### Assignment No: 02

**Title:** SQL Queries:

* 1. Design and Develop SQLDDL statements which demonstrate the use of SQL objects suchas Table, View, Index, Sequence, Synonym, different constraints etc.
  2. Write at least 10 SQL queries on the suitable database application using SQL DMLstatements.

**Aim:** Design and Develop SQL DDL statements which demonstrate the use of SQL objects suchas Table, View, Index, Sequence, Synonym. Design and Develop SQL DML statements

**Prerequisites:** Basics of database, data base Languages.

**Objective:** To learn the various DDL commands, view concepts, types of Indexes, sequence, Synonym and its implementation.To learn the various DML commands

**Outcome:** Develop the ability to handle basic operations on databases.

**Theory:**

**DDL:**

Data Definition Language helps you to define the database structure or schema. Let’s learn about DDL commands with syntax.

Database Tables

A database most often contains one or more tables. Each table is identified by a name (e.g. "Customers" or "Orders"). Tables contain records (rows) with data.

**The SQL CREATE TABLE Statement**

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)

);

**View:**

In SQL, 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 functions, WHERE, and JOIN statements to a view and present the data as if the data were coming from one single table.

**CREATE VIEW Syntax**

CREATE VIEW view\_name

AS SELECT column1, column2, ... FROM table\_name

WHERE condition;

**SQL CREATE VIEW Examples**

If you have the Northwind database you can see that it has several views installed by default.

The view "Current Product List" lists all active products (products that are not discontinued) from the "Products" table. The view is created with the following SQL:

CREATE VIEW [Current Product List] AS

SELECT ProductID, ProductName FROM Products

WHERE Discontinued = No;

Then, we can query the view as follows: SELECT \* FROM [Current Product List];

**SQL Updating a View**

**You can update a view by using the following syntax:**

SQL CREATE OR REPLACE VIEW SyntaxCREATE OR REPLACE VIEW view\_name AS SELECT column1, column2, ...

FROM table\_name WHERE condition;

Now we want to add the "Category" column to the "Current Product List" view. We will update the view with the following SQL:

CREATE OR REPLACE VIEW [Current Product List] AS SELECT ProductID, ProductName, Category

FROM Products

WHERE Discontinued = No; **SQL Dropping a View**

You can delete a view with the DROP VIEW command.

SQL DROP VIEW Syntax

DROP VIEW view\_name; **Index:**

SQL CREATE INDEX Statement

The CREATE INDEX statement is used to create indexes in tables.

Indexes are used to retrieve data from the database very fast. The users cannot see the indexes, they are just used to speed up searches/queries.

**CREATE INDEX Syntax**

Creates an index on a table. Duplicate values are allowed:

CREATE INDEX *index\_name*

ON *table\_name* (*column1*, *column2*, ...);

**CREATE UNIQUE INDEX Syntax**

Creates a unique index on a table. Duplicate values are not allowed:

CREATE UNIQUE INDEX *index\_name*

ON *table\_name* (*column1*, *column2*, ...);

**CREATE INDEX Example**

The SQL statement below creates an index named "idx\_lastname" on the "LastName" column in the "Persons" table:

**CREATE INDEX Example**

The SQL statement below creates an index named "idx\_lastname" on the "LastName" column in the "Persons" table:

If you want to create an index on a combination of columns, you can list the column names within the parentheses, separated by commas:

CREATE INDEX idx\_pname

ON Persons (LastName, FirstName);

**DROP INDEX Statement**

The DROP INDEX statement is used to delete an index in a table.

DB2/Oracle:

DROP INDEX *index\_name*;

MySQL:

ALTER TABLE *table\_name*

DROP INDEX *index\_name*;

**Sequence:**

A sequence is a set of integers 1, 2, 3, ... that are generated in order on demand. Sequences are frequently used in databases because many applications require each row in a table to contain a unique value and sequences provide an easy way to generate them. This chapter describes how to use sequences in MySQL.

Using AUTO\_INCREMENT column:

The simplest way in MySQL to use Sequences is to define a column as AUTO\_INCREMENT and leave rest of the things to MySQL to take care.

Example:

Try out the following example. This will create table and after that it will insert few rows in this table where it is not required to give record ID because it's auto incremented by MySQL. mysql> CREATE TABLE insect

-> (

-> id INT UNSIGNED NOT NULL AUTO\_INCREMENT,

-> PRIMARY KEY (id),

-> name VARCHAR(30) NOT NULL, # type of insect

-> date DATE NOT NULL, # date collected

-> origin VARCHAR(30) NOT NULL # where collected

);

Query OK, 0 rows affected (0.02 sec)

**DML:**

Data Manipulation Language (DML) allows you to modify the database instance by inserting, modifying, and deleting its data. It is responsible for performing all types of data modification in a database.

**DML Commands**

*The SQL INSERT INTO Statement*

The INSERT INTO statement is used to insert new records in a table. **INSERT INTO Syntax**

It is possible to write the INSERT INTO statement in two ways:

1. Specify both the column names and the values to be inserted:

INSERT INTO *table\_name* (*column1*, *column2*, *column3*, ...) VALUES (*value1*, *value2*, *value3*, ...);

1. 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:

INSERT INTO *table\_name*

VALUES (*value1*, *value2*, *value3*, ...);

*The SQL UPDATE Statement*

The UPDATE statement is used to modify the existing records in a table.

**UPDATE Syntax**

UPDATE *table\_name*

SET *column1* = *value1*, *column2* = *value2*, ... WHERE *condition*;

*The SQL DELETE Statement*

The DELETE statement is used to delete existing records in a table.

**DELETE Syntax**

DELETE FROM *table\_name* WHERE *condition*;

**Conclusion:** Thus we have implemented the DDL and DML commands

**Questions:**

* 1. Explain create view and drop view.
  2. Explain concept of Index in Mysql.
  3. Explain DML statements with syntax