Database Jour - Grouping

Goal for today: You'll continue filtering and sorting, and will also generate totals for groups from a database table to answer how many, how much and other questions.

Summary functions
SQL has a series of functions for generating summary totals. We'll start with sum(), max(), min(), avg(), count().

To get started:
use exercises;
We have another command to get started, which I will explain below:
```
SET SQL_MODE='only_full_group_by';
```
You can type it or copy-and-paste it. Make it a regular part of starting you a session in mysql workbench.

First just do a total.
```
Select sum(amount)
from softmoney100k;
```

That totals the amount.
You can give the output field a name with "as."
```
Select sum(amount) as total_amount
from softmoney100k;
```

Using other summary functions: min, max, avg.
```
Select min(amount) as min_amount
from softmoney100k;
```

```
Select max(amount) as max_amount
from softmoney100k;
```

```
Select avg(amount) as avg_amount
from softmoney100k;
```

You can use more than one at a time, separated with commas just like other fields.
```
Select min(amount) as min_amount, max(amount) as max_amount
from softmoney100k;
```

You can count how many there are. The technique shown here is NOT how we will do it in the end; but this is just for introducing the count() total.
```
Select count(amount) as count_contributions
from softmoney100k;
```

Grouping
A key step in generating totals is grouping. You can make a group to answer a question like "Which party received the most money?" We add another line to our SQL after the "from" – "group by".

```
Select party
from softmoney100k
group by party;
```

The “group by” line will create one record – one row out output – for each different value in the group field. So with party, it creates one row each for D, R and 3.

When grouping, we always need to have matching fields in the “Select” and “Group by” lines, such as “Select party” and “Group by party.” That's why we use the command
`SET SQL_MODE='only_full_group_by';`
That makes mysql require matching fields in the “Select” and “Group by” lines. Without matching fields, mysql will guess about which values to give you and may generate unreliable results.

To answer the question of which party got the most money, we group and use sum().

```
Select party, sum(amount) as total_amount
from softmoney100k
group by party;
```

You can add a sort (order by) to show the biggest amount at the top.

```
Select party, sum(amount) as total_amount
from softmoney100k
group by party
order by total_amount desc;
```

We can answer how many contributions each party got with count().

```
Select party, count(party) as contributions
from softmoney100k
group by party
order by contributions desc;
```

But count() produces inaccurate results if there are missing values (null values). So when we count, we don't pick just one field to count. We say count the row itself by using count(*).

```
Select party, count(*) as contributions
from softmoney100k
group by party
order by contributions desc;
```

The count(*) technique is how we will always do counting.

We can combine the functions.

```
Select party, count(*) as contributions,
sum(amount) as total_amount,
min(amount) as min_amount,
max(amount) as max_amount,
avg(amount) as avg_amount
from softmoney100k
group by party
order by total_amount desc;
```

We can combine a where filter with grouping to answer a question like, which party got the most money from the Insurance industry. We'll using grouping, and a sum() and a where, and also a sort (order by).

```
select party, sum(amount) as total
from softmoney100k
where industry = 'Insurance'
group by party
order by total desc;
```

The order of this syntax matters.
"Select" has to be first
"From" comes after "Select"
"Where" has to be immediately after "select"
"Group by" comes after "from" and "where"
"order by" is last.

```
Select ..
From …
Where ..
Group by …
Order by …
```

Another similar question. Which party got the most money from donors in Georgia or Florida?

```
select party, sum(amount) as total
from softmoney100k
where state in ('FL', 'GA')
group by party
order by total desc;
```

You can group by more than one thing. Let's practice grouping by party and sector to create summary total answering the question: How much did each sector give to each party?

```
Select party, sector, sum(amount) as total
From softmoney100k
Group by party, sector
Order by sector, total desc;
```

You can get results in different forms by changing the order of the grouping or the sorting, but the amount in each group will not change. You can also add in a count of contributions.

```
Select party, sector,
sum(amount) as total, count(*) as contributions,
From softmoney100k
Group by party, sector
Order by sector, total desc;
```

What if we did not want to bother seeing the third-party results in that query? We can use a where filter to specify only Democrats and Republicans.

```
Select party, sector,
sum(amount) as total, count(*) as contributions
From softmoney100k
Where party = 'D' or party = 'R'
Group by party, sector
Order by sector, total desc;
```

Can you think of another way of phrasing that “where” filter?