CSCI 403
Database Management

11 – Grouping & Aggregation

CS@Mines

AGGREGATE FUNCTIONS

COUNT

One of the most heavily used functions…

How many rows are in a table?

```
SELECT COUNT(*) FROM tablename;
```

How many rows match a condition?

```
SELECT COUNT(*) FROM tablename
WHERE conditions;
```


COUNT is a Summary

- COUNT is an aggregate function
  - Doesn't act row-by-row
  - Gives a summary of all rows
- COUNT cannot be SELECTed with regular columns…
  - … would be somewhat meaningless…
  - Until we learn how to group!

```
SELECT COUNT(x), x FROM foo;
```

ERROR!

COUNT and NULL

You can count individual columns.
Note: NULL columns don't count!

```
SELECT COUNT(*) AS rows,
       COUNT(1) AS ones,
       COUNT(x) AS xes,
       COUNT(y) AS ys
FROM foo;
```

```
 rows | ones | xes | ys
------+------+-----+----
    5 |    5 |   4 |  3
```

foo

x	y
apple	42
banana	17
cherry	99
pear	<null>
<null>	<null>

Other Aggregate Functions

SUM – adds up non-NULL numeric values
MAX – finds maximum non-NULL entry
MIN – finds minimum non-NULL entry
AVG – calculates the average of non-NULL numbers
STDDEV_SAMP – Sample standard deviation
…
Note these apply to expressions, not rows!

See https://www.postgresql.org/docs/9.5/static/functions-aggregate.html for more!

GROUPING

Grouping

- Compute aggregates on *subsets* of rows
  - Rows organized by equal values of subset of columns
  - Organizing columns listed in GROUP BY clause
  - Aggregates and grouping columns only in SELECT
- Example:

```
SELECT instructor, count(*)
FROM mines_courses
GROUP BY instructor
ORDER BY count(*) DESC;
```


How To Think About Grouping

Suppose you group by columns c1 & c2:

- Find all unique combinations of c1 & c2
- Put all rows matching each unique combination into their own group
- Compute aggregate functions on each group
- Return results for each group

Example

```
SELECT
  x, count(y), sum(y), avg(y)
FROM
  baz
GROUP BY x;
```

```
   x    | count | sum |         avg
--------+-------+-----+---------------------
 apple  |     3 |  60 |  20.0000000000000000
 cherry |     2 | 200 | 100.0000000000000000
 banana |     1 |  17 |  17.0000000000000000
```

baz

x	y
apple	10
apple	20
apple	30
banana	<null>
banana	17
cherry	99
cherry	101

Grouping and Ordering

- You can combine ordering with grouping:
  - ORDER BY always comes at the end of the query
  - ORDER BY any aggregates or grouping columns/expressions (even if not SELECTed)
- Example:

```
SELECT
  substr(course_id, 1, 4) AS subject, count(*)
FROM mines_courses
GROUP BY subject
ORDER BY avg(length(title)) DESC, subject;
```


Filtering and Grouping

- WHERE clause applies before grouping
  - Filters rows only on expressions/columns
  - Cannot filter on aggregate functions (not yet computed!)
- HAVING clause applies *after* grouping
  - Filters group results
  - Can filter on aggregate functions (or expressions/columns)

Example

```
SELECT x, sum(y)
FROM baz
WHERE y < 25
GROUP BY x
HAVING sum(y) > 20;
```

baz

x	y
apple	10
apple	20
apple	30
banana	<null>
banana	17
cherry	99
cherry	101

Example, Dissected

```
SELECT x, y FROM baz
WHERE y < 25 …
   x    | y
--------+----
 banana | 17
 apple  | 20
 apple  | 10

SELECT x, sum(y) … GROUP BY x …
   x    | sum
--------+-----
 apple  |  30
 banana |  17

… HAVING sum(y) > 20
   x    | sum
--------+-----
 apple  |  30
```

baz

x	y
apple	10
apple	20
apple	30
banana	<null>
banana	17
cherry	99
cherry	101

Grouping and Subqueries

Aggregates *cannot* appear in WHERE clause!

This means you can't do something like:

```
SELECT * FROM person
WHERE salary = MAX(salary);
```

Use a subquery instead!

```
SELECT * FROM person
WHERE salary =
  (SELECT MAX(salary) FROM person);
```


Final Example

What MWF timeslots have the most courses?

```
SELECT begin_time, COUNT(*)
FROM mines_courses_meetings
WHERE days = 'MWF'
GROUP BY begin_time
ORDER BY COUNT(*) DESC;
```


Up Next

- Next lecture:
  Miscellaneous topics.