15.3.1E Creating Table from Existing Table

You can define a table and put data into it without going through the usual data definition process. This can be done by using SELECT statement with CREATE TABLE.

The new table stores the result produced by the SELECT statement. The name of the new table must be unique. Following query illustrates this :

```
CREATE TABLE orditem AS
(    SELECT icode, descp
     FROM items
     WHERE QOH < ROL
);
```

NOTE

Alternatively, you may also write : **SHOW COLUMNS FROM <tablename> ;** to view column details of a table.

This will create a new table called *orditem* that stores two columns : *itemcode* and *description* for the items that have their *QOH* less than *ROL* in the relation ***items***.

The newly created table **Orditem** will look as it is shown below :

Table : *Orditem*

Icode	Descp
I06	Cream Roll

The table *Orditem* has derived its contents from table **items**.

If you do not specify the WHERE clause, *icode* & *descp* from all rows of the *items* relation will be copied into *Orditem table*.

The CREATE TABLE <table> AS SELECT... is useful for creating test tables, new tables as copies of existing tables, and for making smaller tables out of large tables.

Scanned with CamScanner

15.4 Changing Data with DML Commands

As you know that DML (Data Manipulation Language) division of SQL, is dedicated to manipulating data in one way or another. This section is going to talk about various DML commands that are used for changing data in tables. Of these commands, INSERT INTO command is used for entering data into tables.

15.4.1 Inserting Data into Table

Values are placed in and removed from attributes of a relation with three DML commands : INSERT, DELETE and UPDATE. These are all referred to in *SQL* as *update* commands in a generic sense. In our text, lowercase "update" will indicate these commands generically and the uppercase for the keyword UPDATE.

15.4.1A INSERT INTO Command

The rows (tuples) are added to relations using INSERT command of *SQL*. In its simplest form, INSERT takes the following syntax :

SYNTAX

```
INSERT INTO <tablename> [ <column list> ]
VALUES ( <value>, <value> ... ) ;
```

For example, to enter a row into *employee* table (defined earlier), you could use the following statement :

```
INSERT INTO employee
VALUES (1001, 'Ravi', 'M', 'E4', 4670.00) ;
```

See the order of values matches the order of columns in the CREATE TABLE command of *employee*. The same can be done with an alternate command as shown below :

```
INSERT INTO employee (ecode, ename, sex, grade, gross)
VALUES (1001, 'Ravi', 'M', 'E4', 4670.00) ;
```

The INSERT statement adds a new row to *employee* giving a value for every column in the row. Note that the data values are in the same order as the column names in the table. Data can be added only to some columns in a row by specifying the columns and their data.

For instance, if you want to insert only *ecode*, *ename* and *sex* columns, you use the command :

```
INSERT INTO employee (ecode, ename, sex)
VALUES (2014, 'Manju', 'F') ;
```

The columns that are not listed in the INSERT command will have their default value, if it is defined for them, otherwise, NULL value.

If any other column (that does not have a **default value** and is defined NOT NULL) is skipped or omitted, an error message is generated and the row is not added.

NOTE

In an INSERT statement, only those columns can be omitted that have either default value defined or they allow NULL values.

Scanned with CamScanner

EXAMPLE 15.3 *Add the following data in the above Table (Employee) as instructed.*

ID	First_Name	Last_Name	User_ID	Salary
1	Dim	Joseph	Jdim	5000
2	Jagannath	Mishra	jnmishra	4000
3	Siddharth	Mishra	smishra	8000
4	Shankar	Giri	sgiri	7000
5	Gautam	Buddha	bgautam	2000

(*i*) *Populate table with first record mentioning the column list in the insert clause.*
(*ii*) *Populate table with next two records without mentioning the column list in the insert clause.*
(*iii*) *Populate table with 4th record and enter only ID and First_Name.*
(*iv*) *Populate table with 5th recorded and enter ID, User_ID, and Last_Name only.*

Solution.

(*i*)
```
INSERT INTO Employee (ID, First_name, Last_name, User_ID, Salary)
Values (1, 'Dim', 'Joseph', 'Jdim', 5000) ;
```

(*ii*)
```
INSERT INTO Employee
Values (2, 'Jagannath', 'Mishra', 'jnmishra', 4000) ;
INSERT INTO Employee
Values (3, 'Siddarth', 'Mishra', 'smishra', 8000) ;
```

(*iii*)
```
INSERT INTO Employee (ID, First_name)
Values (4, 'Shankar') ;
```

(*iv*)
```
INSERT INTO Employee (ID, User_ID, Last_Name)
Values (5, 'bgautam', 'Buddha') ;
```

As you know that following command is used to insert a new row with values for all columns of the table :

```
INSERT INTO <tablename>
VALUES (<value1>, <value2>.....) ;
```

And the following INSERT INTO command allows you to insert a new row with values of select columns :

```
INSERT INTO <tablename> (<columnname1>, <columnname2>.....)
VALUES (<value1>, <value2>.....) ;
```

Let us now learn how you can insert nulls, date-values and data from other table(s).

15.4.1B Inserting NULL values

To insert value NULL in a specific column, you can type NULL without quotes and NULL will be inserted in that column.

Scanned with CamScanner

Consider the following statement :

```
INSERT INTO EMPL (Empno, Ename, Job, Mgr, Hiredate, Sal, Comm, Deptno)
VALUES (8100, 'YASH', 'ANALYST', NULL, '10-MAY-03', 6000, NULL, 20) ;
```

See, for **Mgr** and **Comm** columns, NULL values have been inserted.

15.4.1C Inserting Dates

Dates are by default entered in 'YYYY-MM-DD' format *i.e.*, first four digits depicting year, followed by a hyphen, followed by 2 digits *month*, followed by a hyphen and a two digit day. All this is enclosed in single quotes. In the above two INSERT INTO statements given in previous section, the dates (HIREDATE) have been given in the same format.

15.4.1D Inserting Data from Another Table

INSERT command can also be used to take or derive values from one table and place them in another by using it with a query. To do this, simply replace the VALUES clause with an appropriate query as shown in the following example :

```
INSERT INTO branch1
SELECT * FROM branch2
WHERE gross > 7000.00 ;
```

This query will generate data to be inserted into table

It will extract all those rows from *branch2* that have *gross* more than 7000.00 and insert this produced result into the table *branch1*. To insert using a query, the following conditions must be true :

(*i*) Both the tables must be already created.

(*ii*) The columns of the tables being inserted into, must match the columns output by the subquery.

EXAMPLE 15.4 *Insert into empl table, the employee numbers, names, and salaries of all those employees of Temp table that have completed 2 years.*

The structure of ***Temp*** *table is as follows :*

Enum	NOT NULL Number(4)
Ename	Varchar(10)
Sal	Number(7,2)
Months	Number(3)

Solution.

```
mysql> INSERT INTO Empl(Empno, Ename, Sal)
       SELECT Enum, Ename, Sal
       FROM Temp
       WHERE Months >= 24 ;
```

NOTE

Issue a command COMMIT ; to make the changes to a table permanent.

Scanned with CamScanner