Created by Ahsan Arif

Foreign Keys with cascade delete

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.

A foreign key with a cascade delete means that if a record in the parent table is deleted, then the corresponding records in the child table with automatically be deleted. This is called a cascade delete.

A foreign key with a cascade 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 CASCADE
);
```

For example:

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

CREATE TABLE products
( product_id      numeric(10)  not null,
```

```
  supplier_id      numeric(10)  not null,
  CONSTRAINT fk_supplier
    FOREIGN KEY (supplier_id)
   REFERENCES supplier(supplier_id)
   ON DELETE CASCADE
);
```

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.

Because of the cascade delete, when a record in the supplier table is deleted, all records in the products table will also be deleted that have the same supplier_id value.

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

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

CREATE TABLE products
( product_id       numeric(10)  not null,
  supplier_id      numeric(10)  not null,
  supplier_name varchar2(50)  not null,
  CONSTRAINT fk_supplier_comp
    FOREIGN KEY (supplier_id, supplier_name)
   REFERENCES supplier(supplier_id, supplier_name)
   ON DELETE CASCADE
);
```

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

The cascade delete on the foreign key called *fk_foreign_comp* causes all corresponding records in the products table to be cascade deleted when a record in the supplier table is deleted, based on supplier_id and supplier_name.

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;
```

For 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 (with a cascade delete) called *fk_supplier* that references the supplier table based on the supplier_id field.

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

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