Managing Data & Databases

Session 4
Save your lists

Parts of SQL

- DDL
  - Data Definition Language
- DML
  - Data Manipulation Language
- DCL
  - Data Control Language
- TCL
  - Transaction Control Language

- We mainly cover the first two

Today's Dose of SQL Concepts

- **Field:** Is the unit of data in databases. Ideally each field should contain only one piece of information. The fields can also be completely empty (Null).
- **Null Value:** Nothing. Zilch. This field has been vacated, or nothing has ever been assigned to it.
- **Table:** Is the unit of data structure in RDBMSs. Tables are composed of rows (records) and columns. The intersection of each column and row is a cell called field.
- **Columns:** Each column of a table has a definite data type, and may impose a number of constraints on the data that it captures.
- **Row:** Is a combination of data units called field. Each row should ideally include data about only one entity.
- **Primary Key:** Is the unique identifier of each row. We normally use auto-increment numeric fields to generate PKs, but that rule has many exceptions.
- **Auto-increment:** Is a numeric field that is auto-generated each time you enter a row of data.
- **Constraints** (The list is not exhaustive here. You are encouraged to read more.)
  - **Uniqueness:** Ensures no repetitive values are entered in a column (or a combination of columns).
  - **Not null:** Ensures no row is added with no value in a field affected by not null constraint.
- **Index:** Indexes improve data sorting, searching and joining. We add an index to a column when we know that column will be serving as some kind of search or sort key. Indices usually duplicate the data, so they can become space hogs.

Today's Dose of SQL
DDL

- Create
- Alter
- Drop

SQL Command
CREATE

- Use case
  - Create is used to “create” different types of objects in a database. The one we learn about today is “table”.
- Syntax
  - CREATE TABLE `table_name` (`column 1` data type for column 1 [column 1 constraint(s)], ... , [table constraint(s)]);
  - You can also create a table by right-clicking “Tables” in Workbench’s sidebar and then selecting “Create Table”
- Example
  - CREATE TABLE people (`id` int(11) NOT NULL AUTO_INCREMENT, `first_name` varchar(45) NOT NULL, `last_name` varchar(45) NOT NULL, `birth_date` date DEFAULT NULL, PRIMARY KEY (`id`), UNIQUE KEY `id_UNIQUE` (`id`));

Some naming conventions (1)

- .
  - x.y can be read as “y” in “x”
  - Let’s say table “people” in database “class”: class.people
  - Let’s say field “first_name” in table “people”: people.first_name
  - Let’s combine both: class.people.first_name
  - MySQL’s query parser will guess what object you are referring to by looking at the context you are making the call. Sometimes it is not necessary to give the full address of an object when calling it, but it is generally recommended to do so in order to avoid confusion.

Some naming conventions (2)

- `
  - Backticks are optional to use unless you have use some “special” characters when naming your database objects. MySQL Workbench is just being conservative when surrounding every name with a backtick.
  - Don’t use space nor any sign (underscore “_”) is an exception, or you will have to use backticks when calling your objects all the time
- ‘’ or “”
  - Single or double quotes describe “text”. They tell the query parser that what it sees does not need any interpretation, and it is actually simple text. Whether both work for you and when to use each is a matter of server configuration and many other things. Use the one that works on your DB, or have a look here.
- Define a naming convention for yourself and abide by it. Mine is: Always small letters; words separated by underscore.

SQL Command
ALTER

- Use case
  - ALTER is used to “alter” different types of objects in a database. We don’t cover it in class today, but you may have to learn it in order to add a field to your table, or remove a constrain, etc…
- Syntax
  - ALTER TABLE `table_name` [alter specification];
  - You can also alter a table by right-clicking it’s name in Workbench’s sidebar and selecting “Alter Table”

SQL Command
DROP

- Use case
  - DROP is used to “remove” different types of objects in a database. Here we focus on dropping tables.
- Syntax
  - DROP TABLE `table_name`;
  - You can as well drop a table by right-clicking its name in Workbench’s sidebar and selecting “Drop Table”.
- Example
  - DROP TABLE people;

Today's Dose of SQL DML

- Insert
- Select
- Update
- Delete

SQL Commands
INSERT INTO

- Use case
  - INSERT INTO is used to “insert” data in a table.
- Syntax
  - INSERT INTO "table_name” [("column1”, …)] VALUES ("value1”, ...);
  - You have probably noticed that when you view your table contents you can actually edit the fields.
- Example
  - INSERT INTO people VALUES (NULL, "Joe", "Dassin", NULL);
  - INSERT INTO people (first_name, last_name, birth_date) VALUES (‘Jean’, ‘Nouvel’, ‘1945-08-12’);

SQL Commands
SELECT

- Use case
  - SELECT is used to “fetch” data from tabular data containers (tables, etc…) in a database. Here we focus on selecting from tables.
- Simplified Syntax
  - SELECT `field1_name`, … FROM `table_name`;
  - You can as well select data from a table by right-clicking its name in MySQL’s sidebar and selecting “Select Rows”.
- Example
  - SELECT * FROM people;
  - SELECT id, last_name FROM people;

SQL Commands
WHERE Clause

- Use case
  - WHERE is used to “limit” the domain of reference of many of SQL commands that deal with fetching or manipulating data.
- Simplified Form
  - WHERE `field_name` condition;
- Example (The list of conditions is not exhaustive)
  - SELECT * FROM people WHERE first_name = “Mahmood”;
  - SELECT * FROM people WHERE first_name like "%oo%”;
    - Fetch all info on people with a first name containing double O.
  - SELECT * FROM people WHERE birth_date IS NOT NULL;
  - SELECT last_name FROM people WHERE id BETWEEN 1 AND 3;
    - BETWEEN is inclusive

SQL Commands
UPDATE

- Use case
  - UPDATE is used to “update” the data in a range of rows in a table. Works exclusively with a WHERE clause.
- Syntax
  - UPDATE "table_name” SET "column_1" = [some value] WHERE "condition";
- Example
  - UPDATE people SET birth_date = "1938-11-05" WHERE first_name = "Joe" AND last_name = "Dassin";

SQL Commands
DELETE FROM

- Use case
  - DELETE FROM is used to “delete” a range of rows from a table. In most databases it necessarily requires a WHERE clause to work.
- Syntax
  - DELETE FROM "table_name” WHERE "condition";
- Example
  - DELETE FROM people WHERE birth_date IS NULL;
  - If you encounter an error when trying to delete, see “Why MySQL doesn't let me delete my records?” from our newborn FAQ.
- Bonus
  - If you want to empty a table you don’t have to delete every record. All you have to do is to “truncate” the table.