Aggregate Functions

Applied to one attribute in a multiset (SQL's name for query results)
Result is a single value

There are 5 aggregate functions in standard SQL:

- **Average:** AVG - Numeric attributes only
- **Total:** SUM - Numeric attributes only
- **Minimum:** MIN - any attributes
- **Maximum:** MAX - any attribute
- **# of tuples:** COUNT - any attribute

ex. Consider relation SHIPMENTS (shown in two columns to save space):

<table>
<thead>
<tr>
<th>S#</th>
<th>P#</th>
<th>QTY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1</td>
<td>P1</td>
<td>700</td>
</tr>
<tr>
<td>S1</td>
<td>P2</td>
<td>200</td>
</tr>
<tr>
<td>S2</td>
<td>P2</td>
<td>400</td>
</tr>
<tr>
<td>S3</td>
<td>P1</td>
<td>900</td>
</tr>
<tr>
<td>S3</td>
<td>P4</td>
<td>300</td>
</tr>
<tr>
<td>S4</td>
<td>P3</td>
<td>100</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>S#</th>
<th>P#</th>
<th>QTY</th>
</tr>
</thead>
<tbody>
<tr>
<td>S4</td>
<td>P1</td>
<td>800</td>
</tr>
<tr>
<td>S4</td>
<td>P3</td>
<td>500</td>
</tr>
<tr>
<td>S5</td>
<td>P1</td>
<td>50</td>
</tr>
<tr>
<td>S5</td>
<td>P3</td>
<td>150</td>
</tr>
<tr>
<td>S5</td>
<td>P4</td>
<td>250</td>
</tr>
<tr>
<td>S5</td>
<td>P2</td>
<td>350</td>
</tr>
</tbody>
</table>

SELECT COUNT(S#) AS NUM  
FROM SHIPMENTS  
NUM  
12

SELECT COUNT(DISTINCT S#) AS NUM  
FROM SHIPMENTS  
NUM  
5

SELECT COUNT(S#) AS NUM  
FROM SHIPMENTS  
NUM  
3  
WHERE P# = "P2"

SELECT SUM(QTY) AS TOT_P2  
FROM SHIPMENTS  
TOT_P2  
950  
WHERE P# = "P2"
A more useful operation would be to obtain the total (or average, or maximum, ...) quantity for all parts on order, in one query.

In effect, generating subtotals for each unique part #

SQL supports this feature, using the GROUP BY clause of the SELECT statement:

```
SELECT P#, SUM(QTY) AS TOT
FROM SHIPMENTS
GROUP BY P#
```

result has one tuple for each P#, with SUM(QTY) calculated for each group

<table>
<thead>
<tr>
<th>P#</th>
<th>TOT</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1</td>
<td>2450</td>
</tr>
<tr>
<td>P2</td>
<td>950</td>
</tr>
<tr>
<td>P3</td>
<td>750</td>
</tr>
<tr>
<td>P4</td>
<td>550</td>
</tr>
<tr>
<td>P5</td>
<td>750</td>
</tr>
</tbody>
</table>

How many suppliers do we have in each city? Use the SUPPLIERS relation:

```
SELECT CITY, COUNT(CITY)
FROM SUPPLIERS
GROUP BY CITY
```

```
SELECT CITY, COUNT(*)
FROM SUPPLIERS
GROUP BY CITY
```
When a SELECT statement includes a GROUP BY clause, all of the expressions in the attribute-list for the statement must be single-valued for each group, or be an aggregate function.

Implications of this: we can only include the grouping field, and any aggregate functions we are calculating, in a SELECT/GROUP BY stmt.

ex. Find the average weight of all parts stored in each city, in one query.

Use the PARTS relation: (P#, PNAME, COLOR, WEIGHT, LOCATION)

We can specify conditions that a group must pass before being included in a result with the HAVING clause of a SELECT statement

Note: if HAVING is used in a SELECT statement, GROUP BY must also be used in the SELECT statement!

ex. SELECT P#, COUNT(*) AS NUM FROM SHIPMENTS GROUP BY P# HAVING COUNT(*) > 2

ex. Find the total weight of all parts of any color EXCEPT Blue, in one query.

    SELECT COLOR, SUM(WEIGHT) FROM PARTS GROUP BY COLOR HAVING COLOR <> "BLUE"
There are two ways of obtaining aggregate statistics for one group:

ex. Find the maximum account balance for all accounts for fund # 0032:

```
SELECT FUND_NUM, MAX(BALANCE)
FROM ACCOUNTS
WHERE FUND_NUM = "0032"
```

or

```
SELECT FUND_NUM, MAX(BALANCE)
FROM ACCOUNTS
GROUP BY FUND_NUM
HAVING FUND_NUM = "0032"
```

BUT, there is only one way to obtain aggregates for more than one group separately: GROUP BY/HAVING!

```
SELECT FUND_NUM, MAX(BALANCE)
FROM ACCOUNTS
GROUP BY FUND_NUM
HAVING FUND_NUM = "0032" OR FUND_NUM = "1234"
```

What happens if we try to execute the following?

```
SELECT FUND_NUM, MAX(BALANCE)
FROM ACCOUNTS
WHERE FUND_NUM = "0032" OR FUND_NUM = "1234"
```
Joining and Grouping:  Find the total quantity of each color of part on order
(in other words, how many RED parts are on order, how many FUSCIA parts are on order, & so on.)

```
SELECT P.COLOR, SUM(SP.QTY)
FROM PARTS P, SHIPMENTS SP
WHERE P.P# = SP.P#
GROUP BY P.COLOR
```

Note: WHERE clause same as a join!!
GROUP BY operates on result of the join!

How many students are registered for every course that has the word “Database” in its course description

```
SELECT C.Course#, COUNT(R.Course#)
FROM Courses C, Regis R
WHERE C.Course# = R.Course#
    AND C.Section = R.Section
    AND C.Description LIKE "*Database*"
GROUP BY C.Course#
```