Introduction to SQL

The Basics

Jon Flanders
@jonflanders

pluralsight
hardcore developer training

What is SQL?

- Structured Query Language
- A special-purpose programming language
  - Its purpose is to manipulate relational databases
  - Declarative language
  - Contains both a data definition syntax as well as a data manipulation syntax
- It’s both an ANSI and ISO standard
  - In this course, I’ll stick to the standards-based SQL–hopefully without the standards-based biased toward complex concepts

What is a Database?

- A container to help organize data in a constructive way
- Useful when you have some threshold amount of data
  - Think 500 Excel spreadsheets instead of 5
- Putting all the data in one place makes it easy to
  - Query the data
  - Update the data
  - Insert new data
  - Delete old data
- One source of truth rather than many sources of truth
- There are many different types of databases
  - Relational
  - Object-oriented
  - Document-based
- Some large number of databases in use today are classified as using the “relational model”
  - SQL is a language built for this model

What Does “Relational” Mean?

- The relational model is the database model on which SQL is based
  - It is a way to describe data and the relationships between data entities
- The pure Relational Model is math all the way down
  - Originally was based on relational algebra and tuple relational calculus
- SQL has either “watered-down” or “improved” the “pure” relational model over time
  - But we still use the term relational to describe the system in use today

Tables, Columns, and Rows

- In a relational database, data is stored in a construct known as a Table
  - A Table has a name and a collection of Columns
- Each Column also has a name
  - Each Column also has a restriction that restricts the size and category of data that can be stored in that Column
  - Also each Column can be required or not-required
- Each row contains data for at least all the required columns
- Rows can be retrieved by asking questions of the data
  - The questions will be related to the values in columns (e.g. “Who are all my contacts that have a phone number that starts with 310?”)
  - Asking such questions with SQL is known as querying

Keys

- Another essential piece of a relational model is keys
- Each Table should/must have one column that can uniquely identify a row
  - This column is known as the Primary Key
- If more than one table uses the same primary key, you can then merge those two tables together
  - More than likely you will want to merge a subset of each table
- A table can also have a column that is classified as a Foreign Key
  - A Foreign Key links that table to another Table's Primary Key
  - This also allows rows from each table to be linked together
- Sometimes keys are "natural"
  - Like an ISBN number for a book – unique across all published books on the planet
- Sometime you have to "invent" keys
  - Usually an integer is sufficient
  - Often a database will add this data in each row automatically for you – often called an "identity" column

Example: A Database for Contacts

- Why not just have one table?
- Imagine a simple Contacts database
  - Just first name, last name, and email address

First Name	Last Name	Email
Jon	Flanders	jon@...
Fritz	Onion	fritz@...

- What if Jon has more than one email address?
- Tempting to just add a column
  - Now we are just limited to two emails
  - Something isn’t right here
  - Limits the questions we can potentially ask this data in the future

First Name	Last Name	Email1	Email2
Jon	Flanders	jon@...	jon2@...
Fritz	Onion	fritz@...	null

A Better Solution

Key	First Name	Last Name
1	Jon	Flanders
2	Fritz	Onion

Key	Person Key	Email
1	2	fritz@...
2	1	jon@...
3	1	jon2@...

- Now we can ask better questions
- “What are all of Jon’s email addresses?”
- “How many email addresses does each person have?”

- This is a design process in database design known as Normalization
- Much more to normalization
- Concentrate on what questions can be asked – you’ll do pretty well

SQL Statement

- A valid SQL Statement is made up of an actionable set of valid words
  - Some of the words are defined by SQL, some are defined by you
  - SQL is english-based so it is “readable”
  - Most people think of “query” when they think of SQL but it can do much more than just query
  - Most parts of a SQL Statement can be broken down into “clauses”
- A valid SQL Statement has a semi-colon (;) at the end of the statement
  - You can put multiple statements together, separated by semi-colons
- SQL is not case-sensitive
  - Most people write SQL Statements with SQL-specific words in all caps, and user-defined words in lower case
- Comments
  - Comments are good for documentation of SQL Statements saved for re-use
  - -- is for single-line comments
  - /* is for multi-line comments*/

Commands

- SQL Statements all start with a command
  - Generally the command is a verb
- SELECT is an example

```
SELECT  VALUES
FROM TABLENAME   ;
```

Commands

- SQL Statements all start with a command
  - Generally the command is a verb
- SELECT is an example

```
SELECT  MyColumnName,‘Constant’
FROM MyTableName ;
```

After the Command

- What comes next is highly dependent on the command itself
- For example – after the SELECT command comes
  - A definition of the set you want returned – a list of columns, and/or a wildcard (*) to indicate all columns, and/or a set of literal values. This is not optional
  - A FROM clause which contains one or more table names. This is actually optional (if the set contains only literal values)
- There is much more to using SELECT that will be covered later in this course – stay tuned

Naming Things

- Naming things is also important for being able to use a database successfully
  - Different people have different ideas on the right way to name things in a database
- I will use a few simple rules in this course
  - Table names will be singular: a Table name describes what a row of data in that table “is” (e.g. user, email, phone)
  - Column names will never be repeated inside of a particular database – this is to keep things clear when looking at names of columns
- Names in SQL are “scoped”
  - A Database will have a name
  - Tables inside of a database will have a name, but a Table name should be referred to using its “full” name, which includes the Database name
  - A period is used to separate each part of the name
  - Database.Table
  - Columns are also scoped: Table.Column

Creating Things with SQL

- There is a whole set of SQL commands that relate to creating and modifying constructs in a database

```
--this is not ANSI SQL
--but is supported by most vendors
CREATE DATABASE Contacts;
USE DATABASE Contact;
```

```
CREATE TABLE Contacts.User (...);
```

Data Types (not exhaustive)

- Each Column has a restriction on the type of data that can be stored inside of it

Data Type	Value Space
CHARACTER	Can hold N character values – set to N statically
CHARACTER VARYING	Can hold N character values – set to N dynamically – storage can be less than N
BINARY	Hexadecimal data
SMALLINT	-2^15 (-32,768) to 2^15-1 (32,767)
INTEGER	-2^31 (-2,147,483,648) to 2^31-1 (2,147,483,647)
BIGINT	-2^63 (-9,223,372,036,854,775,808) to 2^63-1 (9,223,372,036,854,775,807)
BOOLEAN	TRUE or FALSE
DATE	YEAR, MONTH and DAY in the format YYYY-MM-DD
TIME	HOUR, MINUTE and SECOND in the format HH:MM:SS[.sF] where F is the fractional part of the SECOND value
TIMESTAMP	Both DATE and TIME

RDMS

- Relational Database Management System
  - More than just the Database
  - Also includes tools and applications to help manage larger-scale database installations
- Most extend the ANSI SQL language with vendor-specific extensions
- Most are proprietary
  - Oracle has PL/SQL
  - SQL Server has T-SQL
- Most ANSI SQL will work with any RDMS
  - Notice the key term here “most”

SQL in This Course

- In this course I am using ANSI SQL
  - When you use your RDMS, some details could be different
  - I'll try to point as many of those differences out that I can
- Why I am using MySQL?
  - Free and open-source
  - Runs on all major platforms: Windows, OSX, Linux
  - Has an ANSI operation mode that enforces ANSI compliance (most RDMS's do not have this feature)

Summary

- SQL is a powerful declarative language that you can grasp by understanding a few basic concepts