Introduction to SQL

BA770 Lab Session

Questrom School of Business, Boston University

July 29, 2019

Keywords and Functions

- SELECT
- FROM
- DISTINCT
- COUNT
- WHERE
- AND/OR/NOT
- BETWEEN
- IN
- IS (NOT) NULL
- (NOT) LIKE
- MAX/MIN/SUM/AVG
- AS
- ORDER BY (DESC)
- GROUP BY
- HAVING
- LIMIT

Selecting Columns: SELECT, FROM

- **SELECT** expression
  SELECT 'SQL'
- **SELECT** arithmetic operation
  SELECT 4/3
- **SELECT** column1, column2, ...
  FROM table
  SELECT id, year
  FROM people

Selecting Columns: DISTINCT, COUNT

- **SELECT DISTINCT** column
  FROM table
  SELECT DISTINCT year
  FROM people

- **SELECT COUNT** (column)
  FROM table
  SELECT COUNT (*)
  FROM people

- **SELECT COUNT** (**DISTINCT** column)
  FROM table
  SELECT COUNT (**DISTINCT** year)
  FROM people

Filtering Rows: WHERE, AND/OR/NOT

- **SELECT** column **FROM** table
  WHERE condition
  SELECT id **FROM** people
  WHERE age >= 50

- **SELECT** column **FROM** table
  WHERE conditions
  SELECT id **FROM** people
  WHERE (year = 2010 **OR** year = 2015)
  AND (age <> 40)

- Wrap conditions properly with parentheses.
- Do not forget to quote text values.

Filtering Rows: BETWEEN/IN

- **SELECT** column **FROM** table
  WHERE expression **BETWEEN** num1 **AND** num2
  SELECT id **FROM** people
  WHERE age **BETWEEN** 30 **AND** 35

- **SELECT** column **FROM** table
  WHERE expression **IN** (list_of_numbers)
  SELECT id **FROM** people
  WHERE age **IN** (30, 31, 32, 33, 34, 35)

- **BETWEEN**...**AND** is inclusive.
- You can negate the above result by prefacing **BETWEEN** or **IN** with **NOT** to exclude specified values.

Filtering Rows: IS (NOT) NULL, (NOT) LIKE

- **SELECT** column **FROM** table
  WHERE column **IS (NOT) NULL**
  SELECT id **FROM** people
  WHERE birthdate **IS NOT NULL**

- **SELECT** column **FROM** table
  WHERE column **(NOT) LIKE** 'pattern'
  SELECT id **FROM** people
  WHERE name **LIKE** 'B%'

- The '_' wildcard matches a single character.
- The '%' wildcard matches zero, one, or many characters.
- A missing value is not necessarily a **NULL** value. For example, some databases use extreme values like 0 or 99.9 to denote 'missingness'. Refer to database descriptions and properly handle missing values.

Aggregation Functions: MAX/MIN/SUM/AVG, AS

- **SELECT MAX/MIN/SUM/AVG** (column)
  FROM table
  SELECT AVG (age)
  FROM people

- **SELECT** expression **AS** new_name
  FROM table
  SELECT MAX(age)-**MIN**(age) **AS** age_range
  FROM people

- Aliases are helpful for making results more readable.

Sorting: ORDER BY, DESC

- **SELECT** column **FROM** table
  ORDER BY column (**DESC**)
  SELECT name **FROM** people
  ORDER BY age

- **SELECT** column **FROM** table
  ORDER BY column1, column2, ... (**DESC**)
  SELECT name **FROM** people
  ORDER BY age, name

- **ORDER BY** works for both numbers and text values.
- Default sorting order: ascending.
- Given multiple sorting columns, sort by the first, then the next, and so on.

Grouping: GROUP BY

- **SELECT** column **FROM** table
  GROUP BY column
  SELECT age, **COUNT**(*) **FROM** people
  GROUP BY age

- **SELECT** column **FROM** table
  GROUP BY column
  ORDER BY column
  SELECT age, **COUNT**(*) **AS** count **FROM** people
  GROUP BY age
  ORDER BY count **DESC**

- **GROUP BY** works with aggregation functions.
- **GROUP BY** always go after **FROM** and before **ORDER BY**.
- Given multiple grouping columns, group by the first, then the next, and so on.

Filtering Based on Aggregation Results: HAVING

- **SELECT** column **FROM** table
  GROUP BY column
  HAVING condition
  SELECT division, **AVG**(age) **FROM** people
  GROUP BY division
  HAVING **AVG**(age) $<$ 30
- **HAVING** always go after **GROUP BY**.
- **WHERE** vs **HAVING**:
  WHERE filters rows before any grouping occurs.
  HAVING filters on the groups or filter using aggregate values.

Limiting the Number of Rows Returned: LIMIT

- **SELECT** column
  FROM table
  WHERE condition
  GROUP BY column
  HAVING condition
  ORDER BY column
  LIMIT integer

A Short Note

- SQL Server functions (mentioned in DataCamp courses) that are not supported in BigQuery:
  TOP, **PERCENT**, **LEN**, **LEFT**, **RIGHT**, **CHARINDEX**, **SUBSTRING**, **REPLACE**