Lab sheet-2: What is Referential integrity? Create two TABLES

Employees (Id, Name, DepartmentID) where Id is primary Key and DepartmentID is foreign key
Department(Id, Dname) where ID is primary key

Referential Integrity

Note: The point that you need to remember is, the Delete and Update rules were not imposed on the master table, they are imposed on the child table, and that too on the foreign key column.

How to add referential Integrity?
ALTER TABLE Employees ADD FOREIGN KEY(DepartmentID) REFERENCES DEPARTMENT(Id)
ON DELETE CASCADE
ON UPDATE CASCADE;

1. Example: CASCADE Referential Integrity Constraints in MySQL

We have set the ON DELETE and ON UPDATE rules as CASCADE. This means, if we delete a record from the Department table for which there are some records in the Employees table, then those records will also be deleted. Similarly, if we update a record in the Department table for which if there are some records in the Employees table, then those records will be updated with the updated primary key value.

Step1: Create two Tables **Employees** and **Department**

Employees (Id, Name, DepartmentID) where Id is primary Key and DepartmentID is foreign key
Department(Id, Dname) where ID is primary key

Step2: Alter Employees table to create foreign key with referential integrity

Example as

ALTER TABLE **Employees** ADD FOREIGN KEY(DepartmentID) REFERENCES **Department** (Id)
ON DELETE CASCADE
ON UPDATE CASCADE;

Step-3: Now, insert some data into the **Employees** and **Department** tables by executing the below SQL statement.

INSERT into Employees VALUES (101, 'Anurag', 10);
INSERT into Employees VALUES (102, 'Pranaya', 20);
INSERT into Employees VALUES (103, 'Hina', 30);

Insert into Department values (10, 'IT');
Insert into Department values (20, 'HR');
Insert into Department values (30, 'INFRA');

Step-4: Now, delete the Department whose Id is 10 from the Department table by executing the below SQL Statement.

DELETE FROM Department WHERE Id = 10;

Now, you can see the above DELETE statement executed successfully, and further if you notice the employees whose DepartmentId is 10 are also deleted from the Employees table automatically.

Step-5: Now, execute the below UPDATE statement.

UPDATE Department SET Id = 40 WHERE Id = 30;

Now, you can see the above UPDATE statement also executed successfully, and further if you notice the employees whose DepartmentId is 30 are also updated as 40 in the Employees table automatically.

2. Example: SET NULL Referential Integrity Constraints in MySQL

Let's delete the existing Employees table and again create the Employees table as shown below. As you can see in the below SQL Script, we have set the ON DELETE and ON UPDATE rules as SET NULL. This means if we delete a record from the Department table for which if there are some records in the Employees table, then those records will also be set as NULL values. Similarly, if we update a record in the Department table for which if there are some records in the Employees table, then those records will be updated as NULL.

Step-1: First, delete the Employees table by executing the below SQL Statement.

DROP TABLE Employees;

Step-2: Then truncate the Department table and add the three records by executing the below SQL Statement.

TRUNCATE TABLE Department;
Insert into Department values (10, 'IT');
Insert into Department values (20, 'HR');
Insert into Department values (30, 'INFRA');

Step-3: Now, create the Employees table by executing the below SQL Statement.

CREATE TABLE Employees(

Id **INT PRIMARY KEY**,
Name **VARCHAR**(100) NOT **NULL**,
DepartmentID **INT**,
FOREIGN KEY (DepartmentId) **REFERENCES** Department(Id)
ON DELETE SET NULL
ON UPDATE SET NULL);

Step-4: Now, insert the following records into the Employees table by executing the below INSERT Statements.

INSERT into Employees **VALUES** (101, 'Anurag', 10);
INSERT into Employees **VALUES** (102, 'Pranaya', 20);
INSERT into Employees **VALUES** (103, 'Hina', 30);

Step-4: Now, delete the Department whose Id is 10 from the Department table by executing the below SQL Statement.

DELETE FROM Department WHERE Id = 10;

Now, you can see the above DELETE statement executed successfully, and further if you notice the Employees table, those employees whose DepartmentId is 10 are set to NULL automatically.

Step-5: Now, execute the below UPDATE statement.

UPDATE Department SET Id = 40 WHERE Id = 30;

Now, you can see the above UPDATE statement also executed successfully, and further if you notice the employees whose DepartmentId is 30 are also updated as NULL in the Employees table automatically.

3. Example: NO ACTION Referential Integrity Constraints in MySQL

Let's delete the existing Employees table and again create the Employees table as shown below. As you can see in the below SQL Script, we have set the ON DELETE and ON UPDATE rules as NO ACTION. NO ACTION is the default action that MySQL performs. It specifies that if an update or deletes statement affects rows in foreign key tables, then the action will be denied and rolled back and an error message will be raised.

Let us understand this with an example.

Step-1: First, delete the Employees table by executing the below SQL Statement.

DROP TABLE Employees;

Step-2: Then truncate the Department table and add the three records by executing the below SQL Statement.

TRUNCATE TABLE Department;

Insert into Department values (10, 'IT');
Insert into Department values (20, 'HR');
Insert into Department values (30, 'INFRA');

Step-3: Now, create the Employees table by executing the below SQL Statement.

CREATE TABLE Employees(
Id **INT PRIMARY KEY**,
Name **VARCHAR**(100) NOT **NULL**,
DepartmentID **INT**,
FOREIGN KEY (DepartmentId) **REFERENCES** Department(Id)
ON DELETE NO ACTION
ON UPDATE NO ACTION);

Step-4: Now, insert the following records into the Employees table by executing the below INSERT Statements.

INSERT into Employees **VALUES** (101, 'Anurag', 10);
INSERT into Employees **VALUES** (102, 'Pranaya', 20);
INSERT into Employees **VALUES** (103, 'Hina', 30);

Step-5: Now, delete the Department whose Id is 10 from the Department table by executing the below SQL Statement.

DELETE FROM Department WHERE Id = 10;

Now, when you try to execute the above DELETE statement, you will get the following error message, and the delete operation is rollback.

Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails (`employeedb`.`employees`, CONSTRAINT `employees_ibfk_1` FOREIGN KEY (`DepartmentID`) REFERENCES `department` (`Id`))

Step-6: Now, execute the below UPDATE statement.

UPDATE Department SET Id = 40 WHERE Id = 30;

Now, when you try to execute the above UPDATE statement then you will get the following error message and the update operation is rollback.
Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails (`employeedb`.`employees`, CONSTRAINT `employees_ibfk_1` FOREIGN KEY (`DepartmentID`) REFERENCES `department` (`Id`))

What is Self-Referential Integrity Constraint?

This is the same as the referential integrity that we have just learned. In earlier cases, we are binding one column of a table (foreign key table) with another column of another table (Primary Key) whereas in self-referential integrity we bind a column of a table with another column of the same table i.e. both the foreign key and primary key will be present in one table only.