Aggregate Functions

- Operate on a column of a relation, and return a value
  - **avg**: average value
  - **min**: minimum value
  - **max**: maximum value
  - **sum**: sum of values
  - **count**: number of values
Example:
Find the average account balance at the Perryridge branch.

```
select avg(balance) 
from Account 
where branch-name="Perryridge"
```
Example with the ‘count’ Function

- Example: Find the number of tuples in the account relation.

  ```sql
  select count(*)
  from Account
  ```
  
  - Remember that ‘*’ stands for all attributes
  
  - Same as:

  ```sql
  select count(branch-name)
  from Account
  ```
  
  - Different from:

  ```sql
  select count(distinct branch-name)
  from Account
  ```
  
  - Because branch-name is not a key in Account

Account (account-number, balance, branch-name)
Example: Find the number of accounts for each branch.

```
select branch-name, count(account-number)
from Account
group by branch-name
```

- For each group of tuples with the same branch-name, count the account numbers for this group.
• Attributes in the `select` clause outside of aggregate functions must appear in the `group by` list. (why?)

```sql
select branch-name, balance, count(distinct account-number)  
from Account  
group by branch-name
```

Account (account-number, balance, branch-name)
Example: Find the number of depositors for each branch.

```sql
select branch-name, count(distinct customer-name)
from Depositor, Account
where Depositor.account-number = Account.account-number
group by branch-name
```

- Perform join, then group by, then `count(distinct )`

- Group by and aggregate functions apply to the join result
```sql
SELECT branch-name, COUNT(DISTINCT customer-name)
FROM Depositor, Account
WHERE Depositor.account-number = Account.account-number
GROUP BY branch-name
```

### Table: Depositor and Account

<table>
<thead>
<tr>
<th>branch-name</th>
<th>cust-name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perryridge</td>
<td>John Wong</td>
</tr>
<tr>
<td>Perryridge</td>
<td>Jacky Chan</td>
</tr>
<tr>
<td>Uptown</td>
<td>John Wong</td>
</tr>
<tr>
<td>Uptown</td>
<td>Mary Kwan</td>
</tr>
<tr>
<td>Downtown</td>
<td>John Wong</td>
</tr>
<tr>
<td>Downtown</td>
<td>Pat Lee</td>
</tr>
<tr>
<td>Downtown</td>
<td>May Cheung</td>
</tr>
</tbody>
</table>

### Table: Depositor and Account (Count)

<table>
<thead>
<tr>
<th>branch-name</th>
<th>count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perryridge</td>
<td>2</td>
</tr>
<tr>
<td>Uptown</td>
<td>2</td>
</tr>
<tr>
<td>Downtown</td>
<td>3</td>
</tr>
</tbody>
</table>
The 'having' clause (condition on the groups)

- Example: Find the names and average of balances of all branches where the average account balance is more than $700

```sql
select branch-name, avg(balance)
from Account
group by branch-name
having avg(balance) > 700
```

- Predicates in the `having` clause are applied to each group after the formation of groups.

- What is the query result if the Account table is this

<table>
<thead>
<tr>
<th>branch-name</th>
<th>account-number</th>
<th>balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perryridge</td>
<td>a-102</td>
<td>400</td>
</tr>
<tr>
<td>Perryridge</td>
<td>a-201</td>
<td>900</td>
</tr>
<tr>
<td>Brighton</td>
<td>a-217</td>
<td>750</td>
</tr>
<tr>
<td>Brighton</td>
<td>a-215</td>
<td>750</td>
</tr>
<tr>
<td>Redwood</td>
<td>a-222</td>
<td>700</td>
</tr>
</tbody>
</table>

Account (account-number, balance, branch-name)
Example: Display the names of all branches in Hong Kong where the average account balance is more than $700

```
select branch-name
from Account, Branch
where Account.branch-name=Branch.branch-name
    and branch-city="Hong Kong"
group by branch-name
having avg(balance) > 700
```

- First, you find the records that satisfy the `where` condition
- Then, you form the groups (including only the above records)
- Finally, you apply the `having` clause to each group
Example: Find the name(s) of branches with the maximum average account balance.

```
select branch-name 
from ( select branch-name, avg(balance) 
from Account 
group by branch-name) 
as Result(branch-name, avg-balance) 
where avg-balance = 
( select max(avg-balance) 
from Result) 
```

Account (account-number, balance, branch-name)