Lecture 20:
  • Grouping

Announcements:
  • HW-5 due
  • Project Step 3 due Tues, 11/15
  • HW-6 out soon
  • PS-4 out soon
  • Exam 2 next Thurs (11/17)

SQL Group By

The `GROUP BY` clause partitions tables into multiple “groups”

For example:

```
SELECT owner, AVG(balance)
FROM account
```

• is not well formed (even though MySQL returns an answer ...)

Whereas this query is legal:

```
SELECT owner, AVG(balance)
FROM account
GROUP BY owner
```

• each owner value forms a “group”
• the average balance is computed over each group
SQL Group By

Consider the following schema and instance

Customer(c_num, name, addr, c_rating, c_amount, c_bal, sp_num)
Salesperson(sp_num, name, address, office)

with FK: customer.sp_num → salesperson.sp_num

And Customer instance

<table>
<thead>
<tr>
<th>c_num</th>
<th>name</th>
<th>address</th>
<th>c_rating</th>
<th>c_amount</th>
<th>c_balance</th>
<th>sp_num</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Alice</td>
<td>xxx</td>
<td>5</td>
<td>1000</td>
<td>1000</td>
<td>101</td>
</tr>
<tr>
<td>2</td>
<td>Bob</td>
<td>yyy</td>
<td>7</td>
<td>5000</td>
<td>4000</td>
<td>101</td>
</tr>
<tr>
<td>3</td>
<td>Chuck</td>
<td>zzz</td>
<td>10</td>
<td>10000</td>
<td>1000</td>
<td>102</td>
</tr>
</tbody>
</table>

Check in: What does this query return? ... note 2 groups!

```sql
SELECT sp_num, COUNT(*)
FROM Customer
WHERE c_rating >= 5
GROUP BY sp_num;
```

Group By Example

A more involved GROUP BY example

<table>
<thead>
<tr>
<th>cust_num</th>
<th>name</th>
<th>...</th>
<th>sp_num</th>
</tr>
</thead>
<tbody>
<tr>
<td>101</td>
<td>alice</td>
<td>...</td>
<td>5</td>
</tr>
<tr>
<td>102</td>
<td>chuck</td>
<td>...</td>
<td>8</td>
</tr>
<tr>
<td>103</td>
<td>bob</td>
<td>...</td>
<td>NULL</td>
</tr>
<tr>
<td>106</td>
<td>gus</td>
<td>...</td>
<td>5</td>
</tr>
<tr>
<td>107</td>
<td>debbie</td>
<td>...</td>
<td>5</td>
</tr>
<tr>
<td>109</td>
<td>eddy</td>
<td>...</td>
<td>2</td>
</tr>
<tr>
<td>110</td>
<td>fionna</td>
<td>...</td>
<td>8</td>
</tr>
</tbody>
</table>

```sql
SELECT sp_num, COUNT(*)
FROM customer
GROUP BY sp_num;
```
Group By Example

The intermediate result includes 4 groups

<p>| | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>103</td>
<td>bob</td>
<td>...</td>
<td>NULL</td>
</tr>
<tr>
<td>101</td>
<td>alice</td>
<td>...</td>
<td>5</td>
</tr>
<tr>
<td>106</td>
<td>gus</td>
<td>...</td>
<td>5</td>
</tr>
<tr>
<td>107</td>
<td>debbie</td>
<td>...</td>
<td>5</td>
</tr>
<tr>
<td>109</td>
<td>eddy</td>
<td>...</td>
<td>2</td>
</tr>
<tr>
<td>102</td>
<td>chuck</td>
<td>...</td>
<td>8</td>
</tr>
<tr>
<td>110</td>
<td>fionna</td>
<td>...</td>
<td>8</td>
</tr>
</tbody>
</table>

With “SELECT sp_num, COUNT(*)” evaluated for each group

<table>
<thead>
<tr>
<th>salesperson</th>
<th>COUNT(*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NULL</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>8</td>
<td>2</td>
</tr>
</tbody>
</table>

Group By

Check In: Find the average customer credit amount and credit balance by credit rating.

```
SELECT c_rating, AVG(c_amount), AVG(c_balance)
FROM customer
GROUP BY c_rating
```

Check In: For each sales office find the corresponding average credit balance of customers whose salesperson is located in the office.

```
SELECT office, AVG(c_balance)
FROM salesperson JOIN customer USING (sp_num)
GROUP BY office
```
Group By

**Check In:** Find the minimum credit rating of each salesperson’s customers who have a credit amount over $1,000 and who work in the “Spokane” office.

```sql
SELECT sp_num, MIN(c_rating)
FROM salesperson JOIN customer USING (sp_num)
WHERE c_amount > 1000 AND office = 'Spokane'
GROUP BY sp_num
```

---

**Group By**

When using GROUP BY, the SELECT clause can only contain:

- the grouping attributes (attributes in GROUP BY clause); or
- aggregate operators (which are applied to the group)

This query is not legal SQL ... but legal in MySQL

```sql
SELECT name FROM customer GROUP BY sp_num;
```

**Check In:** Why wouldn’t name be allowed in the SELECT here?

- there could potentially be many name values in each group

**Check In:** What if we group by a primary key though?

- then there could only one value for the other attributes
- e.g., SELECT name FROM customer GROUP BY c_num
- note: applies generally to functional dependencies (group by lhs)