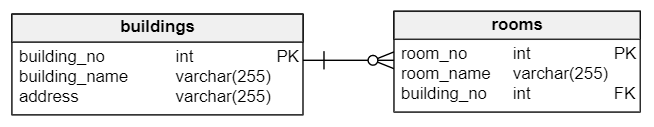
**ON DELETE CASCADE: Deleting Data from Multiple Related Tables**

## ON DELETE CASCADE example

Suppose we have two tables: buildings and rooms. In this database model, each building has one or more rooms. However, each room belongs to one only one building. A room would not exist without a building.

The relationship between the buildings and rooms tables is one-to-many (1:N) as illustrated in the following database diagram:



When we delete a row from the buildings table, we also want to delete the rows in the rooms table that references to the rows in the buildings table. For example, when we delete a row with building no. 2 in the buildings table as the following query:

DELETE FROM buildings

WHERE

    building\_no = 2;

We want the rows in the rooms table that refers to building number 2 will be also removed.

The following are steps that demonstrate how MySQL ON DELETE CASCADE referential action works.

**Step 1**. Create the buildings table:

CREATE TABLE buildings (

    building\_no INT PRIMARY KEY AUTO\_INCREMENT,

    building\_name VARCHAR (255) NOT NULL,

    address VARCHAR (255) NOT NULL

);

**Step 2**. Create the rooms table:

CREATE TABLE rooms (

    room\_no SERIAL PRIMARY KEY,

    room\_name VARCHAR (255) NOT NULL,

    building\_no INT NOT NULL,

    FOREIGN KEY (building\_no) REFERENCES buildings (building\_no) ON DELETE CASCADE);

*Note that we add the ON DELETE CASCADE clause at the end of the foreign key constraint definition.*

**Step 3**. Insert data into the buildings table:

INSERT INTO buildings(building\_name,address)

VALUES('ACME Headquaters','3950 North 1st Street CA 95134'),

      ('ACME Sales','5000 North 1st Street CA 95134');

**Step 4**. Query data from the buildings table:

SELECT \* FROM buildings;

MySQL ON DELETE CASCADE buildings table

We have two rows in the buildings table.

**Step 5**. Insert data into the rooms table:

INSERT INTO rooms(room\_name,building\_no)

VALUES('Amazon',1),

      ('War Room',1),

      ('Office of CEO',1),

      ('Marketing',2),

      ('Showroom',2);

**Step 6**. Query data from the rooms table:

SELECT \* FROM rooms;



We have 3 rooms that belong to building 1 and 2 rooms that belong to the building 2.

**Step 7**. Delete the building with building no. 2:

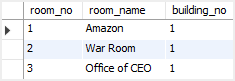
DELETE FROM buildings

WHERE

    building\_no = 2;

**Step 8**. Query data from rooms table:

SELECT \* FROM rooms;



Can see, all the rows that reference to building\_no 2 were deleted.

TRICK: to find tables that associated with the buildings table with the CASCADE  deletion rule  in a database, you use the following query:

USE information\_schema;

SELECT

    table\_name

FROM

    referential\_constraints

WHERE

    constraint\_schema = 'classicmodels'

        AND referenced\_table\_name = 'buildings'

        AND delete\_rule = 'CASCADE'

MySQL ON DELETE CASCADE tips

## 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.

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

CREATE TABLE supplier

( supplier\_id numeric(10) not null,

supplier\_name varchar(50) not null,

contact\_name varchar(50),

CONSTRAINT supplier\_pk PRIMARY KEY (supplier\_id)

);

CREATE TABLE products

( product\_id numeric(10) not null,

supplier\_id numeric(10),

CONSTRAINT fk\_supplier

FOREIGN KEY (supplier\_id)

REFERENCES supplier(supplier\_id)

ON DELETE SET NULL

);

## Using an ALTER TABLE statement

ALTER TABLE products

ADD CONSTRAINT fk\_supplier

FOREIGN KEY (supplier\_id)

REFERENCES supplier(supplier\_id)

ON DELETE SET NULL;

# TRUNCATE TABLE

The TRUNCATE TABLE command deletes the data inside a table, but not the table itself.

The following SQL truncates the table "Categories":

TRUNCATE TABLE Categories;

TRUNCATE TABLE is similar to the DELETE statement with no WHERE clause; however, TRUNCATE TABLE is faster and uses fewer system and transaction log resources.