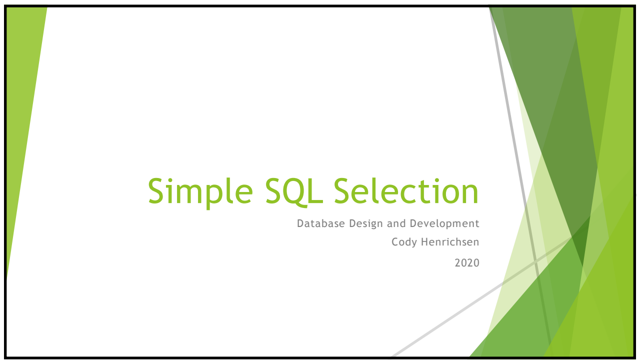
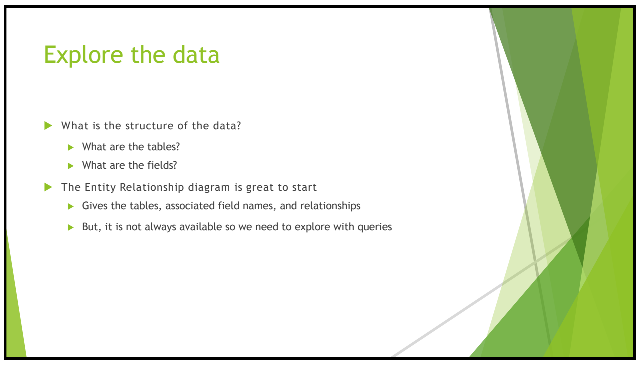


Simple SQL Selection

Database Design and Development
Cody Henrichsen
2020

Extracting data!

- The key questions about a database
  - What do we need?
  - Where is the data?
  - Is there certain data we care about?
  - Do we need to organize it?



Explore the data

- What is the structure of the data?
  - What are the tables?
  - What are the fields?
- The Entity Relationship diagram is great to start
  - Gives the tables, associated field names, and relationships
  - But, it is not always available so we need to explore with queries

Exploration Query

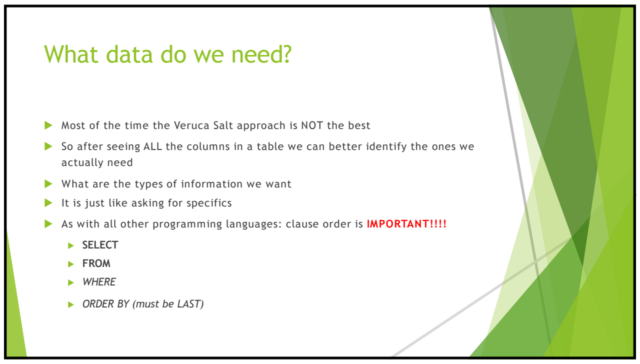
- We don't know what is inside the database so lets take a quick look at all the tables
- We will use a wildcard to get EVERYTHING!!!
- The SELECT keyword tells the database to get stuff
- The asterisk says I want it all!
- The FROM keyword tells the DB which table

SELECT *

- The wildcard selection gives us EVERYTHING about the table
- The metadata is the column headers and should describe what is stored inside the table
- Keep track of the column names as well as the type of information displayed, it will be useful in the future

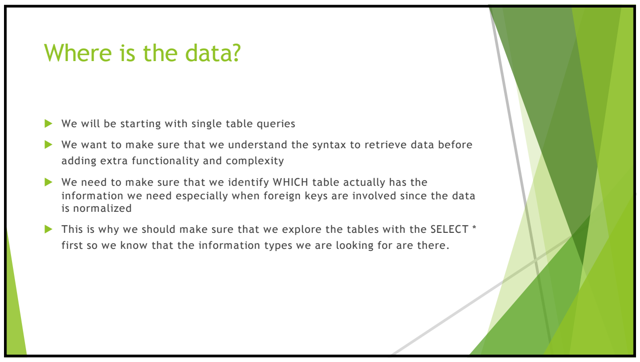
Get it ALL

```
--Get EVERYTHING!!
SELECT
    *
FROM
    table_name;
```



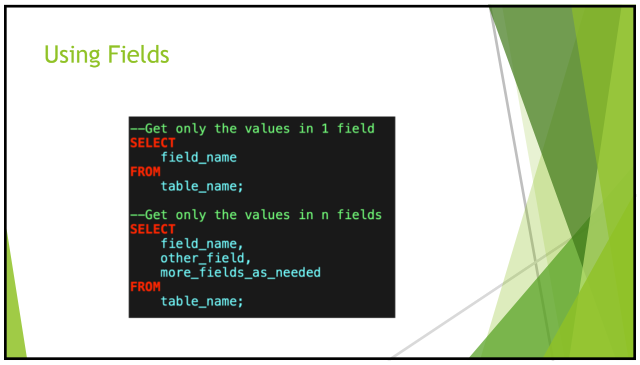
What data do we need?

- Most of the time the Veruca Salt approach is NOT the best
- So after seeing ALL the columns in a table we can better identify the ones we actually need
- What are the types of information we want
- It is just like asking for specifics
- As with all other programming languages: clause order is **IMPORTANT!!!!**
  - SELECT
  - FROM
  - *WHERE*
  - *ORDER BY (must be LAST)*



Where is the data?

- We will be starting with single table queries
- We want to make sure that we understand the syntax to retrieve data before adding extra functionality and complexity
- We need to make sure that we identify WHICH table actually has the information we need especially when foreign keys are involved since the data is normalized
- This is why we should make sure that we explore the tables with the SELECT * first so we know that the information types we are looking for are there.



Using Fields

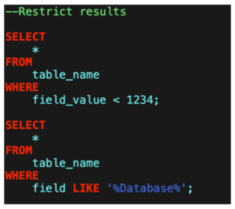
```
--Get only the values in 1 field
SELECT
    field_name
FROM
    table_name;

--Get only the values in n fields
SELECT
    field_name,
    other_field,
    more_fields_as_needed
FROM
    table_name;
```

Do we REALLY need all of this data?

- Often, the amount of data is excessive
- There are just TOO MANY ROWS!
- So we need to narrow our focus
- This compares to the if block for SQL searches
- We will OFTEN use the wildcard and relational operators
  - %, _
  - <, >, =, <=, !=, >=

Using WHERE

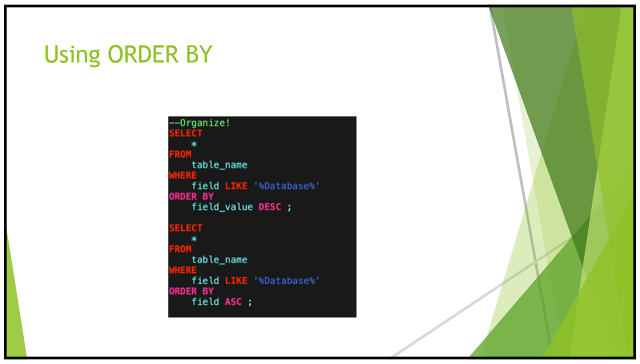


```
--Restrict results
SELECT
    *
FROM
    table_name
WHERE
    field_value < 1234;

SELECT
    *
FROM
    table_name
WHERE
    field LIKE '%Database%';
```

Organization??

- Relational data is UNSORTED by default
- The sorting command is ORDER BY
- You can sort by fields that are not visible!
- Organize forward with ASC (default), or backwards with DESC
- Compound sorts separated by commas



Using ORDER BY

```
--Organize!
SELECT
    *
FROM
    table_name
WHERE
    field LIKE '%Database%'
ORDER BY
    field_value DESC ;

SELECT
    *
FROM
    table_name
WHERE
    field LIKE '%Database%'
ORDER BY
    field ASC ;
```