Querying Data with SQL

By Blake Barr

Talk Overview

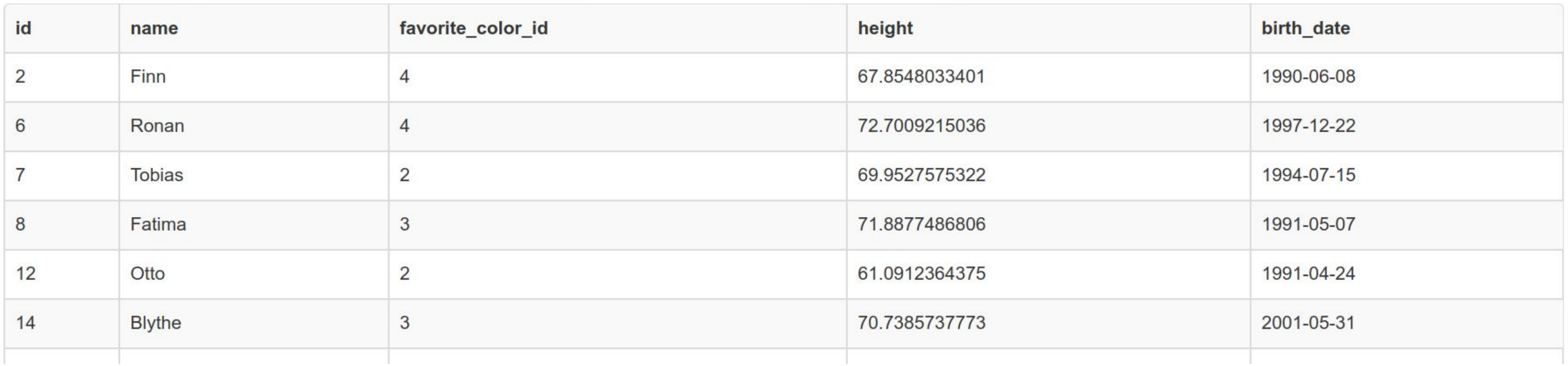
- Introduction to SQL
- Basics
    - SELECT, FROM, ORDER BY, LIMIT, WHERE
- More SQL Commands
    - GROUP BY, HAVING, JOINS
- Even More SQL Commands
    - IF STATEMENTS, , DATE_FORMAT, DISTINCT , CASE STATEMENTS, SUB-QUERIES
- Next Steps

Introduction to SQL

- Structured Query Language
- Standard Language for Relational Databases
- There are slightly different flavors of SQL but core functionality is same
- Popular SQL clients (that I know of)
    - Linux: Mysql Workbench
    - Mac: SQL Pro
    - Windows: Heidi SQL

Tables we will be working with

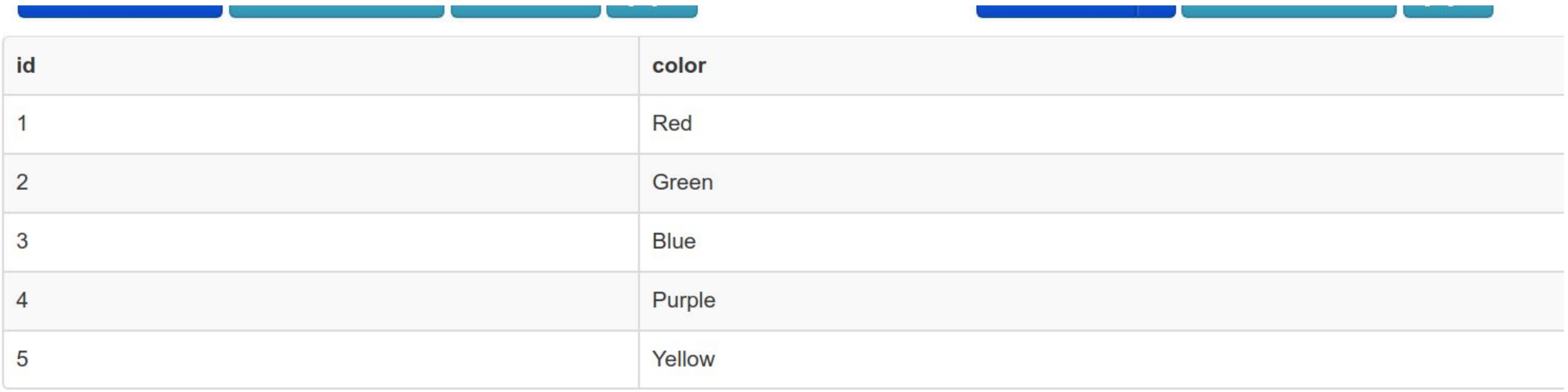
Table 1 - People



id	name	favorite_color_id	height	birth_date
2	Finn	4	67.8548033401	1990-06-08
6	Ronan	4	72.7009215036	1997-12-22
7	Tobias	2	69.9527575322	1994-07-15
8	Fatima	3	71.8877486806	1991-05-07
12	Otto	2	61.0912364375	1991-04-24
14	Blythe	3	70.7385737773	2001-05-31

Tables we will be working with

Table 2 - color



id	color
1	Red
2	Green
3	Blue
4	Purple
5	Yellow

Main SQL Commands: SELECT + FROM

- SELECT
  - Pick which columns you want view
  - SELECT * - VIEW ALL THE COLUMNS
  - SELECT column_1, column_2 - VIEW column_1 and column_2
- FROM
  - Choose the table you want to query data from
- Example:

  SELECT *
- FROM people;
- Can rename columns as well using AS - SELECT height AS height_in_inches

Main SQL Commands: Order by + Limit

- Order By
  - Allows you to sort your output (order by birth_date, earliest birth_dates first)
  - Order by height DESC (shows people who are taller first)
  - Put at the end (only thing that goes after order by is limit)
- Limit
  - Restrict number of rows in your output

Main SQL Commands: WHERE

- WHERE
  - USE TO FILTER THE ROWS YOU WANT
  - PROVIDE BOOLEAN CONDITIONS AND IT RETURNS ROWS WHERE BOOLEAN IS TRUE
- Example: Let’s say you have a table of people and their favorite colors
- SELECT *
- FROM table
- WHERE favorite_color = ‘Green’
- Will give you all the rows for people whose favorite color is green

Main SQL Commands: WHERE

- Things you can do with WHERE
  - Work with numbers
    - WHERE height > 25
    - WHERE weight BETWEEN 150 AND 200 (inclusive)
    - WHERE age != 25
    - WHERE age = 25
    - WHERE age IN (25, 30, 35)
  - Work with dates
    - WHERE created_date > ‘2019-06-01’
    - WHERE created_date = ‘1990-05-01’
    - WHERE created_date BETWEEN ‘1950-01-01’ AND ‘1960-01-01’

Main SQL Commands: WHERE

- Things you can do with WHERE
  - Work with strings
    - WHERE name = ‘Blake’
    - WHERE name IN (‘mickey’, ‘minnie’, ‘pluto’)
- Side Note : String Pattern Matching
  - What if you want all the rows where name starts with J
  - The % symbol allows you to match anything using the LIKE keyword
  - The _ symbol allows you to match any single character
    - Examples: WHERE name LIKE ‘J%’ - matches everything that starts with J
    - WHERE name LIKE ‘%n’ - matches everything that ends with n
    - WHERE name LIKE ‘%ane%’ - matches anyting with this pattern jane, bane, craner
    - WHERE name LIKE ‘Bl_ke’ - will match blake, bloke among anothers

Main SQL Commands: WHERE

- WHERE Exercises

Main SQL Commands: GROUP BY

- Allows you to group rows into categories and calculate statistics for those categories
- Example of why you may want to do this
  - Look at the number of people who come to your site every day (group by day)
  - Look at the number of people who have come your site from each state in last month (group by state and put where clause to only get rows in the last month)

Main SQL Commands: GROUP BY

- Syntax
- You put GROUP BY statement after WHERE clause
- You put aggregate functions in the SELECT statements
- Aggregate Functions - takes in multiple rows data and spits out a single value
  - MAX - get the max date or number MAX(date), MAX(height)
  - MIN - get min of date or number MIN(date), MIN(number)
  - COUNT(*) - count number of rows in a data table
  - COUNT(DISTINCT name) - count number of different names in a data table
  - SUM(country = ‘USA’) - SUM number of rows where country = USA
  - There are others but we will be focusing on these

Main SQL Commands: GROUP BY

- Syntax
- Example
- SELECT MIN(birth_date) as min_birth_date,
- MAX(birth_date) as max_birth_date,
- COUNT(*) as number_of_people
- FROM people
- GROUP BY favorite_color_id;
- Gets me earliest birth date, latest birth date, and number of people for each favorite color

Main SQL Commands: GROUP BY

- GROUP BY Exercises

Main SQL Commands: HAVING

- HAVING allows you to filter our which groups you want to show after a group by
- It’s like a WHERE clause for the group but same syntax with ONE notable difference
- WHERE clauses - have to use column names in table
- HAVING clauses - you can use the name you gave it (after the AS)
- Examples on next slide

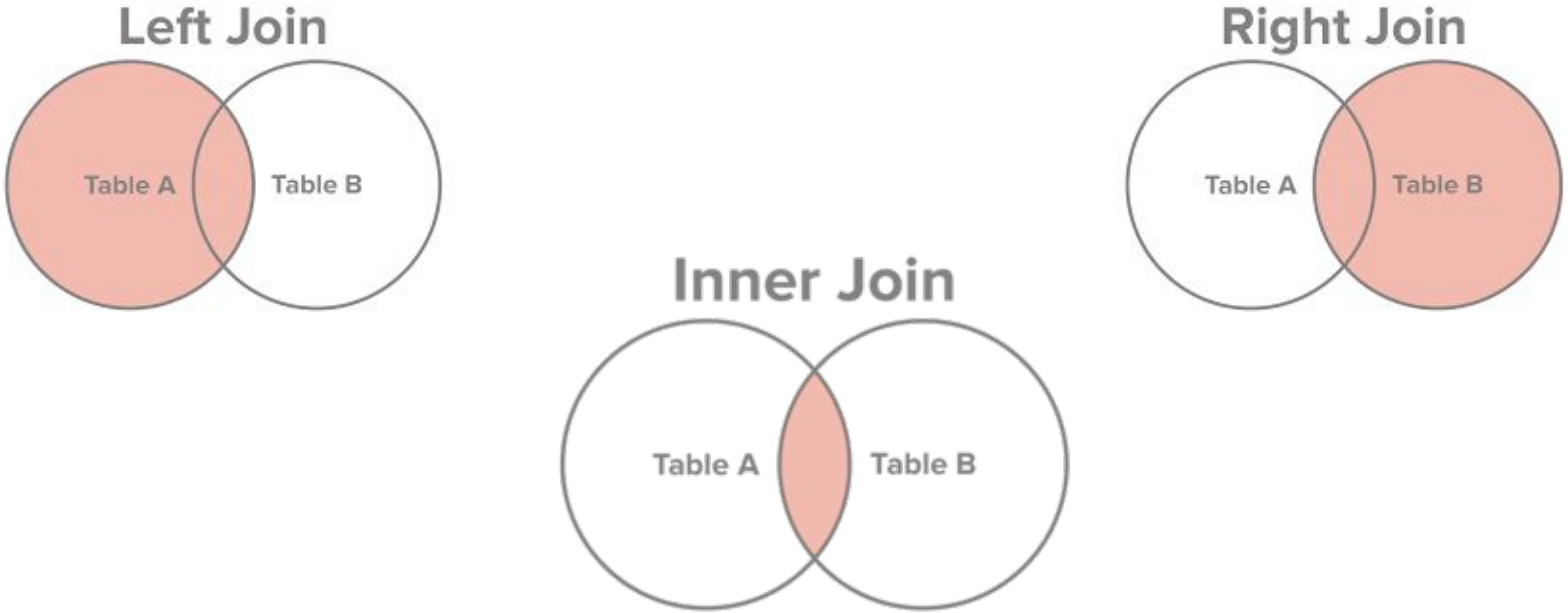
Main SQL Commands: HAVING

- Let’s say I only want to see colors that have 50 people
- SELECT MIN(birth_date) as min_birth_date,
- MAX(birth_date) as max_birth_date,
- COUNT(*) as number_of_people
- FROM people
- GROUP BY favorite_color_id
- HAVING number_of_people >= 50;
- Allows you to use aliased name (number_of_people) as opposed to
- COUNT(*), though you could use this too

Main SQL Commands: HAVING

- Exercises

Main SQL Commands: JOINS



Source: sql-join

Main SQL Commands: JOINS

- Allow you to join multiple tables together
- Ideally you want to join on a number not string (faster typically)
- The join statement comes after the FROM and before the WHERE
- Using our tables:
  - SELECT *
  - FROM people p
  - INNER JOIN color c
  - ON p.favorite_color_id = c.id
- INNER JOIN tells you I want rows that match both tables

Main SQL Commands: JOINS

- Exercises

Even More SQL Commands + Functions

1. DATE_FORMAT("2017-06-15", "%Y-%m") = “2017-06”
2. Boolean Column SELECT age > 25 as gt_25 (produces 0 / 1 column)
3. IF(age BETWEEN 13 AND 19, ‘teenager’, ‘non-teenager’) -> string variable
4. DATEDIFF(date_1, date_2) - produces number of days between date_1 and date_2

Exercises

1. Count the number of people by birth -year month (ex Jan 2017, Jan 2019)
2. Generate Age column (in years)
3. Generate a column where it’s string and “can vote” if person can vote based on birth date, else “cannot vote” - using if statements

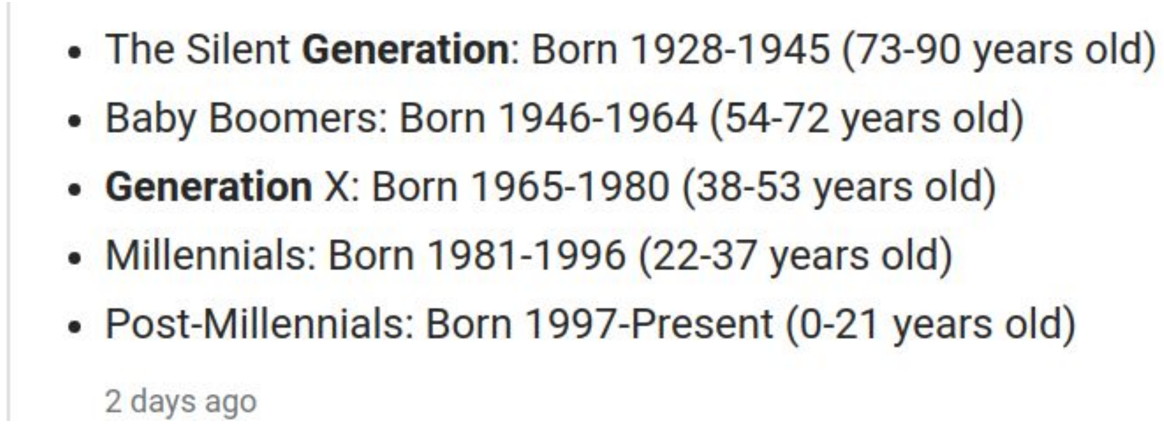
Even More SQL Commands + Functions

1. CASE STATEMENTS

```
 CASE
WHEN dt > ‘2019-06-01’ THEN ‘after_experiment’
 WHEN dt = ‘2019-06-01’ THEN ‘on_experiment’
ELSE ‘before_experiment’
END as experimental_group
```

Exercises

1. Generate a column that contains which generation the person belongs to



- The Silent **Generation**: Born 1928-1945 (73-90 years old)
- Baby Boomers: Born 1946-1964 (54-72 years old)
- **Generation** X: Born 1965-1980 (38-53 years old)
- Millennials: Born 1981-1996 (22-37 years old)
- Post-Millennials: Born 1997-Present (0-21 years old)

2 days ago

Case Statements for Pivot Tables - Rows to Columns

1. Produce there two different Views
   a. One Row per Year-favorite color

Year	Favorite Color	Number of people

2. One Row per year

Year	favorite_color_green	favorite_color_blue	favorite_color_red	favorite_color_yellow

Next Steps

- Start using SQL + Practicing
- I like https://sqlzoo.net/
- Once you get the hand of querying anything you want the next step is query optimization + making them fast