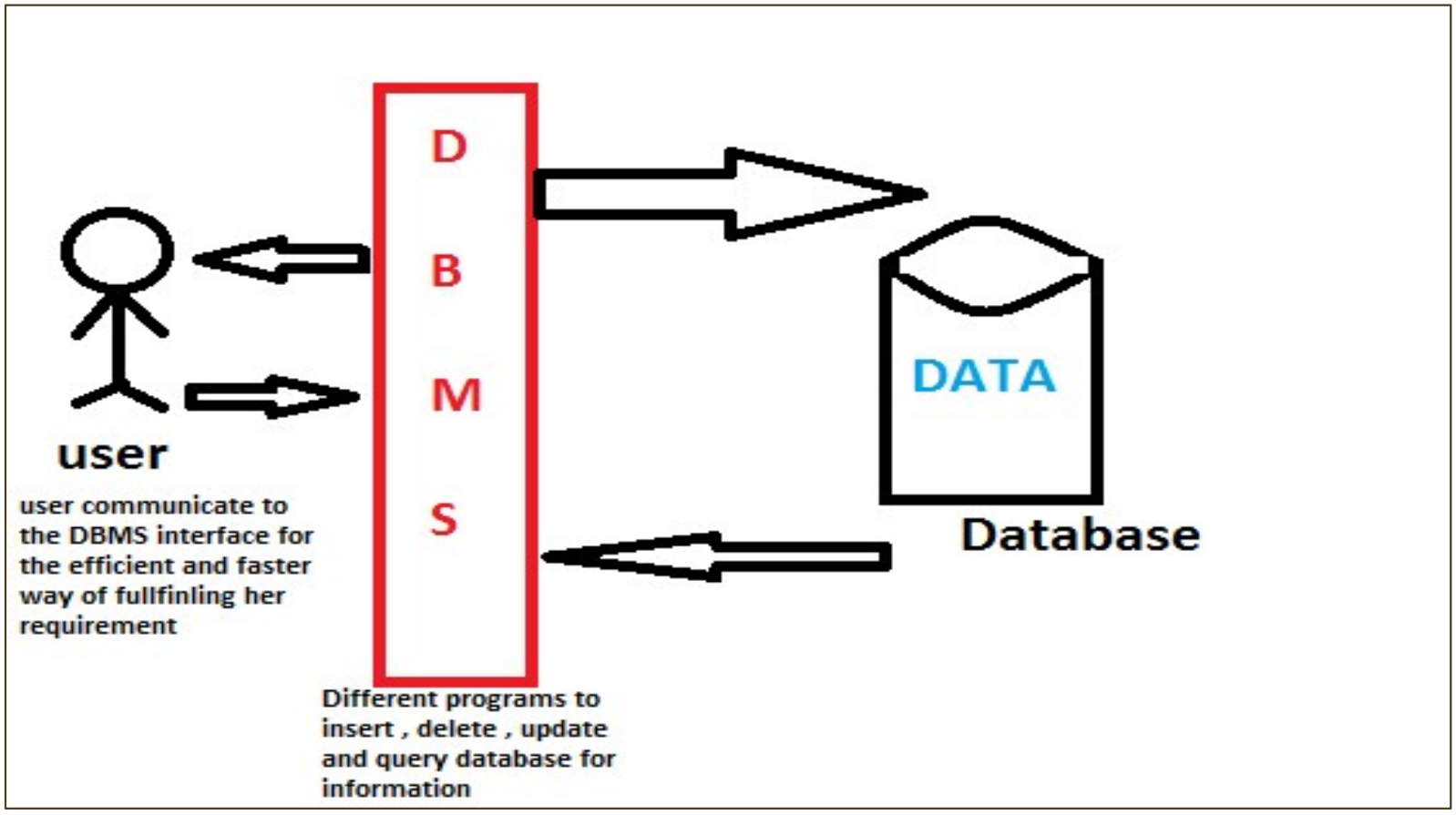
❑ Database System (Database Management System ~ DBMS)

DBMS = Database (DB) + A set of programs (MS)

- *A Database Management System (DBMS) is a software system that is designed to manage and organize data in a structured manner.*

1.1 Basic concepts and definitions

- A **DBMS (Data Base Management System)** is *a collection of interrelated data (database) and set of programs to manipulate those data. Data manipulation involves various operations like storing, retrieving, modifying, removing data.*



1.1 Basic concepts and definitions

➥ Objectives of DBMS

- Data organization	- Data security
- Data availability	- Data sharing
- Data integrity	- Data scalability
- Data independence	

➥ Some examples of DBMS

dBase	**FoxPro**	**MySQL**
Oracle	**SQL Server**	**MS Access**

➥ Operations that can be performed on DBMS

- ✓ Creating tables or files for database.
- ✓ Inserting new data into existing database.
- ✓ Modifying or updating existing database.
- ✓ Removing data from existing database.
- ✓ Deleting or destroying tables or files for database.

1.1 Basic concepts and definitions

➥ Advantages of DBMS

- Reduce redundancy	- Data security
- Data sharing	- Data integrity
- Data recovery and backup	- Makes data management easy

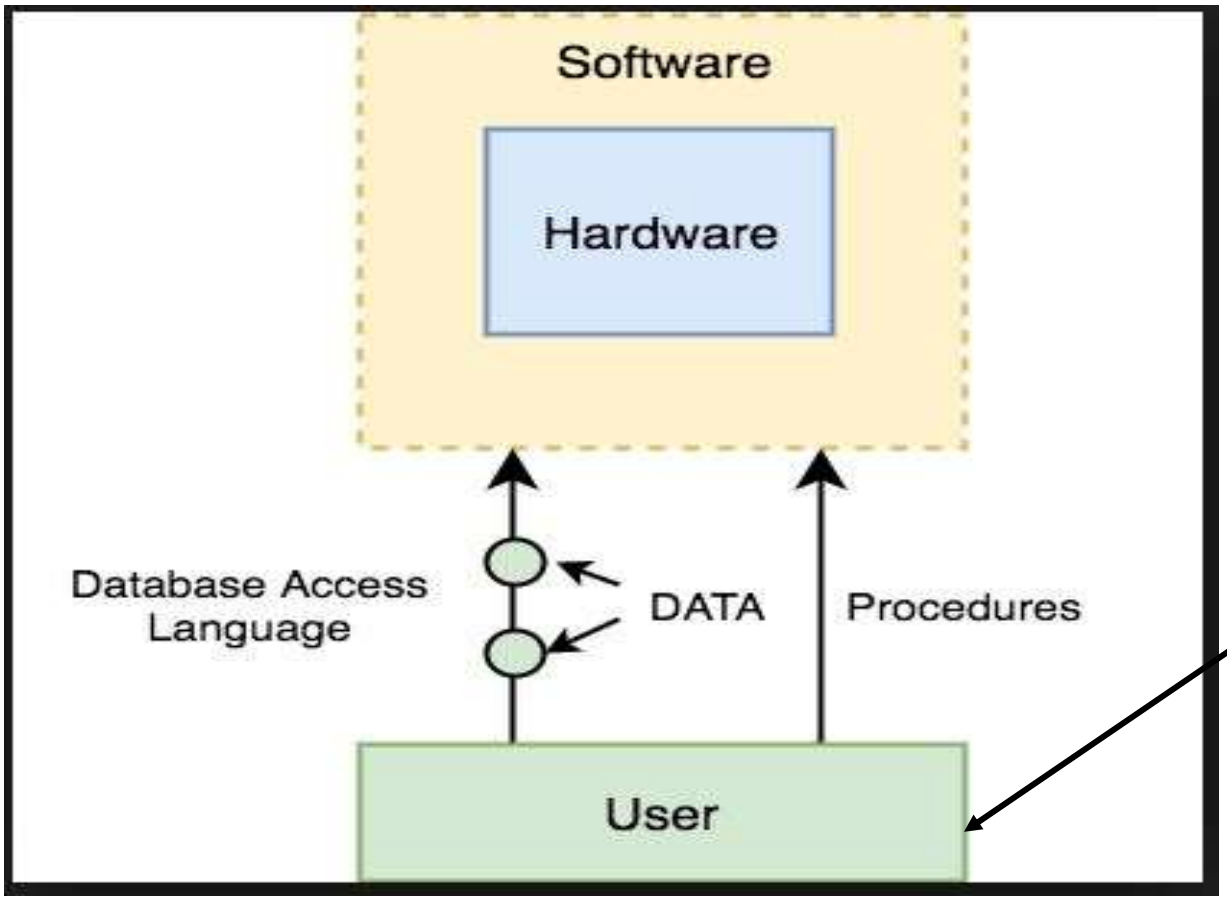
➥ Disadvantages of DBMS

- Installation cost
- Upgradation
- Need technically skilled persons
- Risk when multiple users
- Performance issues in large database

1.1 Basic concepts and definitions

❖ Database system environment

- It is a collective setup of hardware, software, and other resources
- Major components are → data, hardware, software, user

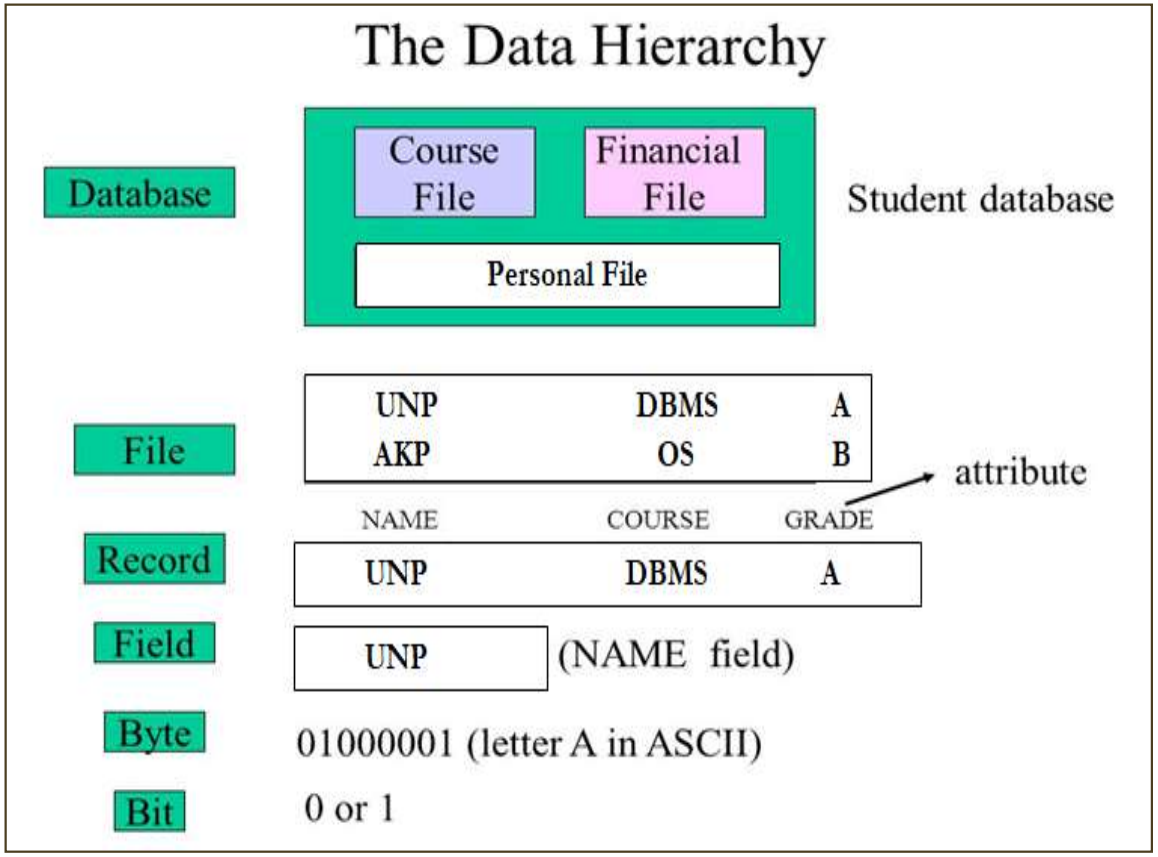


Categories of users:

- Database administrator
- Database designer
- Application programmer
- End user

1.2 Some other basic concepts

❖ Field, Record and File

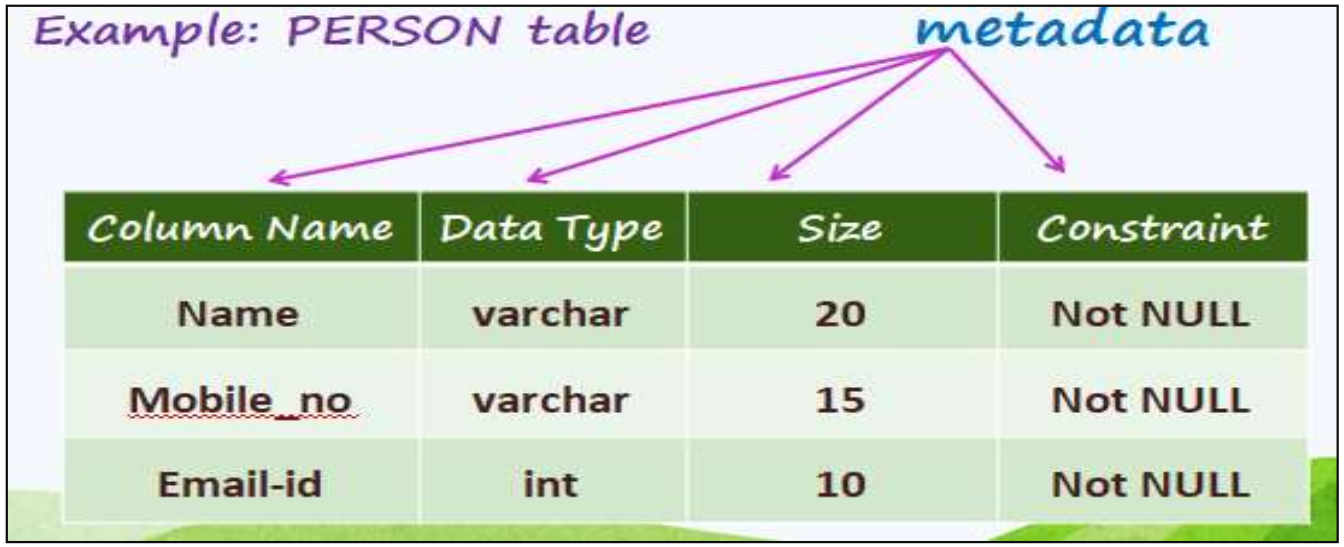


Enrolment No.	Student Name	Email-id
101	ABC	abc@gmail.com
102	PQR	pqr@gmail.com
103	XYZ	xyz@gmail.com

1.2 Some other basic concepts

❖ Metadata

- In a simple term →Data about data
- Also contains rules and constraints about the data.



Column Name	Data Type	Size	Constraint
Name	varchar	20	Not NULL
Mobile_no	varchar	15	Not NULL
Email-id	int	10	Not NULL

- **Advantages of metadata**
  - Get the information about other data
  - Provides roadmap to data warehouse
  - Provides easier access to manipulate the data
  - Provides data independence

1.2 Some other basic concepts

❖ Data dictionary

- It is a file or a set of files that contains a database's metadata.
- It is storing up-to-date information about the tables, indexes, constraints, functions etc.

➥ Data dictionary contains following information:

- Name of tables and schemas.
- Information about constraints.
- Descriptions of users.
- Descriptions of transactions.
- Relationships of tables.
- Also contains physical information of storage.

➥ Components of data dictionary

- **Entity**
  - Real world thing. And Example.

1.2 Some other basic concepts

- **Attribute**
  - Characteristics or properties of an entity.
  - Example
- **Relationship**
  - It is a connection or association.
  - Example.
- **Key**
  - Used to uniquely identify records.
  - Examples.

➥ **Types of data dictionary**

1. Active data dictionary

- Definition.
- Also called integrated data dictionary.
- Always consistent with current updates.

1.2 Some other basic concepts

2. Passive data dictionary

- Definition.
- Also called non-integrated data dictionary.
- It is not consistent with current updates, manual changes required.

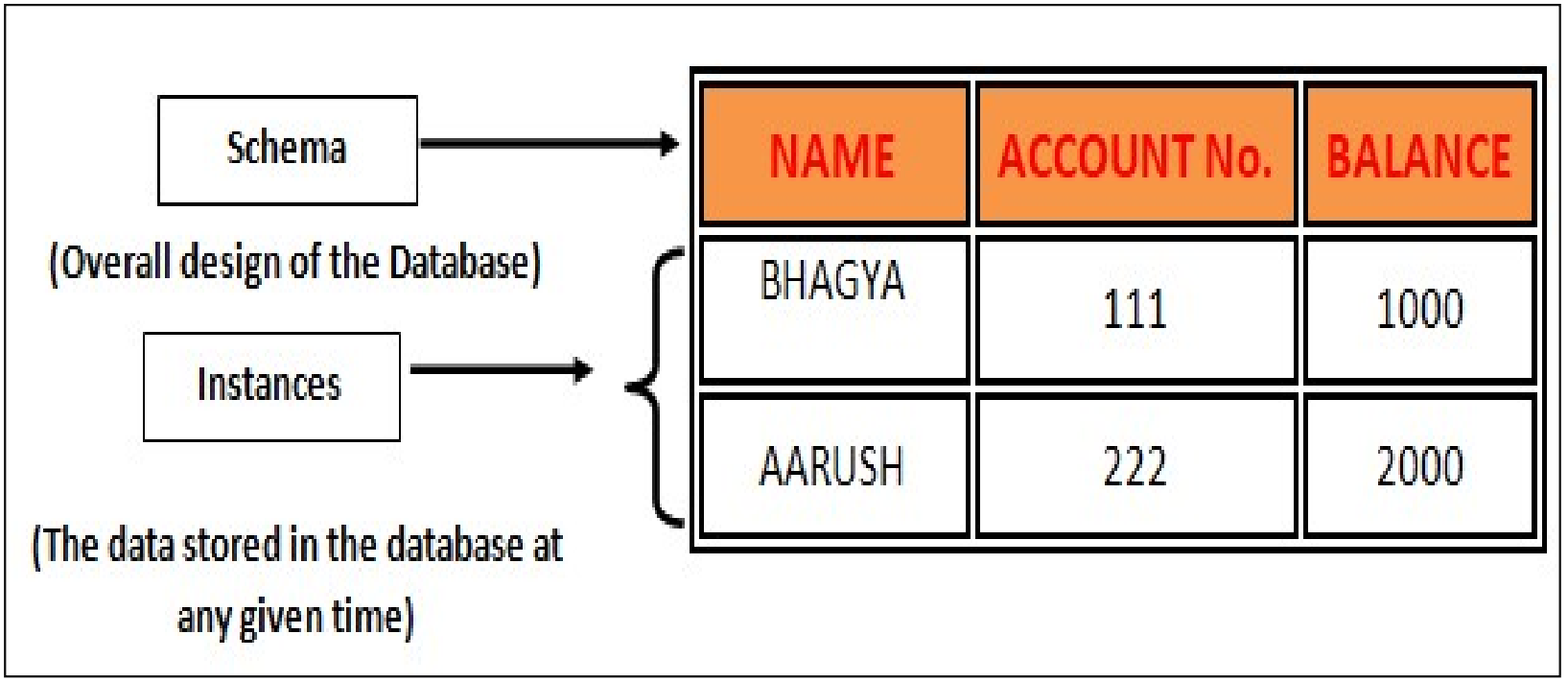
➥ Advantages of data dictionary

- Remove duplication
- Gives well structured information
- Improve communication
- Helpful for DBA

➥ Disadvantages of data dictionary

- Creating new DD is complex task.
- High cost. Hard to understand for non-technical users.
- Care should be taken at the time of creating.

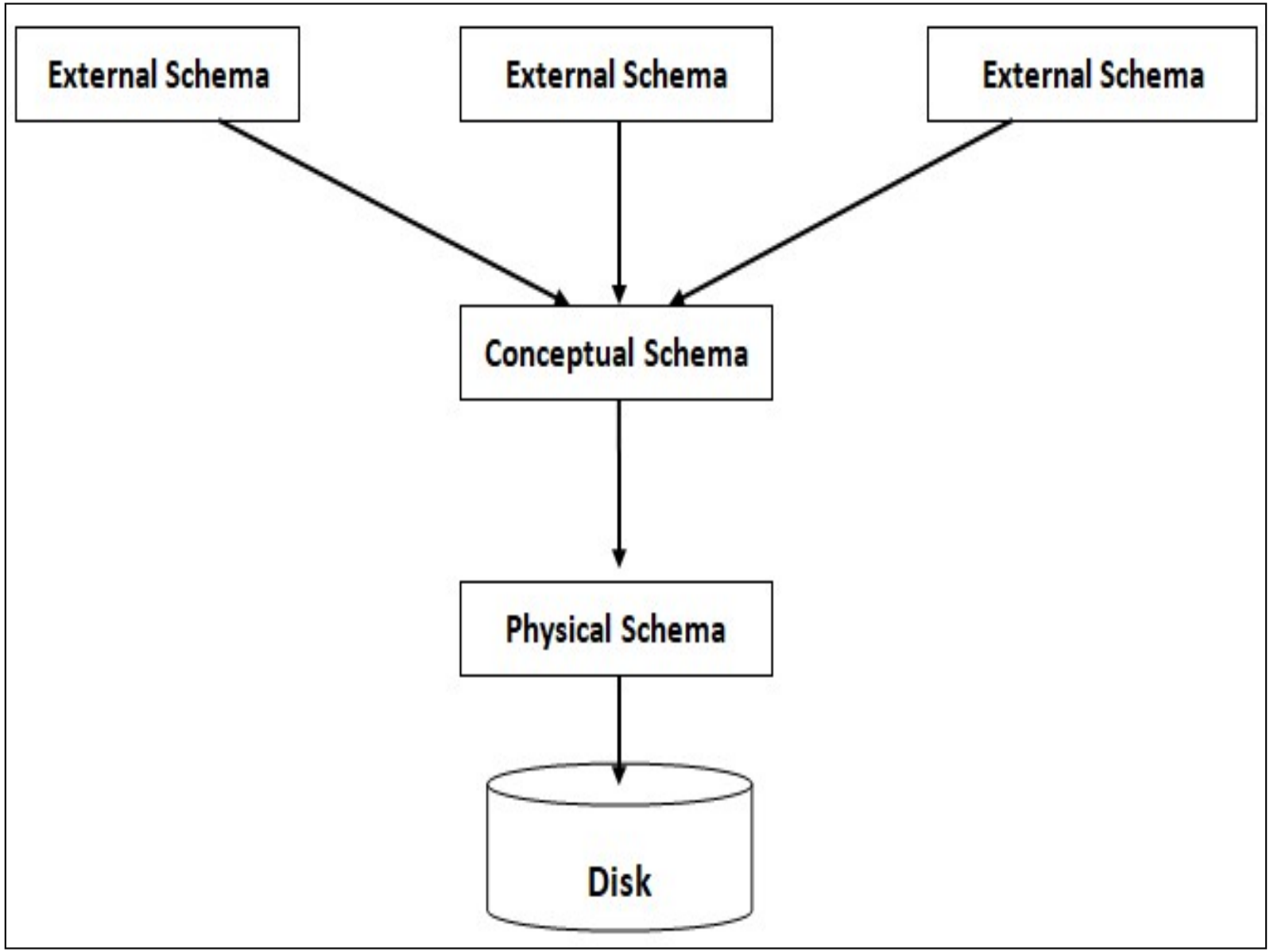
1.3 Schema, Sub-schema and Instances



❖ Schema

- Overall logical design of the database.
- Kind of skeleton.
- Defines organization and relationship of data.
- Does not change frequently.
- According to DBMS architecture, schema types:

1.3 Schema, Sub-schema and Instances



1.3 Schema, Sub-schema and Instances

❖ Sub - schema

- Sub set or sub level of schema.
- It should be a view purpose only. (user's view)

❖ Instance

- Actual data stored in database at particular time.
- Kind of snapshot.
- It changes frequently.
- Example of student table.

1.3 Schema, Sub-schema and Instances

❖ Difference between Schema and Instance

Schema	Instance
✓ It is logical design of database.	✓ It is the information stored in database at particular time.
✓ It defines organization and relationships of data.	✓ It is a kind of snapshot of database.
✓ It is not changed frequently.	✓ It is changed frequently.
✓ It displays table name, column name, data type, data size, constraints etc.	✓ It displays actual data stored in table in the form of records.

1.3 Schema, Sub-schema and Instances

❖ Difference between File system and Database system

File System	Database System
- It coordinates only physical access of data.	- It coordinates both physical and logical access to data.
- It has predetermined access of data (i.e. compiled programs)	- It has flexible access of data (i.e. queries)
- It stores unstructured data.	- It stores structured data.
- No concurrent access for the same time.	- Multiple users can access the same data at same time.
- Low security in it.	- High security in it.
- There should be integrity problems.	- Not such problems due to integrity constraints.
- Data redundancy is there.	- No data redundancy.
- Data is accessed through single or various files.	- Tables (schema) are used to access data.
- Data Inconsistency is more in file system	- Data Inconsistency is less in database system
- It is less complex	- It is more complex
- It is more flexible	- It is less flexible

Introduction to SQL

- Data → Information → Database → DBMS
- Full form of SQL. Developed by IBM (1070s), ANSI standard.
- Non procedural language.
- Command base language.
- Case insensitive.
- Concept of RDBMS with its examples.

❖ Features of SQL

- Open source, Non-procedural, English like language, case in-sensitive language.
- Used by range of users.

❖ What SQL can do?

Introduction to SQL

❖ Some commonly used and most important commands of SQL

- CREATE, DROP, ALTER for tables
- INSERT, UPDATE, DELETE for rows
- GRANT, REVOKE for permission
- SELECT for query and data retrieval.

❖ Some other concepts

- Keyword
- Clause
- Statement
- Command
- Comments

Introduction to SQL

❖ Rules for SQL

- Start with **verb**.
- **Semicolon** at the end of a statement.
- **Comma** to separate parameters.
- Statements can be **split**.
- **Space** to separate keywords.
- Character and date values enclosed in (‘ ‘).

1.4 Data types of SQL

- Concept of data types.
- Different categories of data types.

1. Numerical data type	**2. Character / String data type**
3. Date data type	**4. Binary data type**

➢ Numerical data type

No	Data type	Description
1	number	Floating-point number (size and precision depends on the database we are using)
2	integer / numeric / decimal	Fixed-point number (whole number) with precision of 38 (default size)
2	number (p)	Fixed-point number (whole number) with a scale zero and precision p
3	number (p, s)	Floating-point number P refers to precision S refers to scale
4	float / real	Floating-point numbers with varying precision.

1.4 Data types of SQL

- Different scales and precision:

Actual data value	If defined as	Stored as	Max value that can be stored
123.45	number	123.45	Maximum value supported by number data type
123.45	number (3)	123	999
123.56	number (3)	124	999
123.45	number (5,2)	123.45	999.99
123.45	number (5,1)	123.5	9999.9
123.45	number (5,-1)	120	999994
123.45	number (5,-2)	100	9999949
123.45	number (4,2)	Exceed precision	99.99
123.45	real	123.45	Maximum 63 size supported.
123.45	float	123.45	Maximum 126 size supported.

1.4 Data types of SQL

➢ Character / String data type

- Four different data types available.

No	Data type	Description
1	char(size)	▪ Stores character string of fixed length. ▪ Size represents the number of characters (length) to be stored (default size is 1). ▪ Maximum length size is 255 characters (bytes).
2	varchar(size) varchar2(size)	▪ Stores characters of variable length. ▪ It is more flexible than 'char' type. ▪ No default size is available, so you must specify size. ▪ Maximum length is 2000 characters.
3	Long	▪ Stores large amount of character strings of variable length. ▪ Maximum length is up to 2 GB. ▪ Only one column per table can be defined as a 'long'. ▪ A 'long' column cannot be used with WHERE, ORDER BY or GROUP BY clauses.

1.4 Data types of SQL

➢ Date data type

- Store date and time.
- Standard format is → DD-MON-YY
- SYSDATE function is used to retrieve current date.
- Addition and subtraction is also possible.

➢ Binary data type

- Used to store binary data like images, text files, word files, video files etc.
- Two different data types available.

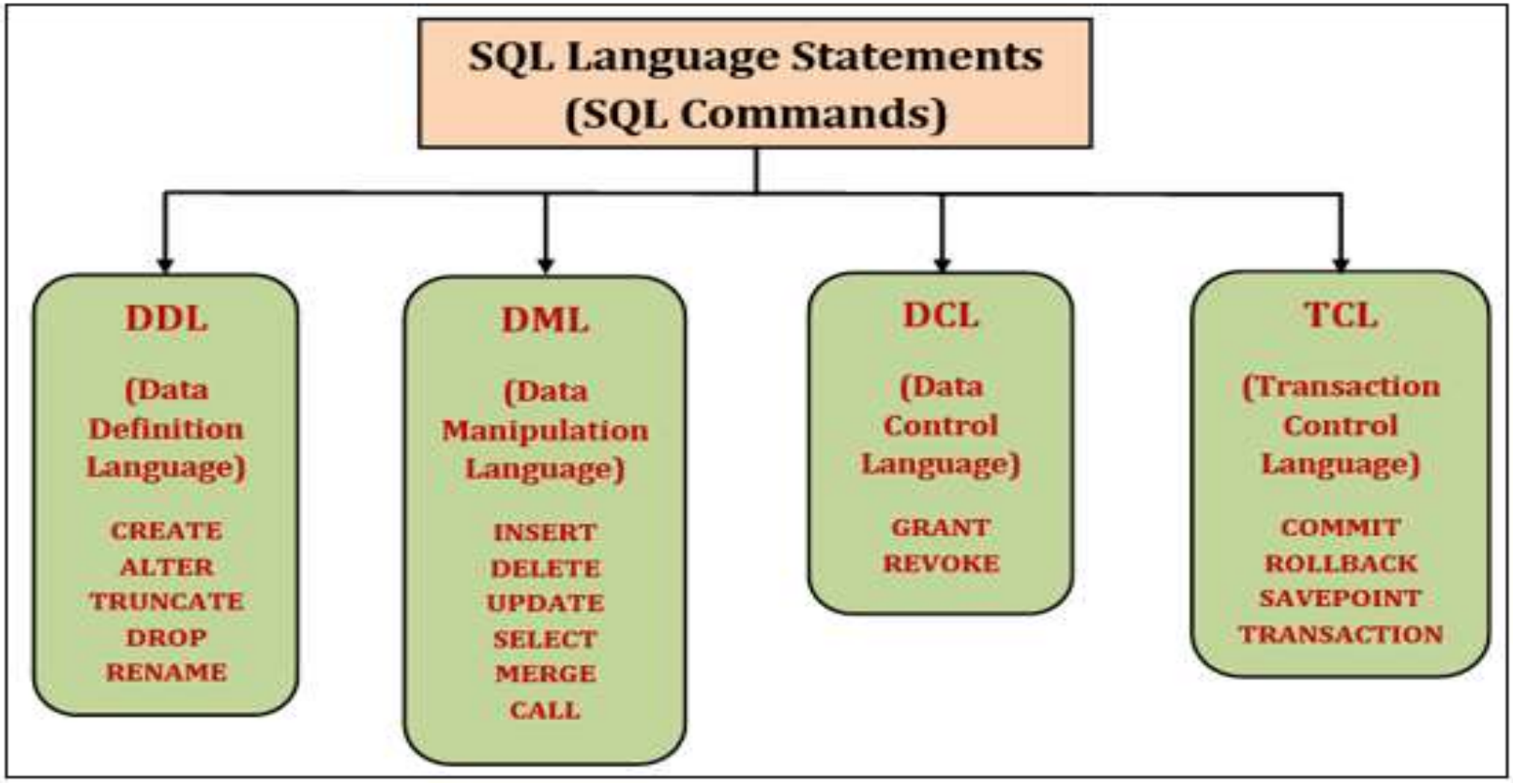
No	Data type	Description
1	RAW	▪ Stores binary type data. ▪ Maximum length is up to 32767 bytes.
2	LONG RAW	▪ Stores large amount of binary type data. ▪ It is often referred as Binary Large Object (BLOB). ▪ Maximum length is up to 2 GB.

Database Languages

- Used to read data, update and store information in DBMS.

Various database languages in DBMS are:

- ✓ **DDL** – Data Definition Language
- ✓ **DML** – Data Manipulation Language
- ✓ **DCL** – Data Control Language
- ✓ **TCL** – Transaction Control Language



1.5 Data Definition Language (DDL)

- Used to define and manage database structure / database objects.
- Deals with database schema, and when DDL statements are executed, data dictionary and system catalogue updated.

Examples of DDL commands:

DDL Commands	Usage
CREATE	It is used to create the database or its objects (like table, index, function, views, store procedure and triggers)
ALTER	It is used to change or modify the structure of database and its objects.
TRUNCATE	It is used to erase all records from the table.
DROP	It is used to delete the database or its objects.
RENAME	It is used to rename the existing objects of database.

1.5 DDL (CREATE)

❖ CREATE (Creating a table or any other database object)

- **Syntax**

```
CREATE TABLE table_name
(column_name1 datatype (size),
column_name2 datatype (size),
--
column_nameN datatype (size) );
```

- Table name must be unique.
- Table name and column name must starts with an alphabet and does not match with any keyword.
- Column definition requires column name, data type and size.
- Column names separated by comma.
- Entire SQL statement terminated by semi colon.
- Example: create a table '*test*' with given schema.

1.5 DDL (CREATE)

Example:

Create a 'test' table having three columns (id, name and age) described below:

test table →

Column name	Data type	Size
id	Number	5
name	Varchar2	15
age	Number	2

Input command →

```
SQL> CREATE TABLE test (id number(5), name varchar2(15), age number(2));
```

Output →

```
Table created.

SQL> DESC test;
 Name                                      Null?    Type
 ----------------------------------------- -------- ----------------------
-
 ID                                                 NUMBER(5)
 NAME                                               VARCHAR2(15)
 AGE                                                NUMBER(2)
```

1.5 DDL (DESCRIBE)

❖ DESCRIBE or DESC (describing a table schema)

- **Syntax**

DESC tablename; OR DESCRIBE tablename;

- Used to check the schema or structure of the table.
- Display information about columns defined in the table.
- It displays column names, NOT NULL constraints and data type with its size of each column of a table.
- **Example (***Show table schema of a table 'test'***)**

Input command →

Output →

```
SQL> DESC test;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
-
 ID                                                 NUMBER(5)
 NAME                                               VARCHAR2(15)
 AGE                                                NUMBER(2)
```

1.5 DDL (ALTER)

❖ ALTER TABLE (altering or modifying table schema)

- Is a DDL command, used to modify the structure of a table.
- Also used to add and drop various constraints on an existing table.
- **Syntax (For adding new column in the table)**

```
ALTER TABLE tablename
ADD newcolumn_name datatype(size), newcolumn_name datatype(size)...;
```

- **Example** (*Add a new column ('gender') to a test table having data type character of size 2)*

Input command →

```
SQL> ALTER TABLE test ADD gender char(2);
```

Output →

```
Table altered.

SQL> DESC test;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
-
 ID                                                 NUMBER(5)
 NAME                                               VARCHAR2(15)
 AGE                                                NUMBER(2)
 GENDER                                             CHAR(2)
```

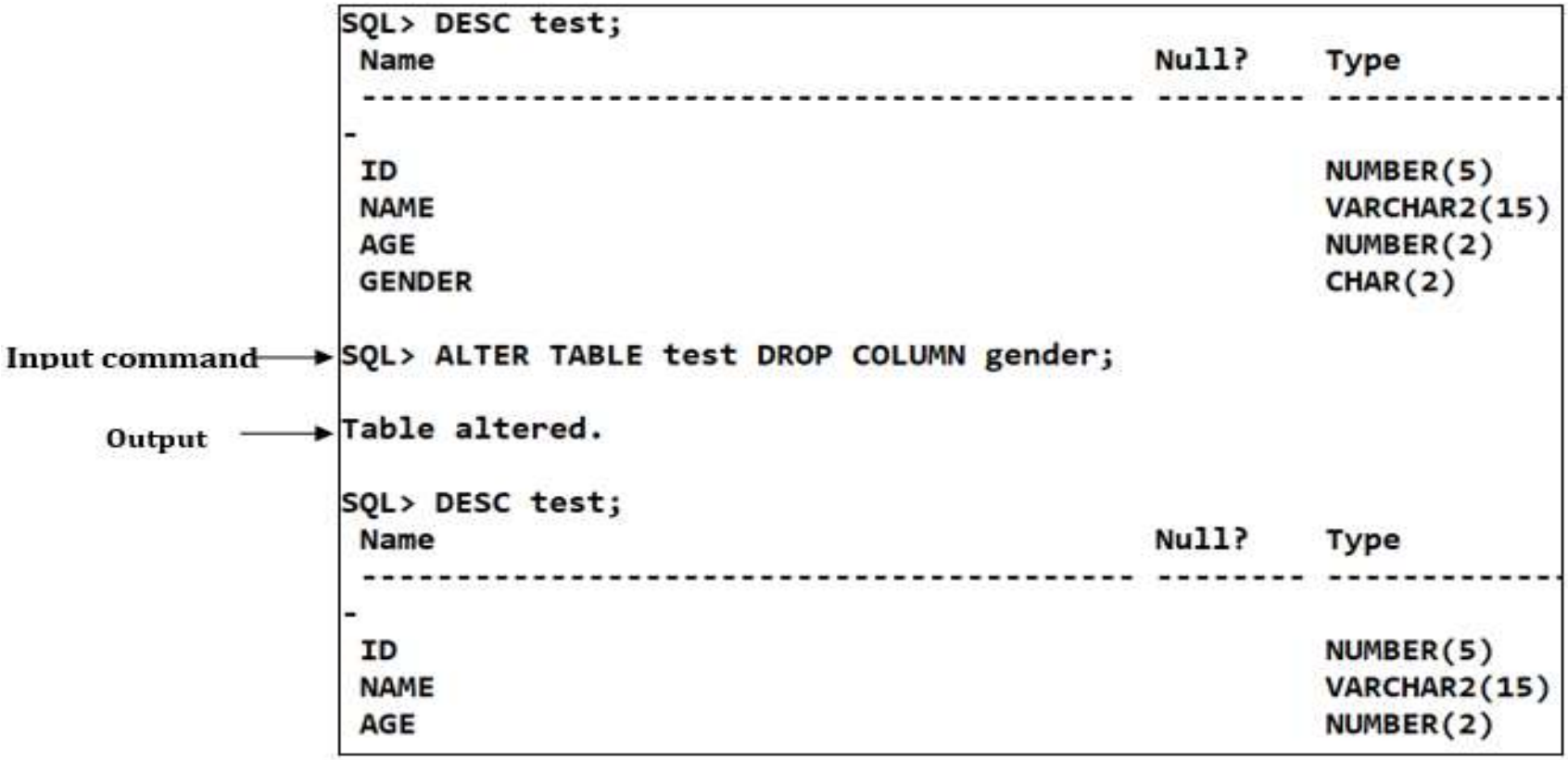
1.5 DDL (ALTER)

- **Syntax (For removing (dropping) an existing column from the table)**

```
ALTER TABLE tablename

DROP COLUMN column_name ;
```

- **Example** (*Remove (or Drop) a column 'gender' from table' test'.)*



1.5 DDL (ALTER)

- **Syntax (For modifying an existing column from the table)**

```
ALTER TABLE tablename

MODIFY column_name NewDatatype (Newsize);
```

- **Example** (*Change the size of the field 'age' from 2 to 4 in the table 'test'*)

```
SQL> DESC test;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
-
 ID                                                 NUMBER(5)
 NAME                                               VARCHAR2(15)
 AGE                                                NUMBER(2)

SQL> ALTER TABLE test MODIFY (age number(4));

Table altered.

SQL> DESC test;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
-
 ID                                                 NUMBER(5)
 NAME                                               VARCHAR2(15)
 AGE                                                NUMBER(4)
```

1.5 DDL (TRUNCATE)

❖ TRUNCATE (deleting all the rows from table)

- It is like a magical eraser.
- It is a DDL command that deletes the data inside a table, but not the table itself.
- Delete all the records unconditionally.
- Deleted records cannot be recovered.
- **Syntax**

TRUNCATE TABLE tablename;

- **Example** (*Delete all the rows from the table test OR Truncate the test table.)*

```
SQL> TRUNCATE TABLE test;

Table truncated.

SQL> select * from test;

no rows selected
```

1.5 DDL (DROP)

❖ DROP (dropping or removing table from database)

- It is used to delete or destroy the table (along with all the records).
- Deleted data cannot be recovered.
- **Syntax**

DROP TABLE tablename;

- **Example** (*Remove the table 'test' from the database)*

```
SQL> DROP TABLE test;

Table dropped.

SQL> DESC test;
ERROR:
ORA-04043: object test does not exist
```

1.5 DDL (RENAME)

❖ RENAME (renaming a table)

- Used when we need to rename a table.
- **Syntax**

RENAME tablename TO NewTableName ;

- **Example** (*Rename the table 'test' and set a new name as 'testnew'*)

```
SQL> RENAME test TO testnew;

Table renamed.

SQL> DESC test;
ERROR:
ORA-04043: object test does not exist


SQL> DESC testnew;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
-
 ID                                                 NUMBER(5)
 NAME                                               VARCHAR2(15)
 AGE                                                NUMBER(4)
```

1.5 DDL (RENAME)

- **Using ‘rename’ and ‘alter’ commands we can change the column name also.**
- **Syntax**

```
ALTER TABLE tablename
RENAME COLUMN old_column_name TO new_column_name ;
```

- **Example** (*Rename the column eno of table ‘test’ to enroll*)

```
SQL> ALTER TABLE test
  2  RENAME COLUMN id TO enroll;

Table altered.

SQL> DESC test;
 Name                                      Null?    Type
 ----------------------------------------- -------- ------------
-
 ENROLL                                             NUMBER(5)
 NAME                                               VARCHAR2(15)
 AGE                                                NUMBER(4)
```

1.6 Data Manipulation Language (DML)

- It supports basic data manipulation on data stored in a table.
- DML affects the records of database.
- It allows users to insert, update, delete and retrieve data from the database and its objects.
- **Examples of DML commands**

DML Commands	Usage
INSERT	It is used to insert data into a table.
DELETE	It is used to delete records from a database table.
UPDATE	It is used to update / modify existing data within a table.
SELECT	It is used to retrieve data from the database.
MERGE	It is used to handle insert, update, delete commands in a single transaction.
CALL	It is used to execute stored procedure or user defined functions.

1.6 DML (INSERT)

❖ INSERT (inserting new data (record) into the table)

- It is a kind of DML command.
- It is used to insert the data into a table or creates a new row in table.
- "INSERT INTO ..." SQL statement is used, and stores the inserted values into respective columns.
- We can write the INSERT INTO statement in ***two different ways***.

➥ **First way (Syntax):**

```
INSERT INTO tablename (column1, column2,...)
VALUES (value1, value2, ...);
```

➥ **Second way (Syntax):**

```
INSERT INTO tablename VALUES (value1, value2, value3...);
```

Explanation of both the ways written above

1.6 DML (INSERT)

- **Example** (*Insert data in the table test*)

```
SQL> INSERT INTO test(id, name, age)
  2  VALUES (1, 'BHAGYA', 15);

1 row created.

SQL> INSERT INTO test VALUES(2, 'AARUSH', 10);

1 row created.

SQL> SELECT * FROM test;

        ID NAME                             AGE
---------- ---------------- ----------
         1 BHAGYA                           15
         2 AARUSH                           10
```

1.6 DML (INSERT)

➥ Inserting NULL values

- We can use different ways by which we can insert NULL.

Example (*Insert a new record in 'test' table that contains NULL value in the 'age' field*)

(Consider the table 'test' where three columns are there id, name and age.

Insert a new record in 'test' table that contains NULL value in the 'age' field.

1st way:

Input is → INSERT INTO test VALUES (3, 'ASMITA', NULL);

2nd way:

Input is → INSERT INTO test VALUES (3, 'ASMITA', '');

3rd way:

Input is → INSERT INTO test (id, name) VALUES (3, 'ASMITA');

1.6 DML (UPDATE)

❖ UPDATE (updating rows of a table)

- It is a kind of DML command.
- It is used to change or modify the data values in a table.
- It updates either all rows or a set of rows from a table.
- The SET clause specifies which column data to modify. Using WHERE, we can update specific record.

Syntax

```
UPDATE tablename
SET column1 = value1, column2 = value2, ....
WHERE condition;
```

1.6 DML (UPDATE)

Example (*Change the age of AARUSH in table 'test' from 10 to 15*)

```
SQL> SELECT * FROM test;

        ID NAME                                AGE
---------- ---------------- ----------
         1 BHAGYA                              15
         2 AARUSH                              10

SQL> UPDATE test SET age=15 WHERE name='AARUSH';

1 row updated.

SQL> SELECT * FROM test;

        ID NAME                                AGE
---------- ---------------- ----------
         1 BHAGYA                              15
         2 AARUSH                              15
```

1.6 DML (DELETE)

❖ DELETE (delete records from a table)

- It is a kind of DML command.
- It can be used to remove either or all the rows of a table, or a set of rows from a table.
- WHERE condition can be used to delete specific records from a table.

Syntax

DELETE FROM tablename WHERE condition;

Example (*Delete the record from the table 'test' whose id is 1*)

```
SQL> SELECT * FROM test;

        ID NAME                           AGE
---------- -------------- ----------
         1 BHAGYA                         15
         2 AARUSH                         15
```

```
SQL> DELETE FROM test WHERE id = 1;

1 row deleted.

SQL> SELECT * FROM test;

        ID NAME                           AGE
---------- -------------- ----------
         2 AARUSH                         15
```

1.6 DML (SELECT)

❖ SELECT (displaying records from one or more tables)

- It is a kind of DML command.
- It is most commonly used SQL statement.
- Mainly used to retrieve data from one or more tables.

Basic Syntax

SELECT column1, column2,.., columnN FROM tablename;

SELECT * FROM tablename;

Example (*Display all the records from the table 'test'*)

```
SQL> SELECT * FROM test;

        ID NAME                          AGE
---------- --------------- ----------
         1 bhagya                        15
         2 aarush                        10
         3 shreeja                        7
         4 kahaan                         3
```

1.6 DML (SELECT)

➥ **Different ways of using SELECT**

1. *Selected columns, All rows*
2. *Selected rows, All columns*
3. *Selected columns, Selected rows*

1. Selected columns, All rows

- This statement retrieves only selected columns and all the rows from a table.

Syntax

SELECT column1, column2,.., columnN FROM tablename;

Example (*Display only name and age of all the records from the table 'test'*)

```
SQL> SELECT name, age FROM test;

NAME                          AGE
--------------- ----------
bhagya                         15
aarush                         10
shreeja                         7
kahaan                          3
```

1.6 DML (SELECT)

2. Selected rows, All columns

- This statement retrieves only selected rows and all the columns from a table. To get the selected rows, WHERE condition is used.

Syntax

SELECT * FROM tablename WHERE condition;

Example (*Display only those records that have age greater than 10*)

```
SQL> SELECT * FROM test WHERE age > 10;

        ID NAME                             AGE
---------- --------------- ----------
         1 bhagya                           15
```

1.6 DML (SELECT)

3. Selected rows, selected columns

- This statement retrieves only selected rows and selected columns from a table. To get the selected rows, WHERE condition is used and to get the selected column, name of the columns applied with SELECT.

Syntax

SELECT column1, column2,.., columnN FROM tablename WHERE condition;

Example (*Display only name whose age greater than 10*)

```
SQL> SELECT name FROM test WHERE age > 10;

NAME
---------------
bhagya
```

1.6 DISTINCT

❖ DISTINCT (eliminating duplicate values)

- It is used to return only distinct (different/unique) values. Duplicates are eliminated.
- It is used with SELECT command.

Syntax

SELECT DISTICT columnName FROM tablename ;

Example

Display distinct (unique) id from table 'test'.

```
SQL> SELECT * FROM test;

        ID NAME                          AGE
---------- -------------- ----------
         1 bhagya                        15
         2 aarush                        10
         3 shreeja                        7
         4 kahaan                         3
         2 aarav                          5
         4 avani                         10

6 rows selected.

SQL> SELECT DISTINCT id FROM test;

        ID
----------
         1
         2
         4
         3
```

Display distinct (unique) age from table 'test'.

```
SQL> SELECT * FROM test;

        ID NAME                          AGE
---------- -------------- ----------
         1 bhagya                        15
         2 aarush                        10
         3 shreeja                        7
         4 kahaan                         3
         2 aarav                          5
         4 avani                         10

6 rows selected.

SQL> SELECT DISTINCT age FROM test;

       AGE
----------
         5
         7
         3
        15
        10
```

Coping data from one table to another (create a table same as another table by copying data)

Syntax

```
CREATE TABLE newTableName (column1, column2,..,, columnN)
AS SELECT column1, column2,...,columnN
FROM SourceTableName
WHERE condition;
```

In above syntax, where condition is optional. we want a new table with all same data like source table, we can use meta character (*), and for that syntax would be:

```
CRAETE TABLE newTableName
AS SELECT * FROM SourceTable;
```

Coping data from one table to another (create a table same as another table by copying data)

Example

```
SQL> CREATE TABLE test1
  2  AS SELECT * FROM test;

Table created.

SQL> SELECT * FROM test1;

        ID NAME                 AGE
---------- -------------- ----------
         1 bhagya                15
         2 aarush                10
         3 shreeja                7
         4 kahaan                 3
         2 aarav                  5
         4 avani                 10

6 rows selected.
```

```
SQL> CREATE TABLE test2 (id, name)
  2  AS SELECT id, name FROM test;

Table created.

SQL> SELECT * FROM test2;

        ID NAME
---------- --------------
         1 bhagya
         2 aarush
         3 shreeja
         4 kahaan
         2 aarav
         4 avani

6 rows selected.
```

*(**Note:** if we want a new table with specific data from source table, then we can use WHERE condition)*

Rename column in display result

- It is used in some cases where we want the column name changed in output of result display. For example, in a table the column name is ‘id’ and we want ‘ENROLLMENT NO’ in output display.
- For that, we can specify the name (which we want in the output) just after the column name with ‘select’ clause

Syntax

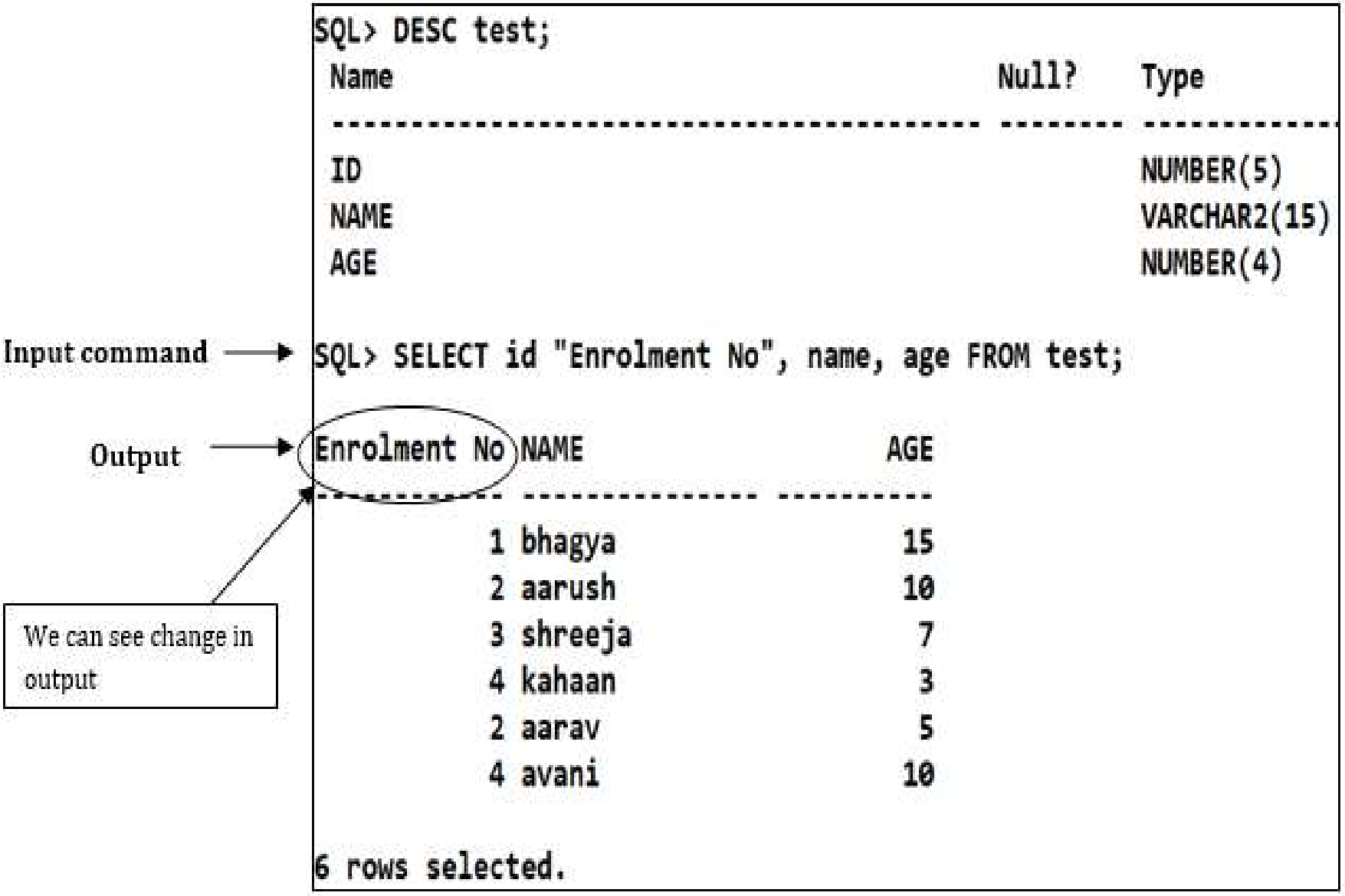
SELECT ColumnName “NewColumnName in output” FROM tablename;

OR

SELECT ColumnName as “NewColumnName in output” FROM tablename;

Rename column in display result

Example



1.7 Transaction Control Language (TCL)

- A transaction is a set of database operations that performs a particular task.
- Transaction in a database must be executed as a whole to maintain database consistency.
- TCL commands are used to manage and control the database transactions.
- **Examples of TCL commands**

TCL Commands	Usage
COMMIT	▪ It is used to save the work permanently.
ROLL BACK	▪ It is working like 'undo' button. ▪ It reverts the changes made during current transactions. ▪ It restores the database to original state since the last COMMIT.
SAVE POINT	▪ It identifies a point in a transaction to which you can later roll back.

1.7 TCL (COMMIT)

- It is a kind of TCL command.
- Used to save all the transactions to the database.
- Once committed, these changes are stored in the database and cannot be undone.
- A transaction can be committed either → ***Explicit*** or ***Implicit***

➥ **Explicit COMMIT**

- It terminates the current transaction and make the changes permanent.

Syntax

```
COMMIT;
```

Example (*self explanation*)

1.7 TCL (COMMIT)

➥ **Implicit COMMIT**

- This COMMIT occurs automatically by Oracle, even we don't specify COMMIT command.
- Implicit COMMIT executed in three following situations:

1. *QUIT command*
   - ✓ To end SQL*PLUS session disconnecting from the oracle.
2. *EXIT command*
   - ✓ To end SQL*PLUS session disconnecting from the oracle.
3. *Data Definition Language (DDL) commands*
   - ✓ Commands like CREATE, ALTER, DROP etc when executed, they save the changes automatically during transaction.

1.7 TCL (ROLLBACK)

❖ ROLLBACK (Cancelling a transaction completely)

- It is a kind of TCL command.
- It is used to undo the transactions that have not already been saved to the database.

Syntax

```
ROLLBACK;
```

Example

(self explanatory using following figure in the next slide)

1.7 TCL (ROLLBACK)

```
SQL> SELECT * FROM test;

        NO NAME                      AGE
---------- ---------- ----------
         1 aarush                     5
         2 bhagya                    10
         3 vihaan                     4
         4 virat                     18

SQL> COMMIT;

Commit complete.

SQL> DELETE FROM test WHERE no=4;

1 row deleted.

SQL> SELECT * FROM test;

        NO NAME                      AGE
---------- ---------- ----------
         1 aarush                     5
         2 bhagya                    10
         3 vihaan                     4

SQL> ROLLBACK;

Rollback complete.

SQL> SELECT * FROM test;

        NO NAME                      AGE
---------- ---------- ----------
         1 aarush                     5
         2 bhagya                    10
         3 vihaan                     4
         4 virat                     18
```

1.7 TCL
(Difference between COMMIT and ROLLBACK)

COMMIT	ROLLBACK
▪ COMMIT permanently saves the changes made by the current transaction.	▪ ROLLBACK undo the changes made by the current transaction.
▪ The transaction cannot UNDO after COMMIT execution.	▪ The transaction reaches to its previous state after ROLLBACK.
▪ Syntax: COMMIT;	▪ Syntax: ROLLBACK;
▪ COMMIT is automatically applied in case of quit, exit or DDL commands.	▪ ROLLBACK is automatically applied in case of transaction aborted, system failure or power failure.
▪ When a transaction is successful, COMMIT is applied.	▪ When the transaction is aborted, incorrect execution, system failure, ROLLBACK occurs.

1.7 TCL (SAVEPOINT)

- It is a kind of TCL command.
- A SAVEPOINT is a point in a transaction where we can rollback the transaction to a certain point without rolling back the entire transaction.
- We can create multiple save points in a transaction where we can roll back.
- Mainly used to cancel the transaction partially.
- SAVEPOINT command is used to create save points.

Syntax

```
SAVEPOINT savepoint_name;
```

1.7 TCL (SAVEPOINT)

Example

Suppose we have a table ‘test’ like:

NO	NAME	AGE
1	aarush	5
2	bhagya	10
3	vihaan	4
4	virat	35
5	vijay	39
6	sachin	25
7	avani	15
8	mahima	23
9	shreeja	2
10	aarav	5

Now in the next slide, we will perform deletion operation two times and create two different save points.

1.7 TCL (SAVEPOINT)

```
SQL> DELETE FROM test WHERE no > 5;

5 rows deleted.

SQL> SELECT * FROM test;

        NO NAME                 AGE
---------- ---------- ----------
         1 aarush                5
         2 bhagya               10
         3 vihaan                4
         4 virat                35
         5 vijay                39

SQL> SAVEPOINT S1;

Savepoint created.

SQL> DELETE FROM test WHERE no > 3;

2 rows deleted.

SQL> SAVEPOINT S2;

Savepoint created.
```

In above transaction, at savepoint S1, we have the table with 5 records and savepoint S2, we have table with 3 records. Now using ROLLBACK command, we can revert back the transaction to any of the savepoints.

1.7 TCL (SAVEPOINT)

- **Syntax to reach at particular savepoint using rollback is:**

```
ROLLBACK TO savepoint_name;
```

```
SQL> ROLLBACK TO S1;

Rollback complete.

SQL> SELECT * FROM test;

        NO NAME                 AGE
---------- ---------- ----------
         1 aarush                5
         2 bhagya               10
         3 vihaan                4
         4 virat                35
         5 vijay                39
```

We can see that using ROLLBACK, we came go back at savepoint S1 and get the data back.

1.8 Data Control Language (DCL)

- First discuss the concept '*how to create user*'.
- By logging in to admin user and then using CREATE USER command, we can create a new user.

Syntax

CREATE USER *user_Name*

IDENTIFIED BY *password*;

Example

Here after login into SYSTEM user, we are creating a new user UNP with password uresh and then we will login into UNP user.

One thing must be note down that after creating a new user, that user must have the permission to create session (OR to get logged in). And that permission given only by the user who creates a new user.

1.8 Data Control Language (DCL)

```
SQL> connect system
Enter password:
Connected.
SQL> CREATE USER UNP
  2  IDENTIFIED BY uresh;

User created.

SQL> connect UNP
Enter password:
ERROR:
ORA-01045: user UNP lacks CREATE SESSION privilege; logon denied
```

- Due to create session permission, the new user is not able to login. That permission can be granted by the user who have created that user.

```
SQL> GRANT CREATE SESSION TO UNP;

Grant succeeded.

SQL> connect unp
Enter password:
Connected.
SQL>
```

Now UNP user is able to login after having CREATE SESSION permission from SYSTEM user who has created him

1.8 Data Control Language (DCL)

❖ DCL Commands

- DCL includes commands such as GRANT and REVOKE which mainly deals with the rights, permissions and other controls to the users of the database system.

Examples of DCL commands

DCL Commands	Usage
GRANT	▪ It gives user's access privileges to database.
REVOKE	▪ It withdraws user's access privileges given by using the GRANT command. ▪ Take back the authority from the users.

1.8 DCL (GRANT)

❖ GRANT (Granting privileges)

- It is a kind of DCL command.
- It is specifically used to provide permission to database objects for a user.
- This command also allows the users to grant permissions to another users which permission they got from the admin.

Syntax

```
GRANT permission_name
ON object_name
TO user_name
[WITH GRANT OPTION];
```

1.8 DCL (GRANT)

➥ **List of permissions (privileges):**

Privilege	Allow user
ALTER	To change the table structure using ALTER command
DELETE	To delete records from the table using DELETE command
INDEX	To create an index on the table using CREATE INDEX command
INSERT	To insert record into the table using INSERT INTO command
REFERENCES	To reference table when using foreign keys
SELECT	To query the table using SELECT command
UPDATE	To modify the record in the table using UPDATE command
ALL	To perform all the operations listed below

1.8 DCL (GRANT)

Example

Suppose we have two users → system and UNP. We have a table 'test' which is created by the user SYSTEM (so we can say test table is owned by the user whose name is SYSTEM). Here by login in 'SYSTEM' user, we will grant SELECT permission to UNP. Before granting the permission UNP user cannot access 'test' table, which we can see in below figure.

```
SQL> CONNECT UNP
Enter password:
Connected.
SQL> SELECT * FROM system.test;
SELECT * FROM system.test
                     *
ERROR at line 1:
ORA-00942: table or view does not exist
```

Now we are giving SELECT permission to the user UNP (to give permission, SYSTEM user must be logged in).

```
SQL> GRANT SELECT
  2  ON test
  3  TO UNP;

Grant succeeded.
```

1.8 DCL (GRANT)

Now, the user UNP is able to view the table ‘test’ of SYSTEM user.

```
SQL> CONNECT UNP
Enter password:
Connected.
SQL> SELECT * FROM system.test;

        NO NAME                          AGE
---------- ---------- ----------
         1 aarush                         5
         2 bhagya                        10
         3 vihaan                         4
         4 virat                         35
         5 vijay                         39
         6 sachin                        25
         7 avani                         15
         8 mahima                        23
         9 shreeja                        2
        10 aarav                          6

10 rows selected.
```

To access another user’s table, dot operator is user along with user name.

1.8 DCL (GRANT)

➥ **WITH GRANT OPTION**

- Using that option, the user to whom we are giving permission can also give permission to other users.
- For example, if we have three users → system, UNP and KGP. Now the system user who is having a table test and he is granting the permission to the user UNP using WITH GRANT OPTION. So, after then UNP can also grant permission of test table (which is owned by system) to another user KGP.

Example

```
SQL> GRANT SELECT
  2  ON test
  3  TO UNP;

Grant succeeded.
```

Here, we have logged in from system user and grant the permission of SELECT for test table to UNP user. But we have not given WITH GRANT OPTION. So the user UNP cannot further grant the privileges of test table. That we can see in below figure.

1.8 DCL (GRANT)

```
SQL> GRANT SELECT
  2  ON system.test
  3  TO KGP;
ON system.test
           *
ERROR at line 2:
ORA-01031: insufficient privileges
```

Now, if we provide WITH GRANT OPTION to UNP, then he further can grant privileges to another user. Check in below figure.

```
SQL> GRANT SELECT
  2  ON test
  3  TO UNP
  4  WITH GRANT OPTION;

Grant succeeded.
```

[system user logged in]

```
SQL> GRANT SELECT
  2  ON system.test
  3  TO KGP;

Grant succeeded.
```

[UNP user logged in]

Now, by logging in to user KGP, we can use SELECT command on *test* table which is owned by the system user.

1.8 DCL (REVOKE)

❖ REVOKE (Revoking or taking back the privileges)

- It is used to remove the permissions or privileges of a user on database objects set by the Grant command.

Syntax

```
REVOKE permission_name
ON object_name
FROM user_name;
```

Example

```
SQL> REVOKE SELECT
  2  ON test
  3  FROM UNP;

Revoke succeeded.
```

[Logged in system user]

1.8 DCL (REVOKE)

- Here in the figure, we have revoked the SELECT permission from UNP user. Now UNP cannot use SELECT command on test table. That we can see below.

```
SQL> connect UNP
Enter password:
Connected.
SQL> SELECT * FROM system.test;
SELECT * FROM system.test
                     *

ERROR at line 1:
ORA-00942: table or view does not exist
```

1.8 DCL (GRANT & REVOKE)

❖ Differences between GRANT and REVOKE

GRANT	REVOKE
– Grant command is specifically used to provide permission to database objects for a user.	– Revoke command is used to remove the permissions or privileges of a user on database objects
– Syntax: GRANT permission_name ON object_name TO user_name [WITH GRANT OPTION];	– Syntax: REVOKE permission_name ON object_name FROM user_name;
– Grant gives access rights to the users.	– Revoke removes the access rights of all users
– When the access is decentralized granting permissions will be easy.	– If decentralized access removing the granted permissions is difficult.

Assignment (MCQs)

1 The smallest piece of information that can be stored in a table is called _____.

a) Record
b) List
c) Field
d) Database

2 Organized collection of interrelated data that is stored electronically in computer is called _____.

a) DBMS
b) Database
c) Data warehouse
d) Information

3 Data about data is _____.

a) Metadata
b) Information
c) Data dictionary
d) Data warehouse

4 Which data dictionary is managed automatically by DBMS?

a) Passive
b) Self dictionary
c) Active
d) DBMS directory

5 The full form of SQL is _____.

a) Structured Queue Language
b) Structured Query Language
c) Systematic Queue Language
d) Systematic Query Language

6 The overall logical design of a database is called _____.

a) Database
b) Entity
c) Attribute
d) Schema

Assignment (MCQs)

7 _____ DDL command is used to destroy the table schema from the database.

a) DROP b) DELETE

c) TRUNCATE d) DESTROY

8 UPDATE is a kind of _____ command.

a) DDL b) DML

c) TCL d) DCL

9 Which TCL command is working like 'undo"?

a) Control Z b) COMMIT

c) UNDO d) ROLLBACK

10 In SQL, which keyword is used to change existing data in a table?

a) CHANGE b) MODIFY

c) UPDATE d) ALTER

11 What does the SQL statement "SELECT * FROM table_name;" do?

a) Selects all columns from the table

b) Selects the first column from the table

c) Selects the first row from the table

d) Selects the number of records in the table

12 Which SQL keyword is used to retrieve a specific subset of records from a table?

a) FILTER b) SEARCH

c) FIND d) WHERE

13 Which of the following is not a valid SQL command?

a) SELECT b) UPDATE

c) PRINT d) MERGE

Assignment (TRUE / FALSE)

True / False	
1	Passive data dictionary is also called integrated data dictionary.
2	CREATE is a DML command.
3	ALTER is a kind of DDL command.
4	We cannot recover the data deleted using TRUNCATE.
5	The SELECT statement in SQL is used to insert data into a database table.
6	The ALTER TABLE statement is used to add, modify, or delete columns in an existing table
7	In DBMS, DDL stands for Directory Definition Language.
8	DBMS improves data redundancy.
9	Database schema refers to the logical layout of the database.
10	In Database NULL and Zero (0) both are same.

Assignment (Short Questions)

Sr. No	Questions	Bloom's Taxonomy Level
1	Define: data, information, database, field, record, file, metadata. *[W22] [S24]*	R
2	Differentiate between *[S24]* i. Data vs Information ii. Char vs Varchar	R
3	Define DBMS. Write applications of DBMS. *[S23]*	R
4	Identify differences between active and passive data dictionary.	A
5	Define: schema, sub-schema and instance. *[W22] [S23] [S24]*	R
6	Give the full form: SQL, DDL, DML, TCL, DCL.	R
7	List different commands used in DDL, DML and DCL.	R
8	Identify differences between DROP and DELETE. *[S23]*	A

Assignment (Descriptive Questions)

Sr. No	Questions	Bloom's Taxonomy Level
1	Write a short note on database system environment.	U
2	Compare and contrast file system with database system. *[W23]*	A
3	Explain responsibilities of database administrator. *[W23]*	U
4	Write a short note on data dictionary.	U
5	List data type of SQL. Explain character data type.	R, U
6	Explain CREATE TABLE command with example.	U
7	Identify differences between TRUNCATE and DELETE. *[W23]*	A
8	Explain SQL UPDATE command with example.	U

Assignment (Descriptive Questions)

Sr. No	Questions	Bloom's Taxonomy Level
9	Explain various DDL commands with example. *[S23]*	U, A
	OR	
	Explain DDL commands. Create an employee table using at least 3 different data types. *[S24]*	
10	Discuss about Data Manipulation Command. *[W23]*	A
	OR	
	Explain DML commands with suitable examples. *[S24]*	
11	Explain various DCL commands with example. *[W22] [S23] [S24]*	U
12	Write a short note on COMMIT command.	U
13	Explain ROLLBACK with example.	U
14	Explain the purpose of SAVEPOINT. *[S23]*	U
15	Explain TCL commands with example. *[W23]*	U
16	Identify differences between GRANT and REVOKE. *[W23]*	A

Bloom's Taxonomy Levels: R (Remember), U (Understanding), A (Apply)

Thank you...

Prepared by:

Uresh N. Parmar

Lecturer, Computer Dept.
C U Shah Govt. Polytechnic
Surendranagar