Group A

Assignment No 2

Title of the Assignment: SQL Queries

a. Design and Develop SQL DDL statements which demonstrate the use of SQL objects such as Table, View, Index, Sequence, Synonym, different constraints etc.

b. Write at least 10 SQL queries on the suitable database application using SQL DML statements.

Objective: To understand and demonstrate DDL statements on various SQL objects.

Outcome:

1. Students will be able to learn and understand various DDL queries like create, drop, truncate.
2. Students will be able to demonstrate creating and dropping SQL objects like table, view, sequence, index etc.

Theory:

❖ **DDL-**

Data Definition Language (DDL) statements are used to define the database structure or schema. Data Definition Language understanding with database schemas and describes how the data should consist in the database, therefore language statements like CREATE TABLE or ALTER TABLE belongs to the DDL. DDL is about "metadata".

DDL includes commands such as CREATE, ALTER and DROP statements. DDL is used to CREATE, ALTER OR DROP the database objects (Table, Views, Users).

Data Definition Language (DDL) are used different statements :

1. CREATE - to create objects in the database
2. ALTER - alters the structure of the database
3. DROP - delete objects from the database
4. TRUNCATE - remove all records from a table, including all spaces allocated for the records are removed
5. RENAME - rename an object

1. CREATE

The CREATE TABLE statement is used to create a new table in a database.

Syntax

```
CREATE TABLE table_name
(
   column1 datatype,
   column2 datatype,
   column3 datatype,
  ....
);
```

Example

```
CREATE TABLE Persons (
   PersonID int,
   LastName varchar(255),
   FirstName varchar(255),
   Address varchar(255),
   City varchar(255)
);
```

2. ALTER

The ALTER TABLE statement is used to add, delete, or modify columns in an existing table.
The ALTER TABLE statement is also used to add & drop various constraints on an existing table.

a. ALTER TABLE - ADD Column

To add a column in a table

Syntax: ALTER TABLE table_name ADD column_name datatype;

Example:ALTER TABLE Customers ADD Email varchar(255);

b. ALTER TABLE - DROP Column

To delete a column in a table,(notice that some database systems don't allow deleting a column):

Syntax: ALTER TABLE table_name DROP COLUMN column_name;

Example: ALTER TABLE Customers DROP COLUMN Email;

c. ALTER TABLE - MODIFY COLUMN

To change the data type of a column in a table

Syntax: ALTER TABLE table_name MODIFY COLUMN column_name datatype;

Example: ALTER TABLE Persons ADD DateOfBirth date;

3. DROP

The DROP TABLE statement is used to drop an existing table in a database.

Syntax: DROP TABLE table_name;

Example: DROP TABLE Shippers;

4. TRUNCATE

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

Syntax: TRUNCATE TABLE table_name;
Example: TRUNCATE TABLE Shippers;

5. RENAME

RENAME statement you can rename a table.

Syntax: RENAME TABLE (tbl_name) TO (new_tbl_name);
Example: RENAME table student to info;

❖ DML-

DML is short name of Data Manipulation Language which deals with data manipulation and includes most common SQL statements such SELECT, INSERT, UPDATE, DELETE, etc., and it is used to store, modify, retrieve, delete and update data in a database.

Data Manipulating Language (DML) are used different statements :

1. SELECT - retrieve data from a database
2. INSERT - insert data into a table
3. UPDATE - updates existing data within a table
4. DELETE - Delete all records from a database table

1. SELECT

The SELECT statement is used to select data from a database.

Syntax: SELECT column1, column2, ...FROM table_name;
Example: SELECT CustomerName, City, Country FROM Customers;

Syntax: SELECT * FROM table_name;
Example: SELECT * FROM Customers;

2. INSERT

The INSERT INTO statement is used to insert new records in a table. It is possible to write the INSERT INTO statement in two ways:

1. Specify both the column names and the values to be inserted:
   Syntax: INSERT INTO table_name (column1, column2, column3, ...) VALUES (value1, value2, value3, ...);
   Example: INSERT INTO Customers (CustomerName, ContactName, Address, City, PostalCode, Country) VALUES ('Cardinal', 'Tom B. Erichsen', 'Skagen 21', 'Stavanger', '4006', 'Norway');

2. If you are adding values for all the columns of the table, you do not need to specify the

column names in the SQL query. However, make sure the order of the values is in the same order as the columns in the table. Here, the INSERT INTO syntax would be as follows:
Syntax: INSERT INTO table_name VALUES (value1, value2, value3, ...);
Example: INSERT INTO Customers (CustomerName, City, Country)VALUES ('Cardinal', 'Stavanger', 'Norway');

3. **UPDATE**
The UPDATE statement is used to modify the existing records in a table.
Syntax: UPDATE table_name SET column1 = value1, column2 = value2, ...WHERE condition;
Example: UPDATE Customers SET ContactName = 'Alfred Schmidt', City = 'Frankfurt' WHERE CustomerID = 1;

4. **DELETE**
The DELETE statement is used to delete existing records in a table.
Syntax: DELETE FROM table_name WHERE condition;
Example: DELETE FROM Customers WHERE CustomerName='Alfreds Futterkiste';

❖ VIEWS

- A view is a virtual table based on the result-set of an SQL statement.
- A view contains rows and columns, just like a real table. The fields in a view are fields from one or more real tables in the database.
- You can add SQL statements and functions to a view and present the data as if the data were coming from one single table.

1. CREATE VIEW

Syntax: CREATE VIEW view_name AS SELECT column1, column2, ... FROM table_name WHERE condition;
Examples: CREATE VIEW [Brazil Customers] AS SELECT CustomerName, ContactName FROM Customers WHERE Country = 'Brazil'; SELECT * FROM [Brazil Customers];

2. CREATE OR REPLACE VIEW

Syntax: CREATE OR REPLACE VIEW view_name AS SELECT column1, column2, ... FROM table_name WHERE condition;
Examples: CREATE OR REPLACE VIEW [Brazil Customers] AS SELECT CustomerName, ContactName, City FROM Customers WHERE Country = 'Brazil';

3. DROPPING A VIEW

A view is deleted with the DROP VIEW statement.

Syntax: DROP VIEW view_name;

Example: DROP VIEW Brazil-Customers

❖ SEQUENCE

- Auto-increment allows a unique number to be generated automatically when a new record is inserted into a table.
- Often this is the primary key field that we would like to be created automatically every time a new record is inserted.

Example

```
CREATE TABLE Persons (
   Personid int NOT NULL AUTO_INCREMENT,
   LastName varchar(255) NOT NULL,
   FirstName varchar(255),
   Age int,
   PRIMARY KEY (Personid)
    );
```

- By default, the starting value for AUTO_INCREMENT is 1, and it will increment by 1 for each new record.

- To let the AUTO_INCREMENT sequence start with another value, use the following SQL statement:

Example: ALTER TABLE Persons AUTO_INCREMENT=100; INSERT INTO Persons (FirstName,LastName)VALUES ('Lars','Monsen');

❖ SYNONYM

A SYNONYM provides another name for database object, referred to as original object, that may exist on a local or another server.

Syntax: Call sys.create_synonym_db(‘old database name’,’new database name’);

Example: Call sys.create_synonym_db(‘student,’information);

❖ CONSTRAINTS

- Constraints are used to specify rules for data in a table.
- Constraints are used to limit the type of data that can go into a table.
- This ensures the accuracy and reliability of the data in the table.
- If there is any violation between the constraint and the data action, the action is aborted.

- **Types of** constraints
  1) NOT NULL Constraint
  2) Unique Constraint
  3) Primary key
  4) Foreign key
  5) Check constraint
  6) Default constraint

1) NOT NULL Constraint

- By default, a column can hold NULL values.
- The NOT NULL constraint enforces a column to NOT accept NULL values.
- This enforces a field to always contain a value, which means that you cannot insert a new record, or update a record without adding a value to this field.

i) NOT NULL on CREATE TABLE

Example: CREATE TABLE Persons (ID int NOT NULL, LastName varchar(255) NOT NULL, FirstName varchar(255) NOT NULL, Age int);

ii) NOT NULL on ALTER TABLE

Example: ALTER TABLE Persons MODIFY COLUMN Age int NOT NULL;

2) UNIQUE Constraint

- The UNIQUE constraint ensures that all values in a column are different.
- Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.
- A PRIMARY KEY constraint automatically has a UNIQUE constraint.
- However, you can have many UNIQUE constraints per table, but only one PRIMARY KEY constraint per table.

i. UNIQUE Constraint on CREATE TABLE(One column)

Example:
```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    UNIQUE (ID)
);
```

ii. UNIQUE Constraint on CREATE TABLE(Multiple column)

Example:
```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    CONSTRAINT UC_Person UNIQUE (ID,LastName)
);
```

iii. UNIQUE Constraint on ALTER TABLE(One column)

Example: ALTER TABLE Persons ADD UNIQUE (ID);

iv. UNIQUE Constraint on ALTER TABLE(Multiple column)

Example: ALTER TABLE Persons ADD CONSTRAINT UC_Person
UNIQUE (ID,LastName);

v. DROP a UNIQUE Constraint

Example: ALTER TABLE Persons DROP INDEX UC_Person;

3) PRIMARY KEY Constraint

- The PRIMARY KEY constraint uniquely identifies each record in a table.
- Primary keys must contain UNIQUE values, and cannot contain NULL values.
- A table can have only ONE primary key; and in the table, this primary key can consist of single or multiple columns (fields).

i. PRIMARY KEY on CREATE TABLE

Example:
```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    PRIMARY KEY (ID)
);
```

ii. PRIMARY KEY on ALTER TABLE

Example: ALTER TABLE Persons ADD PRIMARY KEY (ID);

iii. DROP a PRIMARY KEY Constraint

Example: ALTER TABLE Persons DROP PRIMARY KEY;

4) FOREIGN KEY Constraint

- The FOREIGN KEY constraint is used to prevent actions that would destroy links between tables.
- A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another table.
- The table with the foreign key is called the child table, and the table with the primary key is called the referenced or parent table.

i. FOREIGN KEY on CREATE TABLE

Example:
```
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);
```

ii. FOREIGN KEY on ALTER TABLE

Example:
```
ALTER TABLE Orders ADD FOREIGN KEY (PersonID)
    REFERENCES Persons(PersonID);
```

iii. DROP a FOREIGN KEY Constraint

Example:
```
ALTER TABLE Orders DROP FOREIGN KEY FK_PersonOrder;
```

5) CHECK Constraint

- The CHECK constraint is used to limit the value range that can be placed in a column.
- If you define a CHECK constraint on a column it will allow only certain values for this column.
- If you define a CHECK constraint on a table it can limit the values in certain columns based on values in other columns in the row.

i. CHECK on CREATE TABLE (One column)

Example:
```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    CHECK (Age>=18)
);
```

ii. CHECK on CREATE TABLE (Multiple column)

Example:
```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    City varchar(255),
    CONSTRAINT CHK_Person CHECK (Age>=18 AND
    City='Sandnes')
);
```

iii. CHECK on ALTER TABLE (One column)

Example: ALTER TABLE Persons ADD CHECK (Age>=18);

iv. CHECK on ALTER TABLE (Multiple columns)

Example: ALTER TABLE Persons ADD CONSTRAINT CHK_PersonAge
CHECK (Age>=18 AND City='Sandnes');

v. DROP a CHECK Constraint

Example: ALTER TABLE Persons DROP CONSTRAINT CHK_PersonAge;

6) DEFAULT Constraint

- The DEFAULT constraint is used to set a default value for a column.
- The default value will be added to all new records, if no other value is specified.

i. DEFAULT on CREATE TABLE

Example:
```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    City varchar(255) DEFAULT 'Sandnes'
);
```

ii. DEFAULT on ALTER TABLE

Example: ALTER TABLE Persons ALTER City SET DEFAULT 'Sandnes';

iii. DROP a DEFAULT Constraint

Example: ALTER TABLE Persons ALTER City DROP DEFAULT

Conclusion: In this assignment, we have studied and demonstrated various DDL,DML statements in SQL.

Viva Questions:

- What is DDL & DML with examples?
- What is difference between drop and truncate?
- What mean by constraints and explain its types.
- What difference between unique and primary key constraints?
- What is mean by view & how to create a view?

Date:	
Marks obtained:	
Sign of course coordinator:	
Name of course Coordinator:	