Lab 2: Data Manipulation Language

**Introduction:**

A data manipulation language (DML) is a family of syntax elements similar to a computer programming language used for inserting, deleting and updating data in a database. Performing read-only queries of data is sometimes also considered a component of DML.

A popular data manipulation language is that of Structured Query Language (SQL), which is used to retrieve and manipulate data in a relational database. Other forms of DML are those used by IMS/DLI, CODASYL databases, such as IDMS and others.

**DML Parts:**

- INSERT.
- SELECT.
- UPDATE.
- DELETE.

The INSERT INTO statement is used to insert new rows in a table. It used to add static data or it can take data from another table via some query (discussed later).

**Formula:**

```
insert into tableName
values(value1 , value2 , value3 , ...)
```
Note that value1 mapped to the first column in the table and value2 mapped to the second column in the table and so on.

Example:

```
INSERT INTO "EMPLOYEE" VALUES ('Richard', 'K', 'Marini', '653298653', '30-DEC-52', '98 Oak Forest,Katy,TX', 'M', 37000, '987654321', 4 )
```

- **Specific column:**

  ```
  insert into tableName (column1.column2.column3....)
  values(value1 , value2 , value3 ....)
  ```

  Note that value1 mapped column1 and value2 mapped to column2 and so on.

  Example:

  ```
  INSERT INTO "EMPLOYEE" ("FNAME", "LNAME", "SSN") VALUES ('Ricjhard', 'Mjarini', '6553');
  ```

- **Multi rows**

  We can insert multi row in one query as show next:

  ```
  insert into tableName (column1.column2.column3....)
  values(value1 , value2 , value3 ....)
  ,(value1 , value2 , value3 ....)
  ,(value1 , value2 , value3 ....)
  ```

  Example:

  ```
  insert into "EMPLOYEE"("FNAME","LNAME","SSN") VALUES ('ahmed', 'hani',12316) ,('jahed', 'hani',12312),('fady', 'hani',12313),('mohammed', 'hani',12314)
  ```

<table>
<thead>
<tr>
<th>FNAME</th>
<th>MINIT</th>
<th>LNAME</th>
<th>SSN</th>
<th>BDATE</th>
<th>ADDRESS</th>
<th>SEX</th>
<th>SALARY</th>
<th>SUPERSSN</th>
<th>DNO</th>
</tr>
</thead>
<tbody>
<tr>
<td>Richard</td>
<td>Mjerni</td>
<td></td>
<td>6553</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jehad</td>
<td></td>
<td>hani</td>
<td>12312</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fady</td>
<td></td>
<td>hani</td>
<td>12313</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mohammed</td>
<td></td>
<td>hani</td>
<td>12314</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ahmed</td>
<td></td>
<td>hani</td>
<td>12316</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Richard</td>
<td></td>
<td>Mjerni</td>
<td>653298653</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Franklin</td>
<td>I</td>
<td>Wong</td>
<td>339445555</td>
<td>1955-12-08</td>
<td>638 Vogt,HO M</td>
<td>40000</td>
<td>888665555</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Richard</td>
<td></td>
<td>Marini</td>
<td>653298653</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ramesh</td>
<td>K</td>
<td>Narayan</td>
<td>668584444</td>
<td>1982-09-15</td>
<td>975 Fire Co M</td>
<td>45000</td>
<td>333455555</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>James</td>
<td>E</td>
<td>Brog</td>
<td>888665555</td>
<td>1987-11-10</td>
<td>650 Store,HO M</td>
<td>55000</td>
<td></td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Jennifer</td>
<td>S</td>
<td>Wallace</td>
<td>987654321</td>
<td>1941-06-20</td>
<td>291 Berry,BF</td>
<td>43000</td>
<td>888665555</td>
<td>4</td>
<td></td>
</tr>
<tr>
<td>Alicia</td>
<td>O</td>
<td>Zelaya</td>
<td>999587777</td>
<td>1965-07-19</td>
<td>3311 Castle F</td>
<td>25000</td>
<td>987654321</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>
The SELECT statement is used to select data from a database. The result is stored in a result table, called the result-set.

- Basic

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

Example:

```
SELECT "BDATE", "ADDRESS" FROM "EMPLOYEE" WHERE "FNAME"='John' AND "MINIT"='B' AND "LNAME"='Smith';
```

- Star (*)

Star * is a fast alternative to all column name.

```
SELECT * FROM "EMPLOYEE"
```
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.

Formula:

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

Example:

```
SELECT DISTINCT "SALARY" FROM "EMPLOYEE"
```

You can give a table or a column another name by using an alias. This can be a good thing to do if you have very long or complex table names or column names. An alias name could be anything, but usually it is short.

Formula:

```
SELECT column_name(s) as column_alias_name
FROM table_name as table_alias_name
```
Example:

```
SELECT E."FNAME", E."LNAME", S."FNAME", S."LNAME" FROM "EMPLOYEE" AS E, "EMPLOYEE" AS S WHERE E."SUPERSSN"=S."SSN"
```

The WHERE clause is used to extract only those records that fulfill a specified criterion.

Formula:

```
SELECT column_name(s) as column_alias_name
FROM table_name as table_alias_name
where column_name operator value
```

Operator:

<table>
<thead>
<tr>
<th>Operator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>=</td>
<td>Equal</td>
</tr>
<tr>
<td>!=</td>
<td>Not equal</td>
</tr>
<tr>
<td>&gt;</td>
<td>Greater than</td>
</tr>
<tr>
<td>&lt;</td>
<td>Less than</td>
</tr>
<tr>
<td>&gt;=</td>
<td>Greater than or equal</td>
</tr>
<tr>
<td>&lt;=</td>
<td>Less than or equal</td>
</tr>
<tr>
<td>BETWEEN</td>
<td>Between an inclusive range</td>
</tr>
<tr>
<td>LIKE</td>
<td>Search for a pattern</td>
</tr>
<tr>
<td>%</td>
<td>any number of any characters.</td>
</tr>
<tr>
<td>_</td>
<td>one character.</td>
</tr>
<tr>
<td>IN</td>
<td>If you know the exact value you want to return for at least one of the columns</td>
</tr>
</tbody>
</table>
Example:

```sql
SELECT "FNAME", "LNAME" FROM "EMPLOYEE" WHERE "ADDRESS" LIKE '%Houston, TX%';
```

<table>
<thead>
<tr>
<th>FNAME</th>
<th>LNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>Franklin</td>
<td>Wong</td>
</tr>
<tr>
<td>James</td>
<td>Brog</td>
</tr>
<tr>
<td>John</td>
<td>Smith</td>
</tr>
</tbody>
</table>

Order By

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.

Formula:

```sql
SELECT column_name(s)
FROM table_name
ORDER BY column_name [asc] [desc]
```

Example:

```sql
SELECT "DNAME", "LNAME", "FNAME", "PNAME" FROM "DEPARTMENT", "EMPLOYEE", "PROJECT"
WHERE "DNUMBER"="DNO" AND "SSN"="ESSN" AND "FNO"="FNUMBER" ORDER BY "DNAME", "LNAME"
```

<table>
<thead>
<tr>
<th>DNAME</th>
<th>LNAME</th>
<th>FNAME</th>
<th>PNAME</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Wallace</td>
<td>Jennifer</td>
<td>Reorganization</td>
</tr>
<tr>
<td>2</td>
<td>Wallace</td>
<td>Jennifer</td>
<td>Newbenefits</td>
</tr>
<tr>
<td>3</td>
<td>Zelaya</td>
<td>Alicia</td>
<td>Computerization</td>
</tr>
<tr>
<td>4</td>
<td>Zelaya</td>
<td>Alicia</td>
<td>Newbenefits</td>
</tr>
<tr>
<td>5</td>
<td>Headquar</td>
<td>Brog</td>
<td>Reorganization</td>
</tr>
<tr>
<td>6</td>
<td>Narayan</td>
<td>Ramesh</td>
<td>ProductZ</td>
</tr>
<tr>
<td>7</td>
<td>Smith</td>
<td>John</td>
<td>ProductX</td>
</tr>
<tr>
<td>8</td>
<td>Smith</td>
<td>John</td>
<td>ProductY</td>
</tr>
<tr>
<td>9</td>
<td>Wong</td>
<td>Franklin</td>
<td>Reorganization</td>
</tr>
<tr>
<td>10</td>
<td>Wong</td>
<td>Franklin</td>
<td>Computerization</td>
</tr>
<tr>
<td>11</td>
<td>Wong</td>
<td>Franklin</td>
<td>ProductZ</td>
</tr>
<tr>
<td>12</td>
<td>Wong</td>
<td>Franklin</td>
<td>ProductY</td>
</tr>
</tbody>
</table>
The UPDATE statement is used to update existing records in a table.

Formula:

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

Example:

```
UPDATE "PROJECT" SET "LOCATION" = 'Bellaire',
"DNUM" = 5 WHERE "PNUMBER"=10
```

Query returned successfully: one row affected, 61 ms execution time.

---

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

Formula:

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

Example:

```
DELETE FROM "EMPLOYEE" WHERE "SSN"=123456789;
```

Query returned successfully: one row affected, 11 ms execution time.