SQL

Select and Where Statements

* Basic Syntax Example:
  + SELECT column 1, column 2
  + FROM table name
  + WHERE conditions;
* The WHERE clause comes directly after FROM statement.
* Operators for comparison are the same as in math.
  + <> or != not equal to.
  + AND, OR, and NOT can be added to make a condition stronger.

A customer forgot their wallet at the store. Have to track them down with their email. What is the email for Nancy Thomas?

* Syntax:
  + SELECT email
  + FROM customer
  + WHERE first\_name = ‘Nancy’ AND last\_name = ‘Thomas’

Order By

* ORDER BY is used to sort rows based on a column value in ascending or descending order.
* Basic Syntax:
  + SELECT column\_1, column\_2
  + FROM table name
  + ORDER BY column\_1 ASC/DESC;

Limit

* Limits the number of rows returned.
* Goes at the very end of the query request.
* Should be used in conjunction with ORDER BY.
* Syntax:
  + SELECT \*
  + FROM payment
  + WHERE amount != 0
  + ORDER BY payment\_date DESC
  + LIMIT 5;

Between

* The BETWEEN operator can be used to match a value against a range of values.
  + Value BETWEEN low AND high
* You can also use the NOT BETWEEN operator.
  + Value NOT BETWEEN low AND high (value < low OR value > high).
* The BETWEEN operator for dates must be in the standard ISO format.
  + Date BETWEEN ‘2007-01-01’ AND ‘2007-02-01’ (YYYY-MM-DD).
* Syntax:
  + SELECT \*
  + FROM payment
  + WHERE payment\_date BETWEEN '2007-02-01' AND '2007-02-14';

IN

* The IN operator is used to see the rows within distinct values from a specific column.
* Numbers or integers do not need parenthesis.
* You can also use NOT IN to find all of the rows without the values listed.
  + SELECT \*
  + FROM customer
  + WHERE first\_name IN ('John', 'Jake', 'Julie', 'Dan', 'Steve');

LIKE and ILIKE

* The LIKE operator allows us to perform pattern matching against string data with the use of **wildcard** characters.
  + Percent %
    - Matches any sequence of characters.
  + Underscore \_
    - Matches any single character.
* All names that begin with an ‘A’:
  + WHERE name LIKE ‘A%’
* All names that end with an ‘a’:
  + WHERE name LIKE ‘%a’
* LIKE is case-sensitive.
* Using the underscore allows us to replace just a single character.
* Get all Mission Impossible films:
  + WHERE title LIKE ‘Mission Impossible \_’

[www.postgresql.org/docs/12/functions-matching.html](http://www.postgresql.org/docs/12/functions-matching.html)

* ILIKE is case-insensitive
  + SELECT \*
  + FROM customer
  + WHERE first\_name ILIKE 'j%' AND last\_name LIKE 'S%';
* Finding words with certain letters in them:
  + SELECT \*
  + FROM customer
  + WHERE first\_name LIKE '%er%';
* One character before shows you only want one letter before
  + LIKE ‘\_her%’