Codeup Bayes SQL Lunch n’ Learn

Presented by Kevin Eliasen

SELECT
FROM
JOIN
ON
WHERE
GROUP BY
HAVING
ORDER BY
LIMIT
OFFSET

SQL Key Concepts

Everything is a table

Databases keep everything in tables - even the tables. SQL views everything in terms of structured rows and columns. Think of every step of the querying process as producing more tables.

Indexes are Key

Indexes require more overhead at the point of data entry, but the speed up data retrieval.

Write for readability

SQL is whitespace-friendly, so take advantage of it. We’re not charged by the character or line. The time saved by writing neatly and consistently more than makes up for the time spent trying to decipher cramped code.

SELECT
FROM
JOIN
ON
WHERE
GROUP BY
HAVING
ORDER BY
LIMIT
OFFSET

FROM - JOIN - ON

- Combines all tabular data sources into a single tabular dataset
  - Tables
  - Views (pre-saved queries)
  - Subqueries)
- All datasets must be named; dataset aliases defined here
- Establishes the dimension of the combined dataset
  - Dataset column count = Sum of column counts from each source
  - Dataset row count = Product of row counts from each source
- JOIN statements define relationships amongst source datasets
- ON statements establish top-level filter conditions which determine final the length of the combined dataset

SELECT

FROM

JOIN

ON

WHERE

GROUP BY

HAVING

ORDER BY

LIMIT

OFFSET

SELECT … WHERE

- SELECT identifies the width of the return dataset
  - Columns must be explicitly identified
  - "*" selects all
  - All columns are named, column aliases are defined here
- Calculations identified here
  - Calculations *within the row* can be returned directly
  - Calculations *across multiple rows* are defined here, but not returned in this stage
- WHERE statements establish base-level filtering criteria to reduce the number of rows returned
  - Filtering result of calculation requires stating the calculation in the WHERE clause *even if the calculation is also in SELECT*
  - IN statements augment filtering options

SELECT
FROM
JOIN
ON
WHERE
GROUP BY
HAVING
ORDER BY
LIMIT
OFFSET

GROUP BY - HAVING

- GROUP BY statements define the unique values (or combination of values) that form the basis for computations
- Computations performed on all rows that have the same unique value
- Unique-value fields do not need to be displayed
- HAVING statements filter based on results of aggregate calculations
  - Filtering result of calculation requires stating the calculation in the HAVING clause *even if the calculation is also in SELECT*
  - IN statements augment filtering options

SELECT
FROM
JOIN
ON
WHERE
GROUP BY
HAVING
ORDER BY
LIMIT
OFFSET

ORDER BY

- Runs on cumulative product of FROM, SELECT, and GROUP BY outputs
- Can be applied to multiple columns, ascending or descending
- Does not affect row count

SELECT
FROM
JOIN
ON
WHERE
GROUP BY
HAVING
ORDER BY
LIMIT
OFFSET

LIMIT … OFFSET

- LIMIT places a final cap on the length of the return dataset
- OFFSET shifts the starting point of the return dataset
- Most often used with ORDER BY statements
- LIMIT and OFFSET can be combined into a single statement “LIMIT ##, ##” where the first number is the offset and the second is the limit

SQL Command Order and Impact

SELECT

FROM

JOIN

ON

WHERE

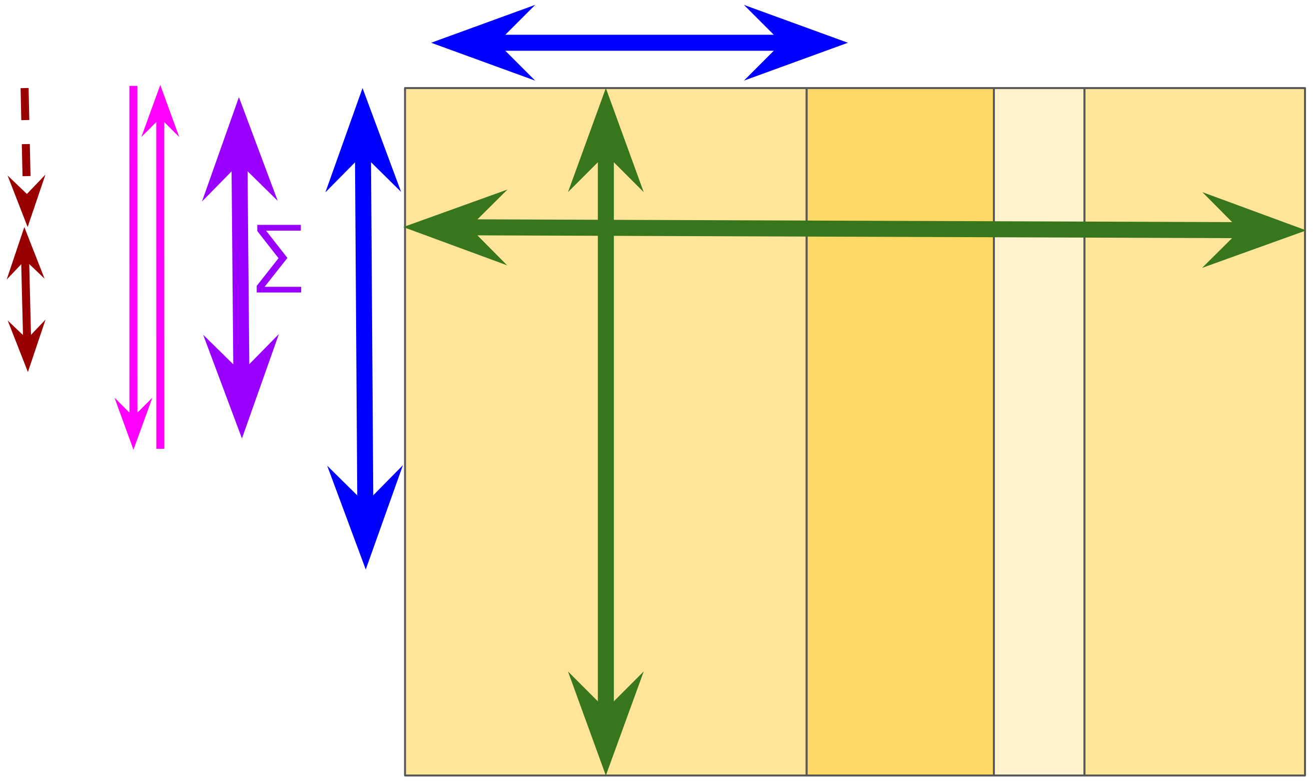
GROUP BY

HAVING

ORDER BY

LIMIT

OFFSET



Aliases and Whitespace

Before you can fix any error, you first need to find it

These two SQL queries perform are equally valid for the computer. The query to the right, however, uses simple aliases and whitespace to enhance readability for the human.

```
SELECT rental.rental_id rentid,
payment.payment_id payid, rental.staff_id
rstaff, payment.staff_id pstaff  FROM rental
JOIN payment ON payment.rental_id =
rental.rental_id WHERE payment.staff_id <>
rental.staff_id;
```

```
SELECT
    r.rental_id rentid
    ,p.payment_id payid
    ,r.staff_id rstaff
    ,p.staff_id pstaff
FROM
    rental r
JOIN
    payment p
    USING(rental_id)
WHERE
    p.staff_id <> r.staff_id;
```