COMPUTER ORGANIZATION AND ARCHITECTURE

Lecture 10

Introduction to Data Manipulation Language

Dr Victoria Mukami

INTRODUCTION

This lecture starts with the introduction to data manipulation language (DML). We look at two main DML commands: INSERT and UPDATE. The INSERT command will be used to put in data to the tables we created, and the UPDATE will be used to update the details within the table. This is a direct continuation of Lecture 9 that dealt with the DDL commands.

Learning objectives

By the end of this topic, you should be able to:

1. Understand the INSERT and UPDATE data manipulation commands.
2. Insert data into various tables
3. Update table row details

OVERVIEW

So far, we have done so much. We have been able to create a database, we have been able to add tables to the database and even add fields to the various tables. This lecture will work on a couple of things. We will first work toward putting data into the tables we created. Next, we will work toward updating some of the data inserted. Once a database administrator has finished creating the database and tables, then data needs to be inserted. This is a lengthy process as all the data must be in the correct format and must be digitized in the event it was manual.

INTRODUCTION

The INSERT and UPDATE commands are data manipulation commands that allow a user to input data into existing tables. During the last lecture, we created three tables that were linked together, and an ERD diagram was created as shown in figure 1.

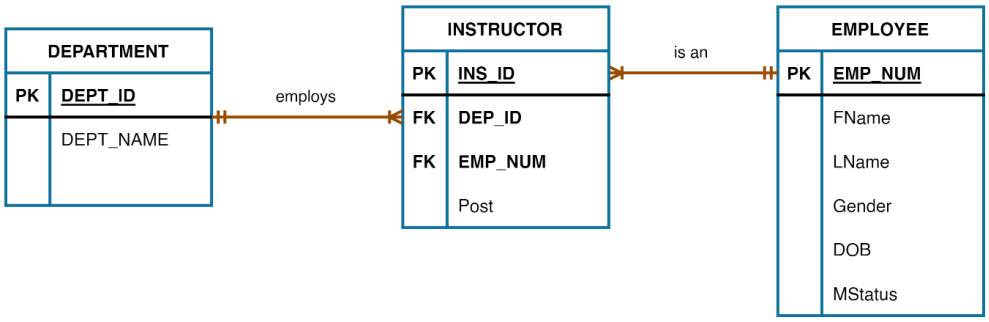

Figure 1: Simplified ERD

Each table has data that was inserted and is shown in Tables 1 to 3. The employee table had a data type change for the Gender Field, therefore we can update the records as shown in Table 1.

Table 1: Employee Data Table

Emp_Num	FName	LName	Gender	DOB	MStatus
EMP001	Agnes	Njue	Female	12/09/1997	Single
EMP002	Will	Omondi	Male	24/08/1989	Married
EMP003	Carol	Njagi	Female	09/01/1993	Single
EMP004	Mary	Munene	Female	06/08/1998	Married
EMP005	Cathy	Resley	Male	16/04/1989	Married
EMP006	Abdul	Muita	Male	30/07/1988	Single
EMP007	Steve	Kimuri	Male	02/03/1999	Married
EMP008	Ruth	Kimuli	Female	07/04/1996	Single

Table 2: Instructor Data Table

ID	Emp_Num	Dep_ID	Post
1	EMP001	CIT	Lecturer
2	EMP002	MTH	Chairman
3	EMP003	LAW	Dean
4	EMP004	CIT	Lecturer
5	EMP005	BUS	Lecturer
6	EMP006	EDU	Senior Lecturer
7	EMP007	MTH	Lecturer
8	EMP008	PSY	Senior Lecturer

Table 3: Department Data Table

Dep_ID	Dep_Name	School
CIT	Computing and Information Technology	SPAS
MTH	Statistics and Mathematics	SPAS
BUS	Business	Business
EDU	Education	Education
PSY	Psychology	Humanities
LAW	Law	Law

INSERT VALUES

The Insert command is used to place values into a table. We will use the command to input the values of the three tables. Ensure that you log in to the MySQL client and select the Bustani database. We will begin with the Employee Table, then the department table and finally the Instructor Table. This is just to ensure that the relationships within the tables are sequential. For instance, the instructor table picks records from the department and the employee tables. It is therefore prudent to start with the initial tables. The Insert Syntax is as follows:

```
INSERT INTO tablename
VALUES (‘data1’, ‘data2’, ‘data3’,.....)
```

The insert syntax will enter data in each cell depending on how many cells exist in the table. A cell in this case is the intersection between a column and a row. The value records are all separated by a comma and surrounded by single quotes. Let us now try a practical example.

Employee Table

The employee table has 8 records. We will first insert one of the records.

```
INSERT INTO Employee
VALUES ('EMP001','Agnes','Njue','Female','12/09/1997','Single')
```

MySQL output

```
mysql> INSERT INTO Employee
    -> VALUES ('EMP001','Agnes','Njue','Female','12/09/1997','Single');
ERROR 1292 (22007): Incorrect date value: '12/09/1997' for column 'DOB' at row 1
mysql>
```

Notice that we encounter an error. The error has to do with our date values that are not in the correct format. Remember in SQL the date data type is represented as YYYY-MM-DD. The corrected syntax should then accept our values.

```
INSERT INTO Employee
VALUES ('EMP001','Agnes','Njue','Female','1997-09-12','Single')
```

Output

```
mysql> INSERT INTO Employee
    -> VALUES ('EMP001','Agnes','Njue','Female','1997-09-12','Single');
Query OK, 1 row affected (0.03 sec)

mysql>
```

We can see the data within the table by using the SELECT statement. We will cover the SELECT statement during the next lecture. In the meantime the syntax we will use for this lecture is:

```
SELECT * FROM tablename;
```

To show the contents of the employee table we would use the following:

```
SELECT * FROM Employee;
```

Output

```
mysql> SELECT * FROM Employee;
+---------+-------+-------+--------+------------+---------+
| Emp_Num | FName | LName | Gender | DOB        | MStatus |
+---------+-------+-------+--------+------------+---------+
| EMP001  | Agnes | Njue  | Female | 1997-09-12 | Single  |
+---------+-------+-------+--------+------------+---------+
1 row in set (0.01 sec)

mysql>
```

As seen in the output, the details were all recorded within the table. To insert several records at once is possible. The records are all separated by brackets as shown.

```
INSERT INTO Employee
VALUES
('EMP003','Carol','Njagi','Female','1993-09-01','Single'),
('EMP004','Mary','Munene','Female','1998-08-06','Married'),
('EMP005','Cathy','Resley','Male','1989-04-16','Married'),
('EMP006','Abdul','Muita','Male','1988-07-30','Single'),
('EMP007','Steve','Kimuri','Male','1999-03-02','Married'),
('EMP008','Ruth','Kimuli','Female','1996-04-07','Single');
```

Output

```
mysql> INSERT INTO Employee
    -> VALUES
    -> ('EMP003','Carol','Njagi','Female','1993-09-01','Single'),
    -> ('EMP004','Mary','Munene','Female','1998-08-06','Married'),
    -> ('EMP005','Cathy','Resley','Male','1989-04-16','Married'),
    -> ('EMP006','Abdul','Muita','Male','1988-07-30','Single'),
    -> ('EMP007','Steve','Kimuri','Male','1999-03-02','Married'),
    -> ('EMP008','Ruth','Kimuli','Female','1996-04-07','Single');
Query OK, 6 rows affected (0.01 sec)
Records: 6  Duplicates: 0  Warnings: 0

mysql>
```

Showing all the records from the employee table.

```
SELECT * FROM Employee;
```

Output

```
mysql> SELECT * FROM Employee;
+---------+-------+--------+--------+------------+---------+
| Emp_Num | FName | LName  | Gender | DOB        | MStatus |
+---------+-------+--------+--------+------------+---------+
| EMP001  | Agnes | Njue   | Female | 1997-09-12 | Single  |
| EMP003  | Carol | Njagi  | Female | 1993-09-01 | Single  |
| EMP004  | Mary  | Munene | Female | 1998-08-06 | Married |
| EMP005  | Cathy | Resley | Male   | 1989-04-16 | Married |
| EMP006  | Abdul | Muita  | Male   | 1988-07-30 | Single  |
| EMP007  | Steve | Kimuri | Male   | 1999-03-02 | Married |
| EMP008  | Ruth  | Kimuli | Female | 1996-04-07 | Single  |
+---------+-------+--------+--------+------------+---------+
7 rows in set (0.00 sec)

mysql>
```

Department Table

```
mysql> INSERT INTO Department
    -> VALUES
    -> ('CIT','Computing and Information Technology','SPAS'),
    -> ('MTH','Statistics and Mathematics','SPAS'),
    -> ('BUS','Business','Business'),
    -> ('EDU','Education','Education'),
    -> ('PSY','Psychology','Humanities');
Query OK, 5 rows affected (0.01 sec)
Records: 5  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Department;
+--------+-----------------------------------+------------+
| Dep_ID | Dep_Name                          | School     |
+--------+-----------------------------------+------------+
| BUS    | Business                          | Business   |
| CIT    | Computing and Information Technology | SPAS    |
| EDU    | Education                         | Education  |
| MTH    | Statistics and Mathematics        | SPAS       |
| PSY    | Psychology                        | Humanities |
+--------+-----------------------------------+------------+
5 rows in set (0.00 sec)

mysql>
```

We missed adding the LAW department. This is shown in the output below.

```
mysql> INSERT INTO Department
    -> VALUES('LAW','Law','Law');
Query OK, 1 row affected (0.01 sec)

mysql> SELECT * FROM Department;
+--------+---------------------------------------+------------+
| Dep_ID | Dep_Name                              | School     |
+--------+---------------------------------------+------------+
| BUS    | Business                              | Business   |
| CIT    | Computing and Information Technology  | SPAS       |
| EDU    | Education                             | Education  |
| LAW    | Law                                   | Law        |
| MTH    | Statistics and Mathematics            | SPAS       |
| PSY    | Psychology                            | Humanities |
+--------+---------------------------------------+------------+
6 rows in set (0.00 sec)

mysql>
```

Instructor Table

Before we can insert the values for the instructor table, you will note that the Ins_ID is an incremental value. We can modify the column so that the numbers can be incrementally added. To do so, we use the ALTER statement and add a constraint called AUTO_INCREMENT.

```
ALTER TABLE Instructor
MODIFY Ins_ID INT NOT NULL UNIQUE AUTO_INCREMENT;
```

Output

```
mysql> ALTER TABLE Instructor
    -> MODIFY Ins_ID INT NOT NULL UNIQUE AUTO_INCREMENT;
Query OK, 0 rows affected, 1 warning (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 1

mysql> DESCRIBE Instructor;
+---------+-------------+------+-----+---------+----------------+
| Field   | Type        | Null | Key | Default | Extra          |
+---------+-------------+------+-----+---------+----------------+
| Ins_ID  | int         | NO   | PRI | NULL    | auto_increment |
| Emp_Num | char(6)     | YES  | MUL | NULL    |                |
| Dep_ID  | char(3)     | NO   | MUL | NULL    |                |
| Post    | varchar(15) | NO   |     | NULL    |                |
| School  | varchar(35) | NO   |     | NULL    |                |
+---------+-------------+------+-----+---------+----------------+
5 rows in set (0.00 sec)

mysql>
```

Now we can add our other values to the table. Note that the statement is slightly different.

```
INSERT INTO Instructor(Emp_Num,Dep_ID,Post)
VALUES
```

```
('EMP001','CIT','Lecturer'),
('EMP002','MTH','Chairman'),
('EMP003','LAW','Dean'),
('EMP004','CIT','Lecturer'),
('EMP005','BUS','Lecturer'),
('EMP006','EDU','Senior Lecturer'),
('EMP007','MTH','Lecturer'),
('EMP008','PSY',’Senior Lecturer');
```

The first line of the code adds the column names. The column names are added because the Ins_ID are incrementally being added.

```
mysql> INSERT INTO Instructor(Emp_Num,Dep_ID,Post)
    -> VALUES
    -> ('EMP001','CIT','Lecturer'),
    -> ('EMP002','MTH','Chairman'),
    -> ('EMP003','LAW','Dean'),
    -> ('EMP004','CIT','Lecturer'),
    -> ('EMP005','BUS','Lecturer'),
    -> ('EMP006','EDU','Senior Lecturer'),
    -> ('EMP007','MTH','Lecturer'),
    -> ('EMP008','PSY','Senior Lecturer');
Query OK, 8 rows affected (0.01 sec)
Records: 8  Duplicates: 0  Warnings: 0

mysql> SELECT * FROM Instructor;
+--------+---------+--------+-----------------+
| Ins_ID | Emp_Num | Dep_ID | Post            |
+--------+---------+--------+-----------------+
|     17 | EMP001  | CIT    | Lecturer        |
|     18 | EMP002  | MTH    | Chairman        |
|     19 | EMP003  | LAW    | Dean            |
|     20 | EMP004  | CIT    | Lecturer        |
|     21 | EMP005  | BUS    | Lecturer        |
|     22 | EMP006  | EDU    | Senior Lecturer |
|     23 | EMP007  | MTH    | Lecturer        |
|     24 | EMP008  | PSY    | Senior Lecturer |
+--------+---------+--------+-----------------+
8 rows in set (0.00 sec)

mysql>
```

What if we wanted to add values to one row that permits null values? For instance, let us assume that we create a table called Test that has the fields Test_No, Name, Type. The Type column allows for null values as shown in the output below.

```
mysql> DESCRIBE Test;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| Test_No | char(4)     | NO   | PRI | NULL    |       |
| Name    | varchar(25) | NO   |     | NULL    |       |
| Type    | varchar(25) | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
3 rows in set (0.00 sec)

mysql>
```

As shown Type allows for Null values. Assuming we wanted to add three records with the third having a null record, then we would do so by using the following statement

```
INSERT INTO Test
VALUES
('T001','Training Program 1','Monthly'),
('T002','Training Program 2','Weekly'),
('T003','Training Program 3','');
```

The empty double quotes in the last statement would insert a blank record. The output is shown.

```
mysql> SELECT * FROM Test;
+---------+--------------------+---------+
| Test_No | Name               | Type    |
+---------+--------------------+---------+
| T001    | Training Program 1 | Monthly |
| T002    | Training Program 2 | Weekly  |
| T003    | Training Program 3 |         |
+---------+--------------------+---------+
3 rows in set (0.00 sec)

mysql>
```

UPDATE TABLE

The update command is used to modify data in a table [1]. The UPDATE command uses the following syntax

```
UPDATE tablename
SET fieldname = datavalue, fieldname = datavalue, ...
WHERE conditon_to_be_met;
```

The UPDATE statement can be used to add values to a specific field. Let us assume that Bustani needs to start collecting Employee ID numbers. Now, the employee table does not have a field for collecting ID numbers. We would need to add the column for ID called ID_Numbers. We would use the following statement.

```
ALTER TABLE Employee
ADD ID_Num INT(8) NOT NULL;
```

Output

```
mysql> ALTER TABLE Employee
    -> ADD ID_Num INT(8) NOT NULL;
Query OK, 0 rows affected, 1 warning (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 1

mysql>
mysql> DESCRIBE Employee;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| Emp_Num | char(6)     | NO   | PRI | NULL    |       |
| FName   | varchar(25) | NO   |     | NULL    |       |
| LName   | varchar(25) | NO   |     | NULL    |       |
| Gender  | varchar(10) | YES  |     | NULL    |       |
| DOB     | date        | NO   |     | NULL    |       |
| MStatus | varchar(15) | NO   |     | NULL    |       |
| ID_Num  | int         | NO   |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
7 rows in set (0.00 sec)

mysql>
```

Now to add employee ID numbers to the added field, we would use a single statement for each employee. The statement is as shown

```
UPDATE Employee

SET ID_Num=17584692

WHERE Emp_Num='EMP001';
```

Output

```
mysql> UPDATE Employee
    -> SET ID_Num=17584692
    -> WHERE Emp_Num='EMP001';
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

```
mysql> SELECT * FROM Employee;
+---------+-------+--------+--------+------------+---------+----------+
| Emp_Num | FName | LName  | Gender | DOB        | MStatus | ID_Num   |
+---------+-------+--------+--------+------------+---------+----------+
| EMP001  | Agnes | Njue   | Female | 1997-09-12 | Single  | 17584692 |
| EMP002  | Will  | Omondi | Male   | 1989-08-24 | Married |        0 |
| EMP003  | Carol | Njagi  | Female | 1993-09-01 | Single  |        0 |
| EMP004  | Mary  | Munene | Female | 1998-08-06 | Married |        0 |
| EMP005  | Cathy | Resley | Male   | 1989-04-16 | Married |        0 |
| EMP006  | Abdul | Muita  | Male   | 1988-07-30 | Single  |        0 |
| EMP007  | Steve | Kimuri | Male   | 1999-03-02 | Married |        0 |
| EMP008  | Ruth  | Kimuli | Female | 1996-04-07 | Single  |        0 |
+---------+-------+--------+--------+------------+---------+----------+
8 rows in set (0.00 sec)

mysql>
```

As you can see EMP001 has gotten a new ID number. The same can be repeated until all the values are entered into the table. One could also update multiple columns. For instance, EMP003 got Married and is also updating their ID number. The statement used would be as follows:

```
UPDATE Employee

SET ID_Num= 12346598, MStatus="Married"

WHERE Emp_Num='EMP003';
```

Output

```
mysql> SELECT * FROM Employee;
+---------+-------+--------+--------+------------+---------+----------+
| Emp_Num | FName | LName  | Gender | DOB        | MStatus | ID_Num   |
+---------+-------+--------+--------+------------+---------+----------+
| EMP001  | Agnes | Njue   | Female | 1997-09-12 | Single  | 17584692 |
| EMP002  | Will  | Omondi | Male   | 1989-08-24 | Married |        0 |
| EMP003  | Carol | Njagi  | Female | 1993-09-01 | Married | 12346598 |
| EMP004  | Mary  | Munene | Female | 1998-08-06 | Married |        0 |
| EMP005  | Cathy | Resley | Male   | 1989-04-16 | Married |        0 |
| EMP006  | Abdul | Muita  | Male   | 1988-07-30 | Single  |        0 |
| EMP007  | Steve | Kimuri | Male   | 1999-03-02 | Married |        0 |
| EMP008  | Ruth  | Kimuli | Female | 1996-04-07 | Single  |        0 |
+---------+-------+--------+--------+------------+---------+----------+
8 rows in set (0.00 sec)

mysql>
```

As shown by the output, the records have been updated.

SUMMARY

During this lecture, we have been able to learn a bit about data manipulation languages. Specifically, we reviewed the INSERT and UPDATE statements. The INSERT statement was used to add values to a table. We also used the INSERT statement to enter empty records. Finally, the UPDATE statement was used to add

records and make changes to various data within columns. During the next lecture, we will deal with advanced data manipulation languages while using the SELECT statement.

DISCUSSION TOPIC

This discussion topic is a continuation of lecture 9 during which we altered tables for the Lands Office. Add at least 10 records per table and ensure to consider where the main relationship exists so that you do not get integrity errors. Try and update some of the records within the tables. Look for guidance from your instructor.

REFERENCES

[1] Database systems: design, implementation, and management, Coronel, C., & Morris, S, Cengage Learning, 2019.

[2] Database Systems: A Practical Approach to Design, Implementation, and Management, Connolly, T., & Begg, C., Pearson, 2015.

[3] Fundamentals of database systems, Elmasri, R., & Navathe, S. B., Pearson Education Limited, 2016.