intro_sql

November 23, 2020

Introduction to SQL

1 1. What is SQL?

- Structured Query Language
- Oftern pronounced "sequel" or by simply saying the letters (i.e., "ess cue el")
- Programming language for relational databases
- Absolutely ubiquitous
- Used by industry, governement, academia, everyone!

1.1 1.1 History, Motivations, and Standardization of SQL

- Relational model proposed by Ted Codd in 1970
- First software developed in the 70s and 80s by IBM and Oracle
- First ANSI standard in 1986
- Declarative language for querying relational databases

1.1.1 1.1.1 What makes SQL a Declarative Language?

- We don't have variables in the traditional sense
- We don't specify steps to take, or operations to peform
  - We don't have explicit loops for iteration
  - Instead we only what the "result set" should contain
  - The only "objects" we can create are `TABLE` objects, which are basically dataframes

1.2 1.2 Relational Databases

- Collection of 1 or more tables
- Tables have columns and rows
  - Rows represent "records"; columns represent "fields" or variables

1.2.1 1.2.1 Example Database Schema

1.2.2 1.2.2 Relational Database Management Systems (RDBMS)

- Software the implents a relational databases
- Allow us to:

  - create databases
  - create tables in databases
  - create users and control their access
  - query tables to extract data
  - add new data in to tables

1.2.3 1.2.3 Varieties of SQL and RDBMS

- Different RDBMS implement slightly different dialects of SQL
- RDBMS also provide features not in SQL standard
- Examples of RDBMS:
  - Microsoft SQL Server
  - Oracle
  - MySQL
  - PostgreSQL
  - Sqlite

2 2. Typical SQL Query

- Commands that query table(s) in a DB and return some output (often in tabular form)

2.1 2.1 The `SELECT` Statement

- Often the start of a query
- Specifies the columns we want

2.1.1 2.1.1 Motivating Example

- Suppose we run an online retailer (e.g., Amazon)
- We have a database with these tables:
  - `customers`
  - `orders`
  - `products`

```
In [1]: %load data/amzn.db
```

2.1.2 2.1.2 Query the `products` Table

```
In [2]: SELECT
            product,           -- these are the columns we want to include
            price,
            product_id
        FROM
            products           -- this is the table we are querying from
        ;
```

Out[2]:
product	price	product_id
rake	19.99	44
shoe horn	5.99	33
potato	0.99	22
bike	123.5	12
table	78.55	123
cup	2.5	232
ball	5.5	28
pencil	2.99	22
teapot	12.49	11
fork	1.99	13
shoelace	0.5	14
hammer	17.49	555
door	159.99	66

2.1.3 2.1.3 Query the `orders` Table

In [3]:
```
SELECT
    order_id,
    customer,
    date
FROM
    orders
;
```

Out[3]:
order_id	customer	date
1	lee	2018-12-23
2	smith	2020-03-12
3	jones	2019-05-01

```
+----------+----------+------------+
| 4        | yang     | 2020-09-12 |
+----------+----------+------------+
| 5        | guerra   | 2020-08-03 |
+----------+----------+------------+
| 6        | diaz     | 2020-11-28 |
+----------+----------+------------+
| 7        | riley    | 2019-05-18 |
+----------+----------+------------+
| 8        | chan     | 2018-10-03 |
+----------+----------+------------+
```

2.1.4 2.1.3 Query using `SELECT *`

In [4]:
```
SELECT
    *
FROM
    orders
;
```

Out[4]:
```
+----------+----------+------------+------------+----------+
| order_id | customer | date       | product_id | quantity |
+----------+----------+------------+------------+----------+
| 1        | lee      | 2018-12-23 | 44         | 1        |
+----------+----------+------------+------------+----------+
| 2        | smith    | 2020-03-12 | 33         | 1        |
+----------+----------+------------+------------+----------+
| 3        | jones    | 2019-05-01 | 212        | 2        |
+----------+----------+------------+------------+----------+
| 4        | yang     | 2020-09-12 | 12         | 1        |
+----------+----------+------------+------------+----------+
| 5        | guerra   | 2020-08-03 | 12         | 2        |
+----------+----------+------------+------------+----------+
| 6        | diaz     | 2020-11-28 | 123        | 2        |
+----------+----------+------------+------------+----------+
| 7        | riley    | 2019-05-18 | 232        | 4        |
+----------+----------+------------+------------+----------+
| 8        | chan     | 2018-10-03 | 28         | 1        |
+----------+----------+------------+------------+----------+
```

2.1.5 2.1.4 Query Without Formatting

- Capitalization, indentation, and other formatting oftern doesn't matter in SQL
- But you should use good standards, so others can read your code

In [5]:
```
select * from orders;
```

Out[5]:
```
+----------+----------+------------+------------+----------+
| order_id | customer | date       | product_id | quantity |
```

```
+----------+----------+------------+------------+----------+
| 1        | lee      | 2018-12-23 | 44         | 1        |
+----------+----------+------------+------------+----------+
| 2        | smith    | 2020-03-12 | 33         | 1        |
+----------+----------+------------+------------+----------+
| 3        | jones    | 2019-05-01 | 212        | 2        |
+----------+----------+------------+------------+----------+
| 4        | yang     | 2020-09-12 | 12         | 1        |
+----------+----------+------------+------------+----------+
| 5        | guerra   | 2020-08-03 | 12         | 2        |
+----------+----------+------------+------------+----------+
| 6        | diaz     | 2020-11-28 | 123        | 2        |
+----------+----------+------------+------------+----------+
| 7        | riley    | 2019-05-18 | 232        | 4        |
+----------+----------+------------+------------+----------+
| 8        | chan     | 2018-10-03 | 28         | 1        |
+----------+----------+------------+------------+----------+
```

2.2 3. The `WHERE` Clause

- We can use `WHERE` to set filering criteria
- Similar to the `filter()` function *dplyr* in R

2.3 3.1 Using `WHERE` in Query

In [6]:
```
SELECT
    order_id,
    customer,
    date,
    quantity
FROM
    orders
WHERE
    quantity > 1
;
```

Out[6]:
```
+----------+----------+------------+----------+
| order_id | customer | date       | quantity |
+----------+----------+------------+----------+
| 3        | jones    | 2019-05-01 | 2        |
+----------+----------+------------+----------+
| 5        | guerra   | 2020-08-03 | 2        |
+----------+----------+------------+----------+
| 6        | diaz     | 2020-11-28 | 2        |
+----------+----------+------------+----------+
| 7        | riley    | 2019-05-18 | 4        |
+----------+----------+------------+----------+
```

2.3.1 3.1.1 More Conditions in `WHERE` Clause

- We can actually specify quite a bit of criteria for rows we want to include using our `WHERE`
  - This includes testing for a columns equality to some value (or set of values)

In [7]:
```
SELECT
    order_id,
    customer,
    date,
    quantity
FROM
    orders
WHERE
    quantity > 1
    OR customer = 'lee'
;
```

Out[7]:
```
+----------+----------+------------+----------+
| order_id | customer | date       | quantity |
+----------+----------+------------+----------+
| 1        | lee      | 2018-12-23 | 1        |
+----------+----------+------------+----------+
| 3        | jones    | 2019-05-01 | 2        |
+----------+----------+------------+----------+
| 5        | guerra   | 2020-08-03 | 2        |
+----------+----------+------------+----------+
| 6        | diaz     | 2020-11-28 | 2        |
+----------+----------+------------+----------+
| 7        | riley    | 2019-05-18 | 4        |
+----------+----------+------------+----------+
```