Grouping and Aggregation
PDBM 7.3–7.3.1.5

Dr. Chris Mayfield

Department of Computer Science
James Madison University

Feb 01, 2019

<table>
<thead>
<tr>
<th>state</th>
<th>city</th>
<th>revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maharashtra</td>
<td>Achalpur</td>
<td>10000</td>
</tr>
<tr>
<td>Uttar Pradesh</td>
<td>Achnera</td>
<td>20000</td>
</tr>
<tr>
<td>Gujarat</td>
<td>Adala</td>
<td>10000</td>
</tr>
<tr>
<td>Uttar Pradesh</td>
<td>Agra</td>
<td>30000</td>
</tr>
<tr>
<td>Gujarat</td>
<td>Ahmedabad</td>
<td>40000</td>
</tr>
<tr>
<td>Maharashtra</td>
<td>Ahmednagar</td>
<td>10000</td>
</tr>
<tr>
<td>Maharashtra</td>
<td>Akola</td>
<td>40000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>state</th>
<th>sum(revenue)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maharashtra</td>
<td>60000</td>
</tr>
<tr>
<td>Uttar Pradesh</td>
<td>50000</td>
</tr>
<tr>
<td>Gujarat</td>
<td>50000</td>
</tr>
</tbody>
</table>
Aggregation

Common functions

- COUNT, SUM, AVG, MIN, MAX

```sql
SELECT min(year), max(year), count(year)
FROM movie
WHERE title = 'Frozen';
```

More commonly used in groups:

```sql
SELECT year, count(*)
FROM movie
WHERE title = 'Frozen'
GROUP BY year
ORDER BY year;
```
GROUP BY

<table>
<thead>
<tr>
<th>state</th>
<th>city</th>
<th>revenue</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maharashtra</td>
<td>Achalpur</td>
<td>10000</td>
</tr>
<tr>
<td>Uttar Pradesh</td>
<td>Achhnera</td>
<td>20000</td>
</tr>
<tr>
<td>Gujarat</td>
<td>Adalaj</td>
<td>10000</td>
</tr>
<tr>
<td>Uttar Pradesh</td>
<td>Agra</td>
<td>30000</td>
</tr>
<tr>
<td>Gujarat</td>
<td>Ahmedabad</td>
<td>40000</td>
</tr>
<tr>
<td>Maharashtra</td>
<td>Ahmednagar</td>
<td>10000</td>
</tr>
<tr>
<td>Maharashtra</td>
<td>Akola</td>
<td>40000</td>
</tr>
</tbody>
</table>

GROUP BY state

<table>
<thead>
<tr>
<th>state</th>
<th>sum(revenue)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maharashtra</td>
<td>60000</td>
</tr>
<tr>
<td>Uttar Pradesh</td>
<td>50000</td>
</tr>
<tr>
<td>Gujarat</td>
<td>50000</td>
</tr>
</tbody>
</table>

©graspqsql.com
HAVING vs WHERE

- **WHERE** = before grouping
- **HAVING** = after grouping

-- titles used more than 50 times
SELECT m.title, k.kind, count(*)
FROM movie AS m
  JOIN kind_type AS k ON m.kind_id = k.id
WHERE year > 2010
  AND m.title NOT LIKE '(%'
GROUP BY m.title, k.kind
HAVING count(*) > 50
ORDER BY count DESC
Aggregation with NULLs

Three ways of counting:

- SELECT count(*) FROM StarsIn; -- count rows
- SELECT count(starName) FROM StarsIn; -- count values
- SELECT count(DISTINCT starName) FROM StarsIn;

NULL values are ignored!

- Does not consider movies with NULL length
- However, count(*) of an empty table is 0

```sql
SELECT m.title, m.year, m.kind_id, -- see GROUP BY
       max(r.info) AS max_runtime, count(*)
FROM movie AS m LEFT JOIN movie_info AS r
  ON m.id = r.movie_id AND r.info_id = 1
GROUP BY m.title, m.year, m.kind_id
```
NULLs can be confusing!

<table>
<thead>
<tr>
<th>a</th>
<th>b</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL</td>
<td>NULL</td>
</tr>
</tbody>
</table>

```sql
SELECT a, count(b)
FROM R
GROUP BY a;
```

- Returns (NULL, 0)

```sql
SELECT a, sum(b)
FROM R
GROUP BY a;
```

- Returns (NULL, NULL)
SQL Exercises: imdb

How many of each kind of movie?
How many of each type of info?
What else would you like to know?
Hello TPC-H

The Transaction Processing Performance Council is a non-profit corporation that defines performance benchmarks for comparing database systems.

- All the major database companies from industry
- Founded in 1988; new benchmarks every few years

TPC-H is a business-oriented “decision support” database consisting of customers, orders, parts, suppliers, etc.

- The data is synthetically generated
- A driver simulates many transactions