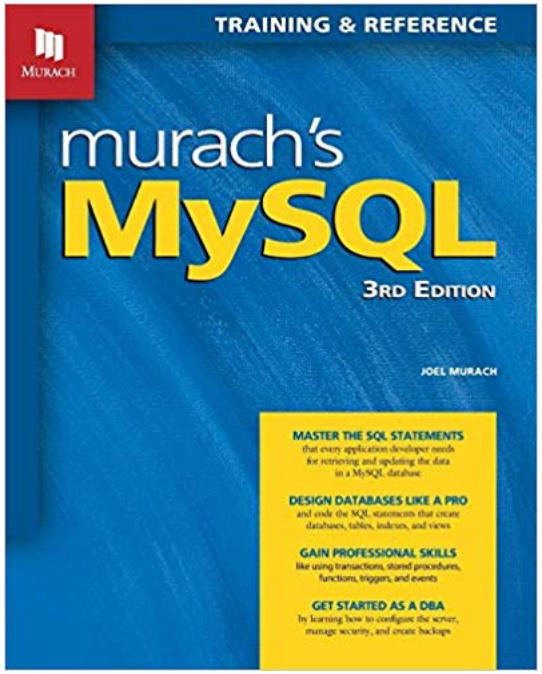
Introduction to SQL CREATE

Topic 2

Lesson 2 – SQL CREATE TABLE, DATABASE

Part of Chapter 11 Murach's MySQL



CREATE a database

```
CREATE DATABASE [IF NOT EXISTS] db_name;
```

A relation must exist within a specific database. To create a relation you must first create the encompassing database.

You can set your context to that database using the USE command : `USE db_name;`

Syntax for the CREATE TABLE command

CREATE TABLE name(field1 field_type1 field1_attributes
[, field2 field_type2 field2_attributes]
[, table_constraints]) ;

MySQL supports the following field constraints: NOT NULL, UNIQUE, DEFAULT default_value, AUTO_INCREMENT

EXAMPLE:

```
CREATE TABLE available_major (
        major VARCHAR(30) PRIMARY KEY );
```

CREATE a table

```
CREATE TABLE available_major (
        major VARCHAR(30) PRIMARY KEY );

CREATE TABLE dbname.available_major (
        major VARCHAR(30) PRIMARY KEY );
```

CREATE table is a data definition command that creates the structure of a table. Once the table is created, we can use the INSERT command to add data to the table.

available_major table

major

EXAMPLE: CREATE a table

```
CREATE TABLE student (id INT AUTO_INCREMENT
                          PRIMARY KEY ,
                      name VARCHAR(30) NOT NULL,
                      school VARCHAR(30),
                      credit_earned INT DEFAULT 0,
                      credit_req INT NOT NULL);
```

id	name	school	credits_earned	credits_req
1	Smith	Khoury	32	120
2	Shah	D'Amore McKim	64	128
3	Li	Khoury	50	120

EXAMPLE: CREATE a table constraint

```
CREATE TABLE student (id INT AUTO_INCREMENT,
        name VARCHAR(30) NOT NULL,
        school VARCHAR(30),
        credit_earned INT,
        credit_req INT,
        CONSTRAINT student_pk PRIMARY KEY (id));
```

Name of constraint

Type of constraint

id	name	school	credits_earned	credits_req
1	Smith	Khoury	32	120
2	Shah	D'Amore McKim	64	128
3	Li	Khoury	50	120

EXAMPLE: CREATE composite primary key

```
CREATE TABLE student_major (
        student_id INT,
        major VARCHAR(30),
        CONSTRAINT major_pk PRIMARY KEY
                (student_id, major));
```

These fields are also foreign keys we use the CONSTRAINT clause to define FOREIGN KEYS.

student_id	major
1	CS
1	Accounting
2	CS
3	DS

EXAMPLE: CREATE foreign keys

```
CREATE TABLE student_major (student_id INT,
        major VARCHAR(30),
        CONSTRAINT major_pk PRIMARY KEY
                (student_id, major),
        CONSTRAINT s_major_fk_s FOREIGN KEY
                (student_id) REFERENCEs student (id),
        CONSTRAINT s_major_fk_m FOREIGN KEY
        (major) REFERENCEs available_major (major));
```

student_id	major
1	CS
1	Accounting
2	CS
3	DS

EXAMPLE: Parent and child tables

A foreign key is defined in the child table. The child table references a tuple in the parent table.

The foreign key puts restrictions on the operations that can be performed on the parent table as well as the child table.

available_major

major
CS
Accounting
DS

student

id	name	school	credits_earned	credits_req
1	Smith	Khoury	32	120
2	Shah	D'Amore McKim	64	128
3	Li	Khoury	50	120

student_major

student_id	major
1	CS
1	Accounting
2	CS
3	DS

Example: child table foreign key behavior

INSERT INTO **student_major** VALUES (6, 'DS');
This operation would fail. Why?

Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails ('db'.'student_major', CONSTRAINT 's_major_fk_s' FOREIGN KEY ('student_id') REFERENCES 'student' ('id'))

major
CS
Accounting
DS

Student_major

student_id	major
1	CS
1	Accounting
2	CS
3	DS

Student

id	name	school	credits_earned	credits_req
1	Smith	Khoury	32	120
2	Shah	D'Amore McKim	64	128
3	Li	Khoury	50	120

Example: Foreign key behavior

UPDATE **student_major** SET student_id = 6
WHERE major = 'DS';

Would this operation fail. Why?

UPDATE of the child table, **Yes**, there is no student in the student table with an id of 6

available_major

major
CS
Accounting
DS

student_major

student_id	major
1	CS
1	Accounting
2	CS
3	DS

Student

id	name	school	credits_earned	credits_req
1	Smith	Khoury	32	120
2	Shah	D'Amore McKim	64	128
3	Li	Khoury	50	120

Example: Parent table update

UPDATE **student** SET student_id = 6 WHERE id = 3;
Would this operation fail?

UPDATE of the parent table,
Yes, since we have tuples In the student_major table for student_id with id = 3

available_major

major
CS
Accounting
DS

Student_major

student_id	major
1	CS
1	Accounting
2	CS
3	DS

Student

id	name	school	credits_earned	credits_req
1	Smith	Khoury	32	120
2	Shah	D'Amore McKim	64	128
3	Li	Khoury	50	120

Example 2: Foreign key behavior

UPDATE **student_major** SET student_id = 1
WHERE major = 'DS';

Would this operation fail?

UPDATE of the child table, **No**, student_id of 1 exists in the student table.

available_major

major
CS
Accounting
DS

student_major

student_id	major
1	CS
1	Accounting
2	CS
3	DS

Student

id	name	school	credits_earned	credits_req
1	Smith	Khoury	32	120
2	Shah	D'Amore McKim	64	128
3	Li	Khoury	50	120

Specifying database behavior for FKs

The Foreign key constraint allows you to specify the type of behavior the database should perform when a tuple referenced by a foreign key is being **update**d or **delete**d in the parent table.

The specifications are:

ON UPDATE [RESTRICT| CASCADE | SET NULL | SET DEFAULT]

ON DELETE [RESTRICT| CASCADE | SET NULL | SET DEFAULT]

RESTRICT is the default behavior, this means if an operation would DELETE or UPDATE a record being referenced with a foreign key, the DELETE or UPDATE operation would fail.

Example: RESTRICT behavior

DELETE **student** where id = 1;

This operation would fail due to the table specifications.

```
CREATE TABLE student_major (student_id INT,
                            major VARCHAR(30),
  CONSTRAINT s_major_pk PRIMARY KEY (student_id, major),
  CONSTRAINT s_major_fk_s FOREIGN KEY student_id REFERENCES
                            Student (id) ON DELETE RESTRICT,
  CONSTRAINT s_major_fk_m FOREIGN KEY major REFERENCES
                            available_major (major) ON DELETE RESTRICT);
```

student_id	major
1	CS
1	Accounting
2	CS
3	DS

major
CS
Accounting
DS

id	name	school	credits_ earned	credits_req
1	Smith	Khoury	32	120
2	Shah	D'Amore McKim	64	128
3	Li	Khoury	50	120

Example: SET NULL behavior

DELETE **available_major** where major = ‘DS”;
This operation would succeed due to the table specifications.

```
CREATE TABLE s_major (m_pk INT AUTO_INCREMENT PRIMARY KEY,
                      studentID INT,
                       major VARCHAR(30),
  CONSTRAINT major_fk_s FOREIGN KEY student_id REFERENCES
                        student (id) ON DELETE RESTRICT,
  CONSTRAINT major_fk_M FOREIGN KEY major REFERENCES
                        Available_Majors (Major) ON DELETE SET NULL);
```

m_pk	student_id	major
1	1	CS
2	1	Accounting
3	2	CS
4	**3**	**DS**

m_pk	student_id	major
1	1	CS
2	1	Accounting
3	2	CS
4	**3**	**NULL**

major
CS
Accounting
DS

major
CS
Accounting

Example: CASCADE behavior

DELETE **student** where id = 1;

What would the schema look like after this command is executed given the following foreign key definitions?

```
CREATE TABLE student_major (student_id INT,
                            major VARCHAR(30),
  CONSTRAINT major_pk PRIMARY KEY (student_id, major),
  CONSTRAINT s_major_fk_s FOREIGN KEY student_id REFERENCES
                            student (id) ON DELETE CASCADE,
  CONSTRAINT s_major_fk_m FOREIGN KEY major REFERENCES
                            available_major (major) ON DELETE CASCADE);
```

Schema before DELETE command

DELETE **student** where id = 1;

Will student with id 1 be deleted? If so, what happens to the 2 tuples in the student_major table that reference that tuple?

available_major

major
CS
Accounting
DS

Student_major

student_id	major
1	CS
1	Accounting
2	CS
3	DS

Student

id	name	school	credits_earned	credits_req
1	Smith	Khoury	32	120
2	Shah	D'Amore McKim	64	128
3	Li	Khoury	50	120

Result of the DELETE command

DELETE student where id = 1;

This operation would delete the Student tuple and all tuples that reference that tuple due to the foreign key ON DELETE specifications, the 2 tuples in the student_major table would also be deleted

```
CREATE TABLE student_major (student_id INT,
                            major VARCHAR(30),
  CONSTRAINT major_pk PRIMARY KEY (student_id, major),
  CONSTRAINT s_major_fk_s FOREIGN KEY student_id REFERENCES
                            student (id) ON DELETE CASCADE,
  CONSTRAINT s_major_fk_m FOREIGN KEY major REFERENCES
                            available_major (major) ON DELETE CASCADE);
```

student_id	major
2	CS
3	DS

id	name	school	credits_earned	credits_req
2	Shah	D'Amore McKim	64	128
3	Li	Khoury	50	120

MYSQL WORK

Let's create a database named student and create tables for the student, available_major and student_major table.

Summary

In this module you learned:

- SQL CREATE DATABASE command
- USE command
- SQL CREATE TABLE command
- Creating field constraints
- Creating primary keys
- Creating foreign keys and specifying UPDATE and DELETE behavior
-