FOREIGN KEY CONSTRAINT

A foreign key is a kind of constraint which establishes a relationship among tables. A foreign key may be a single column or the combination of columns which derive their values based on the primary key or unique key values from another table. A foreign key constraint is also known as the referential integrity constraint, as its values correspond to the actual values of the primary key of other table.

A table in which there is a foreign key, it is called as the detail table or the child table and the table from which it refers the values of the primary key, it is called as the master table or the parent table. A foreign key must have the corresponding primary key or unique key value in the master table. Whenever you insert a value in the foreign key column, it checks the value from the primary key of the parent table. If the value is not present, it gives an error. The parent table can be referenced in the foreign key by using the References option.

Consider two tables (EMP, DEPT) given below.

Target Attribute

EMP

Empno	Ename	Job	Sal	Deptno
1	A	Clerk	4000	10
2	A	Clerk	4000	30
3	B	Mgr	8000	20
4	C	Peon	2000	40
5	D	Clerk	4000	10
6	E	Mgr	8000	50

DEPT

Deptno	Dname	Loc
10	A	Asr
20	B	Jal
30	C	Qadian
40	D	-

If we try to insert information of employee with deptno 50, then this is an invalid information, because there is no deptno 50 exists in the company(as shown in table DEPT).Then, this invalid information should be prevented from insertion, which would only be possible if deptno of EMP table refer the deptno of DEPT table. It means that only those values are permitted in deptno of EMP table, which appears in the deptno attribute of the DEPT table. Thus, we can say that

deptno of EMP table is the foreign key which refers the primary key deptno of DEPT table. Thus, we can insert the empno 6 with any deptno from 10,20,30 and 40.

Null may also be permitted in the deptno of EMP table. Here, deptno of DEPT table is Target attribute and DEPT is the target table.

Consider another example:

Student

Rno	Name	Class_Code
1	A	2
2	B	1
3	C	-

Class

Class_Code	Name
1	B.TECH
2	B.TECH
3	BBA

Here, college has three valid classes with class code 1,2 and 3. The class_code(foreign key) of student table refers class_code of class table.

The syntax of the foreign key at the column level is

```
Columnname datatype (size) references tablename [(columnname)]
```

Example

Create a table emp_detail in which the deptno field refers its value from the deptno field of the dept table.

```
SQL> CREATE table emp_detail
  (Empno number (4) primary key,
    Ename varchar2 (20) not null,
    Hiredate date not null,
    Deptno number (2) references dept (deptno) ,
    Salary number (6, 2));
```

So, in the emp_detail table deptno is a foreign key, which gets its values from the parent table i.e. the dept table.

The syntax of the foreign key at the table level is

```
Foreign key (columnname [,columnname……..]) references tablename (columnname [,columnname……. ])
```

Example

- Create a table emp_detail in which the deptno and dname together refers its value from deptno and dname field of dept table.

```
SQL> CREATE table emp_detail
      (Empno number (4) primary key,
       Ename varchar2 (20) not null,
       Hiredate date not null,
       Deptno number (2),
       Dname varchar2 (10),
       Salary number (6, 2)
       Foreign key (deptno,dname) references dept(deptno,dname)) ;
```

So in this table deptno and dname is the foreign key which references the corresponding fields of the dept table.

Important Note:

It is important to consider what will happen to child record if parent record is updated or deleted.

For example, let us consider the following database:

Emp (eno, ename, job, dno) Having eno as primary key and dno as forign key referencing dept (dno).

Dept (dno, dname) having dno as primary key.

Here, dept table is parent as shown below:

EMP					DEPT	
ENO	ENAME	JOB	DNO		DNO	DNAME
1	RAJ	CLERK	10		10	COMPUTER
2	RAM	PROF	20		20	CIVIL
3	RAHAT	PROF	10			
4	RISHAN	ASSOC PROF	20			

Now, it is important to consider what will happen, if user tries to delete record of deptno 20 from DEPT table. Then of course record of emp no 2 and 4 become invalid as they belong to this department.

There are four options to handle this situation.

- Prevent the dept being deleted until all its employees are re allotted to some other dept
- Automatic delete the employees who belong to that dept
- Set the dept column for the employees who belong to that dept to NULL
- Set the dept column of these employees to some default value which indicate they are currently not allotted to any dept

To achieve this, four options van be set in CREATE TABLE command

ON DEELTE RESTRICT

ON DELETE CASCADE

ON DELETE SET NULL

ON DELETE SET DEFAULT

The syntax to each of these cases has been given below:

```
SQL> CREATE TABLE DEPT(DNO NUMBER(2) PRIMARY KEY,
DNAME CHAR(20));
```

Case-I

```
SQL> CREATE TABLE EMP(ENO NUMBER(2) PRIMARY KEY,
ENAME CHAR(20), JOB CHAR(20)
DNO NUMBER(2) REFERENCES DEPT(DNO) ON DELETE RESTRICT;
```

It will prevent the dept being deleted until all its employees are re allotted to some other dept. This is the default case and will be applicable if did not give any option. It means the below command will also has the same effect.

```
SQL> CREATE TABLE EMP(ENO NUMBER(2) PRIMARY KEY,
ENAME CHAR(20), JOB CHAR(20)
DNO NUMBER(2) REFERENCES DEPT(DNO);
```

Case-II

```
SQL> CREATE TABLE EMP(ENO NUMBER(2) PRIMARY KEY,
ENAME CHAR(20), JOB CHAR(20)
DNO NUMBER(2) REFERENCES DEPT(DNO) ON DELETE CASCADE;
```

It will also delete the employees who belong to that dept.

Case-III

```
SQL> CREATE TABLE EMP(ENO NUMBER(2) PRIMARY KEY,
ENAME CHAR(20), JOB CHAR(20)
DNO NUMBER(2) REFERENCES DEPT(DNO) ON DELETE SET NULL;
```

It will set the dept column for the employees who belong to that dept to NULL.

Case-IV

```
SQL> CREATE TABLE EMP(ENO NUMBER(2) PRIMARY KEY,
ENAME CHAR(20), JOB CHAR(20)
DNO NUMBER(2) REFERENCES DEPT(DNO) ON DELETE SET DEFAULT;
```

It will set the dept column of these employees to some default value which indicate they are currently not allotted to any dept. This concept is not implemented in Oracle.