https://www.w3schools.com/sql/sql_in.asp

http://localhost/phpmyadmin/

Table of Contents

Figure 5.6
One possible database state for the COMPANY relational database schema.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

1. login to the database server

```
>mysql -u root -p
Enter password: root
mysql>
```

2. Create Database

Syntax:

```
CREATE DATABASE <databasename>;
```

Example:

CREATE DATABASE COMPANY;

3. Show Databases

mysql> **SHOW DATABASES;**

SHOW SCHEMAS;

4. Use databases

Syntax: Use <databasename>;

USE COMPANY;

5. Delete database

Syntax: DROP DATABASE <name>;

DROP DATABASE COMPANY;

6. Create Tables

```
CREATE TABLE EMPLOYEE(
Ssn INT NOT NULL,
Fname varchar(25),
Mnit varchar(25),
Lname varchar(25),
Bdate date,
Address varchar(255),
Sex  char(1),
Salary decimal(6,2),
Super_ssn int,
Dno INT,
PRIMARY KEY (Ssn));
```

```
--------------------------------------

CREATE TABLE DEPARTMENT (
Dname VARCHAR(10) NOT NULL,
Dnumber INT NOT NULL,
Mgr_ssn INT,
Mgr_start_date date,
PRIMARY KEY (Dnumber));
-----------------------------

CREATE TABLE PROJECT (
Pname  VARCHAR(10) NOT NULL,
Pnumber         INTEGER         NOT NULL,
Plocation CHAR(9),
Dnum   INTEGER          NOT NULL,
PRIMARY KEY (Pnumber);
---------------------------

CREATE TABLE DEPENDENT (
Essn      int NOT NULL,
Dependent_name          varchar(25),
Sex CHAR(1),
Bdate date,
Relationship VARCHAR(25),
PRIMARY KEY (Essn,Dependent_name);
-------------------------

CREATE TABLE DEPT_LOCATIONS (
Dnumber INT NOT NULL,
Dlocation CHAR(9),
PRIMARY KEY (Dnumber);
----------------------------

CREATE TABLE WORKS_ON  (
Essn      INT NOT NULL,
Pno INT NOT NULL,
Hours INT,
PRIMARY KEY (Essn,Pno);
----------------------------------
```

- Adding the Foreign Key Constraint

```
ALTER TABLE EMPLOYEE ADD FOREIGN KEY (Dno) REFERENCES DEPARTMENT (Dnumber);
```

Or

```
ALTER TABLE EMPLOYEE ADD FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber)
ON DELETE CASCADE
ON UPDATE CASCADE;
```

7. Describe the Tables (List its columns' definition)

describe EMPLOYEE;

----

8. Delete Table(Relation)

The DROP TABLE command deletes a table in the database.

DROP TABLE <Table name>;

9. TRUNCATE TABLE

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

Example

TRUNCATE TABLE EMPLOYEE;

10. ALTER TABLE (Used to add an attribute to one of the base relations)

Syntax:

ALTER TABLE table_name
ADD column_name datatype;
Example:

ALTER TABLE EMPLOYEE ADD JOB VARCHAR(12);

11. ALTER TABLE - DROP COLUMN

Syntax:

ALTER TABLE table_name
DROP COLUMN column_name;

12. ALTER TABLE - ALTER/MODIFY COLUMN

To change the data type of a column in a table, use the following syntax:

ALTER TABLE table_name
ALTER COLUMN column_name datatype;
Or

```
ALTER TABLE table_name
MODIFY COLUMN column_name datatype;
```

13. INSERT tuples in Relations

```
INSERT INTO EMPLOYEE VALUES ('Richard','K','Marini', '653298653', '30-DEC-52','98 Oak Forest,Katy,TX',
'M', 37000,'987654321', 4 );
```

```
INSERT INTO EMPLOYEE (Fname, Lname, Ssn)

        VALUES ('Richard', 'Marini', '653298653');
```

14. SQL Insert Multiple Rows

```
CREATE TABLE customer (
  first_name VARCHAR(100),
  last_name VARCHAR(100)
);
```

Now, we can INSERT multiple rows in SQL by repeating the list of values inside the brackets:

```
INSERT INTO customer (first_name, last_name) VALUES
('Kristen', 'Rowley'),
('Jed', 'Tomlinson'),
('Margie', 'Escobar'),
('Harriette', 'Mejia'),
('Francis', 'Little');
```

15. DELETE tuples from Relation

Delete Tuples from Relation:

```
        DELETE FROM table
        WHERE condition;
```

```
DELETE FROM EMPLOYEE WHERE Lname='Brown’;
```

```
DELETE FROM EMPLOYEE WHERE Ssn='123456789’;
```

```
DELETE FROM EMPLOYEE WHERE Dno IN
(SELECT Dnumber      FROM DEPARTMENT     WHERE Dname='Research’);
```

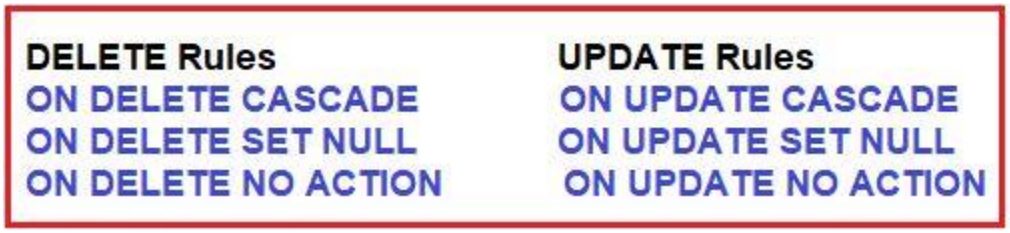
(To remove all rows in the employees table)

```
DELETE FROM EMPLOYEE;
```

16. UPDATE

```
UPDATE  PROJECT
SET Plocation='Bellaire', Dnum = 5
WHERE Pnumber=10;
---------------------
```

17. Referential Integrity



DELETE Rules
ON DELETE CASCADE
ON DELETE SET NULL
ON DELETE NO ACTION

UPDATE Rules
ON UPDATE CASCADE
ON UPDATE SET NULL
ON UPDATE NO ACTION

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.

- Adding the Foreign Key Constraint

ALTER TABLE EMPLOYEE ADD FOREIGN KEY (Dno) REFERENCES DEPARTMENT (Dnumber);

Or

ALTER TABLE EMPLOYEE ADD FOREIGN KEY(Dno) REFERENCES DEPARTMENT(Dnumber)

ON DELETE CASCADE

ON UPDATE CASCADE;

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.

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.

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 in MySQL?

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.

---

18. SQL Create Constraints

Constraints can be specified when the table is created with the CREATE TABLE statement, or after the table is created with the ALTER TABLE statement.

Syntax

```
CREATE TABLE table_name (
  column1 datatype constraint,
  column2 datatype constraint,
  column3 datatype constraint,
  ....
);
```

The following constraints are commonly used in SQL

- **NOT NULL** - Ensures that a column cannot have a NULL value
- **UNIQUE** - Ensures that all values in a column are different
- **PRIMARY KEY** - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
- **FOREIGN KEY** – A FOREIGN KEY is a field (or collection of fields) in one table, that refers to the PRIMARY KEY in another table. The FOREIGN KEY constraint prevents invalid data from being inserted into the foreign key column
- **CHECK** - Ensures that the values in a column satisfies a specific condition
- **DEFAULT** - Sets a default value for a column if no value is specified
- **CREATE INDEX** - Used to create and retrieve data from the database very quickly

Practice from link: https://www.w3schools.com/sql/sql_constraints.asp

1. SQL NOT NULL on CREATE TABLE

Example

```
CREATE TABLE Persons (
   ID int NOT NULL,

   LastName varchar(255) NOT NULL,

   FirstName varchar(255) NOT NULL,

   Age int

);
```

SQL NOT NULL on ALTER TABLE

```
ALTER TABLE Persons
MODIFY Age int NOT NULL;
```

2. SQL UNIQUE Constraint

SQL Server / Oracle / MS Access:

```
CREATE TABLE Persons (
   ID int NOT NULL UNIQUE,
   LastName varchar(255) NOT NULL,
   FirstName varchar(255),
   Age int
);
```

MySQL:

```
CREATE TABLE Persons (
   ID int NOT NULL,
   LastName varchar(255) NOT NULL,
   FirstName varchar(255),
   Age int,
   UNIQUE (ID)
);
```

SQL UNIQUE Constraint on ALTER TABLE

```
ALTER TABLE Persons
ADD UNIQUE (ID);
```

3. SQL CHECK Constraint

The CHECK constraint is used to limit the value range that can be placed in a column.

MySQL:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    CHECK (Age>=18)
);
```

SQL Server / Oracle / MS Access:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int CHECK (Age>=18)
);
```

4. SQL DEFAULT Constraint

The DEFAULT constraint is used to set a default value for a column.

My SQL / SQL Server / Oracle / MS Access:

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    City varchar(255) DEFAULT 'Sandnes'
);
```

5. SQL CREATE INDEX Statement

The CREATE INDEX statement is used to create indexes in tables.

Indexes are used to retrieve data from the database more quickly than otherwise. The users cannot see the indexes, they are just used to speed up searches/queries.

CREATE INDEX Syntax: Creates an index on a table. Duplicate values are allowed

```
CREATE INDEX index_name

ON table_name (column1, column2, ...);
```

Example1:

CREATE INDEX idx_lastname
ON Persons (LastName);

Example2:

CREATE INDEX idx_pname
ON Persons (LastName, FirstName);

19. Example for constraints Practice

Create DEPARTMENT table with attributes (Dname, Dnumber, Mgr_ssn, Mgr_start_date) in COMPANY Database. The Dname should be **DEFAULT** value 'COMPUTER', Dnumber should be **NOT NULL** and Mgr_ssn should be **less than or equal to 8**. Check the constraints with inserting values in the DEPARTMENT Table.

Step-1: Create Database COMPANY

CREATE DATABASE COMPANY;

Step-2: Use COMPANY Database

USE COMPANY;

Step-3: Create DEPARTMENT table with all constraints

CREATE TABLE DEPARTMENT (
Dname VARCHAR(10) **DEFAULT** 'COMPUTER',
Dnumber INT **NOT NULL**,
Mgr_ssn INT **CHECK** (Mgr_ssn <=8) ,
Mgr_start_date date,
PRIMARY KEY (Dnumber));

Step-4: Insert diffent values and check the constraints

INSERT INTO DEPARTMENT (Dname, Dnumber, Mgr_ssn, Mgr_start_date) VALUES
('PHYSICS', 1, 4, ''30-DEC-52'),
('CHEMISTRY', 2, 5, ''18-DEC-52'),
('COMPUTER', 3, 6, ''17-DEC-52');

INSERT INTO DEPARTMENT (Dnumber, Mgr_ssn, Mgr_start_date) VALUES
(4, 8, ''19-DEC-52'),
(5, 9, ''20-DEC-52');