**Data Manipulation Language (DML) Statements**

Data manipulation language (DML) statements query and manipulate data in existing schema objects. These statements do not implicitly commit the current transaction.

The following are the DML statements available in Oracle.

* INSERT          :Use to Add Rows to existing table.
* UPDATE        :Use to Edit Existing Rows in tables.
* DELETE         :Use to Delete Rows from tables.
* MERGE          :Use to Update or Insert Rows depending on condition.

**Insert**

Use the Insert Statement to Add records to existing Tables.

Examples.

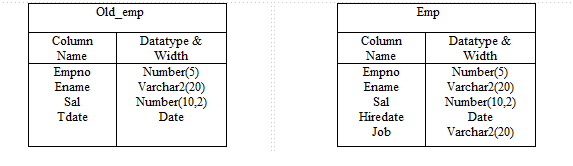
To add a new row to an emp table.

Insert into emp values (101,’Sami’,’G.Manager’,  
             ’8-aug-1998’,2000);

If you want to add a new row by supplying values for some columns not all the columns then you have to mention the name of the columns in insert statements. For example the following statement inserts row in emp table by supplying values for empno, ename, and sal columns only. The Job and Hiredate columns will be null.

Insert into emp (empno,ename,sal) values (102,’Ashi’,5000);

Suppose you want to add rows from one table to another i.e. suppose we have Old\_Emp table and emp table with the following structure



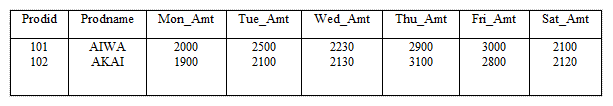
Now we want to add rows from old\_emp table to emp table. Then you can give the following insert statement

Insert into emp (empno, ename, sal)  
      select empno, ename, sal from old\_emp;

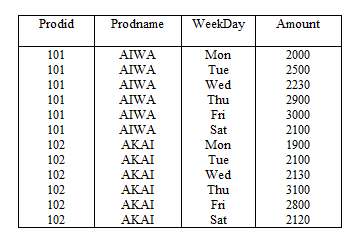
**Multitable Insert**

Suppose we have sales table with the following structure.

Sales



Now we want to add the rows from SALES table  Weekly\_Sales Table in the following Structure.



To achieve the above we can give a multi table INSERT statement given below

Insert all  
     Into week\_sales(prodid,prodname,weekday,amount)  
     Values (prodid,prodname,’Mon’,mon\_amt)  
     Into week\_sales(prodid,prodname,weekday,amount)  
     Values (prodid,prodname,’Tue’,tue\_amt)  
     Into week\_sales(prodid,prodname,weekday,amount)  
     Values (prodid,prodname,’Wed’,wed\_amt)  
     Into week\_sales(prodid,prodname,weekday,amount)  
     Values (prodid,prodname,’Thu’,thu\_amt)  
     Into week\_sales(prodid,prodname,weekday,amount)  
     Values (prodid,prodname,’Fri’,fri\_amt)  
     Into week\_sales(prodid,prodname,weekday,amount)  
     Values (prodid,prodname,’Sat’,sat\_amt)  
        Select prodid,prodname,mon\_amt,tue\_amt,wed\_amt,thu\_amt  
            Fri\_amt,sat\_amt from sales;

**Update**

Update statement is used to  update rows in existing tables which is in your own schema or if you have update privilege on them.

For example to raise the salary by Rs.500 of employee number 104. You can give the following statement.

update emp set sal=sal+500 where empno = 104;

In the above statement if we did not give the where condition then all employees salary will be raised by Rs. 500. That’s why always specify proper WHERE condition if don’t want to update all employees.

For example We want to change the name of employee no 102 from ‘Sami’ to ‘Mohd Sami’ and to raise the salary by 10%. Then the statement will be.

update emp set name=’Mohd Sami’,  
  sal=sal+(sal\*10/100) where empno=102;

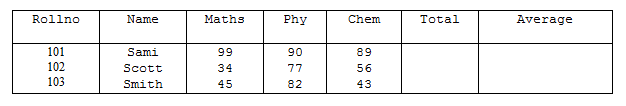
Now we want to raise the salary of all employees by 5%.

update emp set sal=sal+(sal\*5/100);

Now to change the names of all employees to uppercase.

update emp set name=upper(name);

Suppose We have a student table with the following structure.



Now to compute total which is sum of Maths,Phy and Chem and average.

update student set total=maths+phy+chem,  
    average=(maths+phy+chem)/3;

Using Sub Query in the Update Set Clause.

Suppose we added the city column in the employee table and now we want to set this column with corresponding city column in department table which is join to employee table on deptno.

update emp set city=(select city from dept  
                          where deptno= emp.deptno);

**Delete**

Use the DELETE statement to delete the rows from existing tables which are in your schema or if you have DELETE privilege on them.

For example to delete the employee whose empno is 102.

delete from emp where empno=102;

If you don’t mention the WHERE condition then all rows will be deleted.

Suppose we want to delete all employees whose salary is above 2000. Then give the following DELETE statement.

delete from emp where salary > 2000;

The following statement has the same effect as the preceding example, but uses a subquery:

DELETE FROM (SELECT \* FROM emp)  
    WHERE sal > 2000;

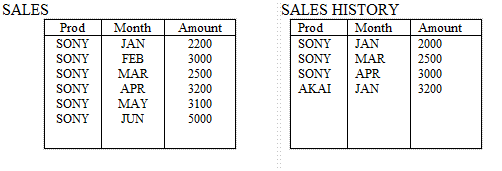
To delete all rows from emp table.

delete from emp;

**Merge**

Use the MERGE statement to select rows from one table for update or insertion into another table. The decision whether to update or insert into the target table is based on a condition in the ON clause. It is a new feature of Oracle Ver. 9i. It is also known as UPSERT i.e. combination of UPDATE and INSERT.

For example suppose we are having sales and sales\_history table with the following structure.



Now we want to update sales\_history table from sales table i.e. those rows which are already present in sales\_history, their amount should be updated and those rows which are not present in sales\_history table should be inserted.

merge into sales\_history sh  
   using sales s  
   on (s.prod=sh.prod and s.month=sh.month)  
    when matched then update set sh.amount=s.amount  
     when not matched then insert values (prod,month,amount);

After the statement is executed sales\_history table will look like this.

