

SQL DDL and CRUD

Structured Query Language

- ▶ Practical implementation of the relational model
- ▶ Originally SEQUEL (Structured English QUEry Language) at IBM research
- ▶ SQL became standard in 1986
- ▶ Supported by all major RDBMS vendors, with minor (and sometimes major) differences

SQL's big advantage: if you stick to ANSI SQL, your database code is portable between RDBMS systems.

SQL Relational Model

- ▶ Relations are **tables**
- ▶ Tuples are **rows**
- ▶ Attributes are **columns**

For the most part these terms are interchangeable.

- ▶ Important difference: tables allow duplicate rows

Schemas and Catalogs

A **schema** (database in the relational model) is a collection of related tables and constructs. A schema has:

- ▶ a schema name
- ▶ an authorization identifier (user who owns the schema)

In MySQL `create schema` is a synonym for `create database`.

A catalog is a named collection of schemas. MySQL includes a `table_catalog` column in its `information_schema.tables` table for compatibility with the SQL standard, but does not use catalogs.

CREATE TABLE

The CREATE TABLE command creates a **base table** (CREATE VIEW creates a **virtual** or **derived** table):
General form:

```
CREATE TABLE <table_name> (  
    <column_name> <column_type>  
        <column_constraints>...,  
    [... ,]  
    <table_constraints>,  
    [...]  
);
```

CREATE TABLE Example

```
CREATE TABLE pub (  
  pub_id INT PRIMARY KEY,  
  title VARCHAR(16) NOT NULL,  
  book_id INT NOT NULL REFERENCES book(book_id)  
);
```

By convention, SQL keywords are in ALL CAPS in instructional examples but not when typing.

Note: see [pubs-schema.sql](#) and [pubs-data.sql](#) for examples of SQL database creation and population commands.

Column Types

Each column, or attribute, is given a data type (domain in the relational model). MySQL has

- ▶ Numeric data types,
- ▶ String data types, and
- ▶ Temporal data types.

Get comprehensive documentation at <http://dev.mysql.com/doc/refman/5.7/en/data-types.html>.
We'll cover the most commonly used data types.

Numeric Data Types

- ▶ INT
- ▶ FLOAT or DOUBLE
- ▶ DECIMAL

String Data Types

- ▶ CHAR
- ▶ VARCHAR
- ▶ TEXT
- ▶ ENUM

Temporal Dta Types

- ▶ DATE
- ▶ DATETIME
- ▶ TIMESTAMP
- ▶ TIME

Constraints

- ▶ Attribute (a.k.a. column) constraints
- ▶ Key (a.k.a. unique)
- ▶ Primary key
- ▶ Foreign key

We'll also learn named constraints, assertions and triggers in Advanced SQL.

Key and Primary Key Constraints

Key:

```
name CHAR(10) UNIQUE,
```

Primary key:

```
pub_id INT PRIMARY KEY,
```

A primary key is implicitly UNIQUE

Foreign Key Constratins

```
book_id INT NOT NULL REFERENCES book(book_id)
```

Notice also that we don't allow `book_id` to be `NULL`. So `pub` totally participates in its relationship with `book`.

CHECK Constraints

```
CREATE TABLE bartender (  
    id INT PRIMARY KEY,  
    name VARCHAR(10) NOT NULL,  
    age INT CHECK (age > 20)  
);
```

Note: MySQL does not enforce CHECK constraints. We'll learn about triggers in Advanced SQL.

INSERT Command

General form is

```
INSERT INTO <table_name> (<column_name> [, ...])  
VALUES (<new_value> [, ...]);
```

Example:

```
insert into author (author_id, first_name,  
second_name)  
values (1, "Jenny","McCarthy");
```

Can leave off column names list to insert positionally:

```
insert into author values (1,  
"Jenny","McCarthy");
```

UPDATE Command

General form:

```
UPDATE <table_name> SET  
    <column_name>=<new_value> [, ...] WHERE  
    expression
```

Example: Surely we meant Lisp inventor, AI co-founder, and Turing Award winner John McCarthy instead of anti-vaxxer Jenny McCarthy.

```
update author set first_name='John' where  
    last_name='McCarthy';
```

Notice that we can use single or double quotes in most RDBMS systems.

DELETE Command

General form:

```
DELETE FROM <table_name> WHERE  
    <boolean_expression>;
```

Example:

```
delete from author where last_name="Batty";
```

Can also drop whole tables:

```
DROP TABLE <table_name>
```

Referential Integrity

To maintain integrity on update or delete specify:

- ▶ For ON DELETE:
 - ▶ SET NULL
 - ▶ SET DEFAULT
- ▶ For ON UPDATE
 - ▶ CASCADE

Note: for MySQL ON DELETE RESTRICT is the default.

Referential Integrity - SET NULL

Example:

```
CREATE TABLE pub (  
  pub_id INT PRIMARY KEY,  
  title VARCHAR(16) NOT NULL,  
  book_id INT,  
  foreign key (book_id) REFERENCES book(book_id)  
    ON DELETE SET NULL  
);
```

Means that if the row from the book table containing book_id is deleted, then book_id is set to NULL for each affected row in the pub table.

Notice that if you choose SET NULL as your ON DELETE action your column definition must allow nulls.

Referential Integrity Constraints in MySQL

MySQL will only enforce referential integrity constraints that are specified separately from column definitions as above. The following syntax:

```
CREATE TABLE pub (  
    pub_id INT PRIMARY KEY,  
    title VARCHAR(16) NOT NULL,  
    book_id INT REFERENCES book(book_id) ON DELETE  
        SET NULL  
);
```

is valid SQL syntax but is ignored by MySQL's default InnoDB engine.

Referential Integrity - SET DEFAULT

Example:

```
CREATE TABLE pub (  
  pub_id INT PRIMARY KEY,  
  title VARCHAR(16) NOT NULL,  
  book_id INT DEFAULT 0 REFERENCES book(book_id)  
    ON DELETE SET DEFAULT  
);
```

Means that if the row from the book table containing book_id is deleted, then book_id is set to 0 for each affected row in the pub table.

Note: MySQL's default InnoDB engine does not implement ON DELETE SET DEFAULT.

Referential Integrity - CASCADE

Example:

```
CREATE TABLE pub (  
    pub_id INT PRIMARY KEY,  
    title VARCHAR(16) NOT NULL,  
    book_id INT NOT NULL,  
    FOREIGN KEY (book_id) REFERENCES book(book_id)  
        ON UPDATE CASCADE  
);
```

Means that if a book_id value changes for a row in the book table, the change is applied to the affected rows of the pub table also.

Multiple Referential Integrity Constraints

You would normally set constraints for updates and deletes.

Example:

```
CREATE TABLE pub (  
  pub_id INT PRIMARY KEY,  
  title VARCHAR(16) NOT NULL,  
  book_id INT,  
  FOREIGN KEY (book_id) REFERENCES book(book_id)  
    ON UPDATE CASCADE  
    ON DELETE SET NULL  
);
```