Database System

Lecture 7

SQL Sub Languages

DDL - Data Definition Language

TABLE CONSTRAINT

Foreign Key constraint

Prepared By

Dhafer Sabah Yaseen

SQL Components Or SQL Sub Languages

DCL: Data Control Language
Example: Grant, Revoke.

DDL: Data Definition Language.
Example: Create, Alter, Drop, Rename and Truncate.

DML: Data Manipulation Language
Example: Insert, Update, Delete

DRL: Data Retrieval Language
Example: Select

TCL: Transaction Control Language
Example : Rollback, Commit, Savepoint

DDL - TABLE CONSTRAINT

Types of constraint

- Primary Key Constraint.
- Foreign Key constraint.
- Unique constraint.
- Check Constraint.
- Not NULL Constraint.
- Default Constraint.

SQL Components Or SQL Sub Languages

After this lecture you will be able to understand :

What is Foreign KEY CONSTRAINT?
How to add Foreign key constraints?
- Using a CREATE TABLE statement .
- Using a ALTER TABLE statement.
- What is ERD?
- What is a foreign key with Cascade DELETE in Oracle?
- What is a foreign key with "Set NULL on Delete" in Oracle?
- How to Enable and Disable a Foreign key constraints?
- How to Drop a Foreign key constraints?

DDL - TABLE CONSTRAINT

- ***Foreign Key Constraint***

A foreign key is a way to enforce referential integrity within your Oracle database. A foreign key means that values in one table must also appear in another table.

The referenced table is called the *parent table* while the table with the foreign key is called the *child table*.

The foreign key in the child table will generally reference a primary key in the parent table

DDL - TABLE CONSTRAINT

The syntax for creating a foreign key Using a CREATE TABLE statement is:

```
CREATE TABLE table_name
(
column1 datatype null/not null,
column2 datatype null/not null,
...
CONSTRAINT fk_column     FOREIGN KEY (column1,
column2, ... column_n)
REFERENCES parent_table (column1, column2, ...
column_n)
ON DELETE CASCADE
);
```

DDL - TABLE CONSTRAINT

Example:

First we must create parant table .

```
CREATE TABLE supplier
(
supplier_id number(10) not null,
supplier_name varchar2(50) not null,
 contact_name varchar2(50),
CONSTRAINT supplier_pk PRIMARY KEY (supplier_id)
);
```

→ → →

DDL - TABLE CONSTRAINT

Second we must create child table :

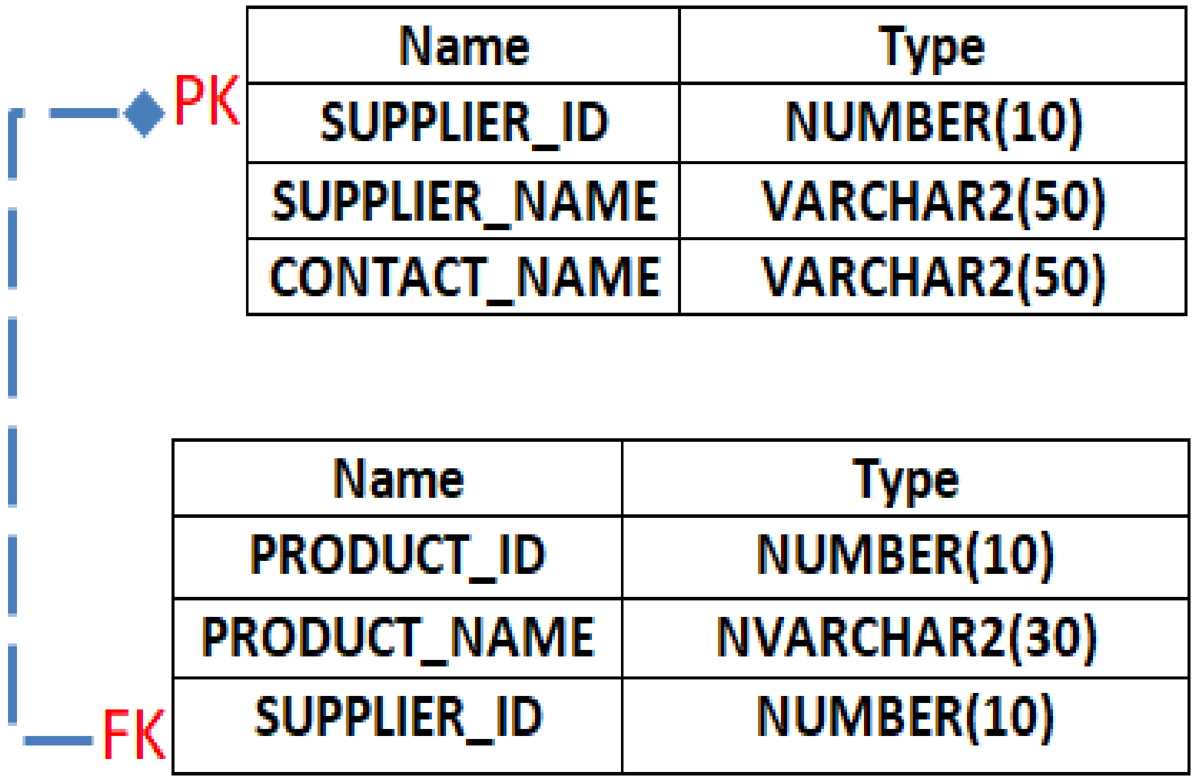
```
CREATE TABLE products
(
product_id number(10) not null,
 product_name Nvarchar2(30) not null,
supplier_id number(10) not null,
CONSTRAINT fk_supplier    FOREIGN KEY (supplier_id)
REFERENCES supplier(supplier_id)
 ON DELETE CASCADE
 );
```

→ → →

DDL - TABLE CONSTRAINT

In this example,
we've created a primary key on the ***supplier*** table called ***supplier_pk***.
It consists of only one field - the ***supplier_id*** field.
Then
we've created a foreign key called ***fk_supplier*** on the ***products*** table that references the ***supplier*** table based on the ***supplier_id*** field.

DDL - TABLE CONSTRAINT



Name	Type
SUPPLIER_ID	NUMBER(10)
SUPPLIER_NAME	VARCHAR2(50)
CONTACT_NAME	VARCHAR2(50)

Name	Type
PRODUCT_ID	NUMBER(10)
PRODUCT_NAME	NVARCHAR2(30)
SUPPLIER_ID	NUMBER(10)

Entity Relationship Diagram (ERD)

DDL - TABLE CONSTRAINT

Using an ALTER TABLE statement
The syntax for creating a foreign key in an ALTER TABLE statement is:

ALTER TABLE table_name

ADD CONSTRAINT constraint_name

FOREIGN KEY (column1, column2, ... column_n)
REFERENCES parent_table (column1, column2, ... column_n)

ON DELETE CASCADE;

DDL - TABLE CONSTRAINT

Using an ALTER TABLE statement

Example:

```
ALTER TABLE products
ADD CONSTRAINT  fk_supplier  FOREIGN KEY (supplier_id)
REFERENCES supplier(supplier_id)
ON DELETE CASCADE;
```

In this example,
we've created a foreign key called ***fk_supplier*** that references the **supplier** table based on the **supplier_id** field.

DDL - TABLE CONSTRAINT

We could also create a foreign key with more than one field as in the example below:

Using an Create TABLE statement

```
CREATE TABLE supplier2
(
supplier_id number(10) not null,
supplier_name varchar2(50) not null,
 contact_name varchar2(50),
CONSTRAINT supplier2_pk PRIMARY KEY (supplier_id,
supplier_name)
);
```

→ → →

DDL - TABLE CONSTRAINT

Using an Create TABLE statement

```
CREATE TABLE products2
(
 product_id number(10) not null,
 product_name Nvarchar2(30) not null,
 supplier_id number(10) not null,
supplier_name varchar2(50) not null,
          CONSTRAINT  fk_supplier2    FOREIGN KEY (supplier_id,
          supplier_name)
          REFERENCES supplier2(supplier_id, supplier_name)
          ON DELETE CASCADE
 );
```

In this example,
our foreign key called **fk_supplier2** references the **supplier** table based on two fields - the **supplier_id** and **supplier_name** fields.

DDL - TABLE CONSTRAINT

Using an ALTER TABLE statement

```
CREATE TABLE supplier3
(
supplier_id number(10) not null,
supplier_name varchar2(50) not null,
 contact_name varchar2(50)
);

ALTER TABLE supplier3
ADD CONSTRAINT supplier3_pk PRIMARY KEY (supplier_id,
supplier_name);
```

→ → →

DDL - TABLE CONSTRAINT

Using an ALTER TABLE statement

```
CREATE TABLE products3
(
 product_id number(10) not null,
 product_name Nvarchar2(30) not null,
 supplier_id number(10) not null,
supplier_name varchar2(50) not null
);

ALTER TABLE products3
ADD CONSTRAINT  fk_supplier3   FOREIGN KEY (supplier_id,
           supplier_name)
REFERENCES supplier3 (supplier_id, supplier_name)
           ON DELETE CASCADE    ;
```

DDL - TABLE CONSTRAINT

What is a foreign key with Cascade DELETE in Oracle?

A foreign key with cascade delete means that if a record in the parent table is deleted, then the corresponding records in the child table will automatically be deleted. This is called a cascade delete in Oracle.
A foreign key with a cascade delete can be defined in either a CREATE TABLE statement or an ALTER TABLE statement.

What is a foreign key with "Set NULL on Delete" in Oracle?
A foreign key with "set null on delete" means that if a record in the parent table is deleted, then the corresponding records in the child table will have the foreign key fields set to null. The records in the child table will **not** be deleted.

DDL - TABLE CONSTRAINT

A foreign key with a "set null on delete" can be defined in either a CREATE TABLE statement or an ALTER TABLE statement.

Using a CREATE TABLE statement

The syntax for creating a foreign key using a CREATE TABLE statement is:

CREATE TABLE table_name
(
column1 datatype null/not null, column2 datatype null/not null, ...   CONSTRAINT fk_column   FOREIGN KEY (column1, column2, ... column_n)   REFERENCES parent_table (column1, column2, ... column_n)
ON DELETE SET NULL
);

DDL - TABLE CONSTRAINT

Using an ALTER TABLE statement

The syntax for creating a foreign key in an ALTER TABLE statement is:

```
ALTER TABLE table_name
ADD CONSTRAINT constraint_name
  FOREIGN KEY (column1, column2, ... column_n)
 REFERENCES parent_table (column1, column2, ... column_n)
ON DELETE SET NULL;
```

DDL - TABLE CONSTRAINT

Disable/Enable a foreign key

Once you have created a foreign key in Oracle, you may encounter a situation where you are required to disable the foreign key. You can do this using the ALTER TABLE statement in Oracle.

The syntax to disable a foreign key in Oracle/PLSQL is:

ALTER TABLE table_name
DISABLE/Enable
CONSTRAINT constraint_name;

Example:
ALTER TABLE products2 DISABLE CONSTRAINT fk_supplier2;
ALTER TABLE products2 ENABLE CONSTRAINT fk_supplier2;

DDL - TABLE CONSTRAINT

Drop a Foreign Key

Once a foreign key has been created, you may find that you wish to drop the foreign key from the table. You can do this with the ALTER TABLE statement in Oracle.

The syntax to drop a foreign key in Oracle/PLSQL is:

```
ALTER TABLE table_name
DROP CONSTRAINT constraint_name;
```

Example

```
ALTER TABLE products2
DROP CONSTRAINT fk_supplier2;
```

?
?
Q
&
&
A
!
!

Dhafer Sabah Yaseen