LAB # 2

To Demonstrate the usage of DDL & DML category commands of SQL

Objectives

- To demonstrate how to create and alter schema of a database
- To demonstrate how to modify entities and their attributes

SQL DDL and DML

SQL can be divided into two parts: The Data Manipulation Language (DML) and the Data Definition Language (DDL).

The DDL part of SQL permits database tables to be created or deleted. It also defines indexes (keys), specify links between tables, and impose constraints between tables.

The most important DDL statements in SQL are:

- **CREATE DATABASE** - creates a new database
- **ALTER DATABASE** - modifies a database
- **CREATE TABLE** - creates a new table
- **ALTER TABLE** - modifies a table
- **DROP TABLE** - deletes a table
- **CREATE INDEX** - creates an index (search key)
- **DROP INDEX** - deletes an index

The query and update commands form the DML part of SQL:

- **SELECT** - extracts data from a database
- **UPDATE** - updates data in a database
- **DELETE** - deletes data from a database
- **INSERT INTO** - inserts new data into a database

SQL INSERT INTO Statement

The INSERT INTO statement is used to insert new record or row in a table.

SQL INSERT INTO Syntax

It is possible to write the INSERT INTO statement in two forms.

The first form doesn't specify the column names where the data will be inserted, only their values:

```
INSERT INTO table_name VALUES
(value1, value2, value3,...)
```

The second form specifies both the column names and the values to be inserted:

```
INSERT INTO table_name (column1, column2, column3,...)
VALUES (value1, value2, value3,...)
```

SQL INSERT INTO Example

We have the following "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger

Now we want to insert a new row in the "Persons" table. We use the following SQL statement:

```
INSERT INTO Persons
VALUES (4,'Nilsen', 'Johan', 'Bakken 2', 'Stavanger')
```

The "Persons" table will now look like this:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger

Insert Data Only in Specified Columns

It is also possible to only add data in specific columns.

The following SQL statement will add a new row, but only add data in the "P_Id", "LastName" and the "FirstName" columns:

```
INSERT INTO Persons (P_Id, LastName, FirstName)
VALUES (5, 'Tjessem', 'Jakob')
```

The "Persons" table will now look like this:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob		

SQL UPDATE Statement

The UPDATE statement is used to update records in a table

SQL UPDATE Syntax

```
UPDATE table_name
SET column1=value, column2=value2,...
WHERE some_column=some_value
```

Note: Notice the WHERE clause in the UPDATE syntax. The WHERE clause specifies which record or records that should be updated. If you omit the WHERE clause, all records will be updated!

SQL UPDATE Example

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob		

Now we want to update the person "Tjessem, Jakob" in the "Persons" table. We use the following SQL statement:

```
UPDATE Persons
SET Address='Nissestien 67', City='Sandnes'
WHERE LastName='Tjessem' AND FirstName='Jakob'
```

The "Persons" table will now look like this:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob	Nissestien 67	Sandnes

SQL UPDATE Warning

Be careful when updating records. If we had omitted the WHERE clause in the example above, like this:

```
UPDATE Persons
SET Address='Nissestien 67', City='Sandnes'
```

The "Persons" table would have looked like this:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Nissestien 67	Sandnes
2	Svendson	Tove	Nissestien 67	Sandnes
3	Pettersen	Michael	Nissestien 67	Sandnes
4	Nilsen	Johan	Nissestien 67	Sandnes
5	Tjessem	Jakob	Nissestien 67	Sandnes

SQL DELETE Statement

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

SQL DELETE Syntax

```
DELETE FROM table_name WHERE
some_column=some_value
```

Note: Notice the WHERE clause in the DELETE syntax. The WHERE clause specifies

Which record or records that should be deleted. If you omit the WHERE clause, all records will be deleted!

SQL DELETE Example

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger
4	Nilsen	Johan	Bakken 2	Stavanger
5	Tjessem	Jakob	Nissestien 67	Sandnes

Now we want to delete the person "Tjessem, Jakob" in the "Persons" table.

We use the following SQL statement:

```
DELETE FROM Persons
WHERE LastName='Tjessem' AND FirstName='Jakob'
```

The "Persons" table will now look like this:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger

4	Nilsen	Johan	Bakken 2	Stavanger

Delete All Rows

It is possible to delete all rows in a table without deleting the table. This means that the table structure, attributes, and indexes will be intact:

```
DELETE FROM
table_name or
DELETE * FROM table_name
```

Note: Be very careful when deleting records. You cannot undo this statement!

SQL SELECT Statement

The SELECT statement is used to select data from a database.The result is stored in a result table, called the result-set.

The syntax used for SELECT query is:

```
SELECT       column_name(s)
FROM table_name
```

And

SELECT * FROM table_name

Note: SQL is not case sensitive. SELECT is the same as select.

Example

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger

Now we want to select the content of the columns named "LastName" and "FirstName" from the table above.

We use the following SELECT statement:

```
SELECT LastName,FirstName FROM Persons
```

The result-set will look like this:

LastName	FirstName
Hansen	Christ
Svendson	Tove
Pettersen	Michael

SELECT * Example

Now we want to select all the columns from the "Persons" table. We use the following SELECT statement:

```
SELECT * FROM Persons
```

Tip: The asterisk (*) is a quick way of selecting all columns! The result-set will look like this:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger

The SQL SELECT DISTINCT Statement

In a table, some of the columns may contain duplicate values. This is not a problem, however, sometimes you will want to list only the different (distinct) values in a table.

The DISTINCT keyword can be used to return only distinct (different) values. Its Syntax is:

```
SELECT DISTINCT column_name(s) FROM
table_name
```

Example

Now we want to select only the distinct values from the column named "City" from the table above.

We use the following SELECT statement:

SELECT DISTINCT City FROM Persons

The result-set will look like this:

City Sandnes Stavanger

The WHERE Clause

The WHERE clause is used to extract only those records that fulfill a specified criterion. The syntax of the command is:

SELECT column_name(s) FROM table_name

WHERE column_name operator value

Example

Now we want to select only the persons living in the city "Sandnes" from the table above. We use the following SELECT statement:

```
SELECT * FROM Persons
WHERECity='Sandnes'
```

Quotes Around Text Fields

SQL uses single quotes around text values (most database systems will also accept double quotes).

Although, numeric values should not be enclosed in quotes.

For text values:

This is correct:

SELECT * FROM Persons WHERE FirstName='Tove'

This is wrong:

SELECT * FROM Persons WHERE FirstName=Tove

For numeric values:

This is correct:

SELECT * FROM Persons WHERE Year=1965

This is wrong:

SELECT * FROM Persons WHERE Year='1965'

Operators Allowed in the WHERE Clause

With the WHERE clause, the following operators can be used:

Operator	Description
=	Equal
<>	Not equal
>	Greater than
<	Less than
>=	Greater than or equal
<=	Less than or equal
BETWEEN	Between an inclusive range
LIKE	Search for a pattern
IN	If you know the exact value you want to return for at least one of the columns

Note: In some versions of SQL the <> operator may be written as !=

The AND & OR Operators

The AND & OR operators are used to filter records based on more than one condition. The AND operator displays a record if both the first condition and the second condition is true. The OR operator displays a record if either the first condition or the second condition is true.

AND Operator Example

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger

Now we want to select only the persons with the first name equal to "Tove" AND the last name equal to "Svendson":

We use the following SELECT statement:

```
SELECT * FROM Persons
WHERE FirstName='Tove' AND
LastName='Svendson'
```

The result-set will look like this:

P_Id	LastName	FirstName	Address	City
2	Svendson	Tove	Borgvn 23	Sandnes

OR Operator Example

Now we want to select only the persons with the first name equal to "Tove" OR the first name equal to "Christ":

We use the following SELECT statement:

```
SELECT * FROM Persons
WHERE FirstName='Tove' OR
FirstName='Christ'
```

The result-set will look like this:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes

Combining AND & OR

You can also combine AND and OR (use parenthesis to form complex expressions).

Now we want to select only the persons with the last name equal to "Svendson" AND the first name equal to "Tove" OR to "Christ":

We use the following SELECT statement:

```
SELECT * FROM Persons WHERE
LastName='Svendson'
AND (FirstName='Tove' OR
FirstName='Christ')
```

The result-set will look like this:

P_Id	LastName	FirstName	Address	City

2	**Svendson**	**Tove**	**Borgvn 23**	**Sandnes**

The ORDER BY Keyword

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

If you want to sort the records in a descending order, you can use the DESC keyword. Its syntax is:

```
SELECT column_name(s)
FROM table_name
ORDER BY column_name(s) ASC|DESC
```

Example

The "Persons" table:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger
4	Nilsen	Tom	Vingvn 23	Stavanger

Now we want to select all the persons from the table above, however, we want to sort the persons by their last name.

We use the following SELECT statement:

```
SELECT * FROM Persons
ORDER BY LastName
```

The result-set will look like this:

P_Id	LastName	FirstName	Address	City
1	Hansen	Christ	Timoteivn 10	Sandnes
4	Nilsen	Tom	Vingvn 23	Stavanger
3	Pettersen	Michael	Storgt 20	Stavanger
2	Svendson	Tove	Borgvn 23	Sandnes

ORDER BY DESC Example

Now we want to select all the persons from the table above, however, we want to sort the persons descending by their last name.

We use the following SELECT statement:.

```
SELECT * FROM Persons ORDER
BY LastName DESC
```

The result-set will look like this:

P_Id	LastName	FirstName	Address	City
2	Svendson	Tove	Borgvn 23	Sandnes
3	Pettersen	Michael	Storgt 20	Stavanger
4	Nilsen	Tom	Vingvn 23	Stavanger
1	Hansen	Christ	Timoteivn 10	Sandnes

Lab Tasks

Task 1

Create the following table using SQL and using the INSERT INTO command, insert the following values in the table created.

Name	Reg_No	Courses	Course_Code	Offered_By
Ali	01	DIP	1001	Mr. A

Basit	02	DBMS	1002	Mr. X
Akram	03	OS	1003	Mr. Y
Asad	04	DBMS	1002	Mr. X
Zeeshan	05	DIP	1001	Mr. A
Muneer	06	OS	1003	Mr. Y
Shafqat	07	NM	1004	Mr. H
Ahsan	08	OS	1003	Mr. Y
Ikram	09	DIP		
Hassan	10			

Task 2

Using the UPDATE statement, update the above table for the following values:

Name	Reg_No	Courses	Course_Code	Offered_By
Ali	01	DIP	1001	Mr. A
Basit	02	DBMS	1002	Mr. X
Akram	03	OS	1003	Mr. Y
Asad	04	DBMS	1002	Mr. X
Zeeshan	05	DIP	1001	Mr. A
Muneer	06	OS	1003	Mr. Y
Shafqat	07	NM	1004	Mr. H
Ahsan	08	OS	1003	Mr. Y
Ikram	09	DIP	1001	Mr. A
Hassan	10	DSP	1005	Mr. Z

TASK 3

Using the DELETE statement, delete the record for the student having name Akram and Ahsan in the above table. Also delete the record for the course having course code=1001.

TASK 4

Select distinct values from the above table for the last three columns.

TASK 5

Sort the above table in descending order by their name.

Rubric for Lab Assessment

The student performance for the assigned task during the lab session was:			
Excellent	The student completed assigned tasks without any help from the instructor and showed the results appropriately.	4	
Good	The student completed assigned tasks with minimal help from the instructor and showed the results appropriately.	3	
Average	The student could not complete all assigned tasks and showed partial results.	2	
Worst	The student did not complete assigned tasks.	1	

Instructor Signature: _____ Date: _____