61A Lecture 34

Wednesday, April 22
Announcements
Announcements

- Project 4 due Thursday 4/23 @ 11:59pm
Announcements

• Project 4 due Thursday 4/23 @ 11:59pm
  • Early point #2: All questions (including Extra Credit) by Wednesday 4/22 @ 11:59pm
Announcements

• Project 4 due Thursday 4/23 @ 11:59pm
  ▪ Early point #2: All questions (including Extra Credit) by Wednesday 4/22 @ 11:59pm
• Recursive Art Contest Entries due Monday 4/27 @ 11:59pm
Announcements

- Project 4 due Thursday 4/23 @ 11:59pm
  - Early point #2: All questions (including Extra Credit) by Wednesday 4/22 @ 11:59pm
- Recursive Art Contest Entries due Monday 4/27 @ 11:59pm
  - Email your code & a screenshot of your art to cs61a-tae@imail.eecs.berkeley.edu (Albert)
Announcements

• Project 4 due Thursday 4/23 @ 11:59pm
  ▪ Early point #2: All questions (including Extra Credit) by Wednesday 4/22 @ 11:59pm
• Recursive Art Contest Entries due Monday 4/27 @ 11:59pm
  ▪ Email your code & a screenshot of your art to cs61a-tae@imail.eecs.berkeley.edu (Albert)
• Homework 9 merged with Homework 10; both are due Wednesday 4/29 @ 11:59pm
Announcements

• Project 4 due Thursday 4/23 @ 11:59pm
  ▪ Early point #2: All questions (including Extra Credit) by Wednesday 4/22 @ 11:59pm
• Recursive Art Contest Entries due Monday 4/27 @ 11:59pm
  ▪ Email your code & a screenshot of your art to cs61a-tae@mail.eecs.berkeley.edu (Albert)
• Homework 9 merged with Homework 10; both are due Wednesday 4/29 @ 11:59pm
• Quiz 4 (SQL) released on Tuesday 4/28 is due Thursday 4/30 @ 11:59pm
Aggregation
Aggregate Functions
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time.
 Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

```
select [columns] from [table] where [expression] order by [expression];
```

```markdown
[expression] as [name], [expression] as [name], ...
```
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

```
select [expression] as [name], [expression] as [name], ...
```

An aggregate function in the [columns] clause computes a value from a group of rows
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

\[
\text{[expression]} \text{ as [name]}, \text{[expression]} \text{ as [name]}, \ldots
\]

select [columns] from [table] where [expression] order by [expression];

An aggregate function in the [columns] clause computes a value from a group of rows

create table animals as
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

```
[expression] as [name], [expression] as [name], ...
```

```sql
select [columns] from [table] where [expression] order by [expression];
```

An aggregate function in the [columns] clause computes a value from a group of rows

```sql
create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
```
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

```
expression as name, expression as name, ...
```

```sql
select [columns] from [table] where [expression] order by [expression];
```

An aggregate function in the [columns] clause computes a value from a group of rows

```sql
create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
select "cat" , 4 , 10 union
```
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

\[
[\text{expression]} \text{ as [name]}, [\text{expression}] \text{ as [name]}, ...;
\]

\[
\text{select [columns] from [table] where [expression] order by [expression]};
\]

An aggregate function in the [columns] clause computes a value from a group of rows

```
create table animals as
  select "dog" as kind, 4 as legs, 20 as weight union
select "cat", 4, 10 union
select "ferret", 4, 10 union
```
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

```
[expression] as [name], [expression] as [name], ...;
```

```
select [columns] from [table] where [expression] order by [expression];
```

An aggregate function in the [columns] clause computes a value from a group of rows

```sql
create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
select "cat" , 4 , 10 union
select "ferret" , 4 , 10 union
select "parrot" , 2 , 6 union
```
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

```
[expression] as [name], [expression] as [name], ...
```

```
select [columns] from [table] where [expression] order by [expression];
```

An aggregate function in the [columns] clause computes a value from a group of rows

```
create table animals as
    select "dog" as kind, 4 as legs, 20 as weight union
    select "cat" , 4 , 10  union
    select "ferret" , 4 , 10  union
    select "parrot" , 2 , 6  union
    select "penguin" , 2 , 10 union
```
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time.

\[
\text{select [expression] as [name], [expression] as [name], ...}
\]

\[
\text{select [columns] from [table] where [expression] order by [expression];}
\]

An aggregate function in the [columns] clause computes a value from a group of rows.

```sql
create table animals as
    select "dog" as kind, 4 as legs, 20 as weight union
    select "cat" , 4 , 10     union
    select "ferret" , 4 , 10     union
    select "parrot" , 2 , 6     union
    select "penguin" , 2 , 10     union
    select "t-rex" , 2 , 12000;
```
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

```
[expression] as [name], [expression] as [name], ...
```

```
select [columns] from [table] where [expression] order by [expression];
```

An aggregate function in the [columns] clause computes a value from a group of rows

```
create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
select "cat"  , 4  , 10  union
select "ferret", 4  , 10  union
select "parrot", 2  , 6   union
select "penguin", 2  , 10  union
select "t-rex"  , 2  , 12000;
```

```
<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
```
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

```
select [columns] from [table] where [expression] order by [expression];
```

An aggregate function in the [columns] clause computes a value from a group of rows

```
create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
select "cat" , 4 , 10 union
select "ferret" , 4 , 10 union
select "parrot" , 2 , 6 union
select "penguin" , 2 , 10 union
select "t-rex" , 2 , 12000;

select max(legs) from animals;
```
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

\[
\text{[expression] as [name], [expression] as [name], ...}
\]

\[
\text{select [columns] from [table] where [expression] order by [expression];}
\]

An aggregate function in the [columns] clause computes a value from a group of rows

```sql
create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
select "cat" , 4 , 10 union
select "ferret" , 4 , 10 union
select "parrot" , 2 , 6 union
select "penguin" , 2 , 10 union
select "t-rex" , 2 , 12000;
```

```sql
select max(legs) from animals;
```

<table>
<thead>
<tr>
<th>animals:</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Aggregate Functions

So far, all SQL expressions have referred to the values in a single row at a time

\[
\text{select [expression] as [name], [expression] as [name], ...} \\
\text{select [columns] from [table] where [expression] order by [expression];}
\]

An aggregate function in the [columns] clause computes a value from a group of rows

create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
select "cat" , 4 , 10 union
select "ferret" , 4 , 10 union
select "parrot" , 2 , 6 union
select "penguin" , 2 , 10 union
select "t-rex" , 2 , 12000;

select max(legs) from animals;

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>max(legs)</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

animals:

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Mixing Aggregate Functions and Single Values

```sql
create table animals as
    select "dog" as kind, 4 as legs, 20 as weight union
    select "cat" , 4 , 10  union
    select "ferret" , 4 , 10  union
    select "parrot" , 2 , 6  union
    select "penguin" , 2 , 10  union
    select "t-rex" , 2 , 12000;
```

**animals:**

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Mixing Aggregate Functions and Single Values

An aggregate function also selects a row in the table, which may be meaningful

```sql
create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
select "cat", 4, 10 union
select "ferret", 4, 10 union
select "parrot", 2, 6 union
select "penguin", 2, 10 union
select "t-rex", 2, 12000;
```

animals:

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Mixing Aggregate Functions and Single Values

An aggregate function also selects a row in the table, which may be meaningful

```sql
SELECT MAX(weight), kind FROM animals;
```

create table animals as
```sql
SELECT "dog" as kind, 4 as legs, 20 as weight union
SELECT "cat" , 4 , 10 union
SELECT "ferret" , 4 , 10 union
SELECT "parrot" , 2 , 6 union
SELECT "penguin" , 2 , 10 union
SELECT "t-rex" , 2 , 12000;
```

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Mixing Aggregate Functions and Single Values

An aggregate function also selects a row in the table, which may be meaningful

```sql
select max(weight), kind from animals;
select min(kind), kind from animals;
```

```sql
create table animals as
    select "dog" as kind, 4 as legs, 20 as weight union
    select "cat", 4, 10 union
    select "ferret", 4, 10 union
    select "parrot", 2, 6 union
    select "penguin", 2, 10 union
    select "t-rex", 2, 12000;
```

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Mixing Aggregate Functions and Single Values

An aggregate function also selects a row in the table, which may be meaningful

```
select max(weight), kind from animals;
select max(legs), kind from animals;
select min(kind), kind from animals;
```

```
create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
select "cat" , 4 , 10 union
select "ferret" , 4 , 10 union
select "parrot" , 2 , 6 union
select "penguin" , 2 , 10 union
select "t-rex" , 2 , 12000;
```

```
animals:

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
```
Mixing Aggregate Functions and Single Values

An aggregate function also selects a row in the table, which may be meaningful:

- `select max(weight), kind from animals;`  
- `select min(kind), kind from animals;`  
- `select max(legs), kind from animals;`  
- `select avg(weight), kind from animals;`

```sql
create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
select "cat" , 4 , 10 union
select "ferret" , 4 , 10 union
select "parrot" , 2 , 6 union
select "penguin" , 2 , 10 union
select "t-rex" , 2 , 12000;
```

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Mixing Aggregate Functions and Single Values

An aggregate function also selects a row in the table, which may be meaningful

```
select max(weight), kind from animals;
select min(kind), kind from animals;
select max(legs), kind from animals;
select avg(weight), kind from animals;
```

(Demo)

```sql
create table animals as
select "dog" as kind, 4 as legs, 20 as weight union
select "cat", 4, 10 union
select "ferret", 4, 10 union
select "parrot", 2, 6 union
select "penguin", 2, 10 union
select "t-rex", 2, 12000;
```

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Groups
Grouping Rows
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group.
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

```
expressions as [name], expressions as [name], ...

select [columns] from [table] group by [expression] having [expression];
```
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

```
select [columns] from [table] group by [expression] having [expression];
```

The number of groups is the number of unique values of an expression
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

```
select [columns] from [table] group by [expression] having [expression];
```

The number of groups is the number of unique values of an expression

**animals:**

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

```
[expression] as [name], [expression] as [name], ...
```

```
select [columns] from [table] group by [expression] having [expression];
```

The number of groups is the number of unique values of an expression

```
select legs, max(weight) from animals group by legs;
```

### animals:

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

\[
\text{select [columns] from [table] group by [expression] having [expression];}
\]

The number of groups is the number of unique values of an expression

\[
\text{select legs, max(weight) from animals group by legs;}
\]

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

```
[expression] as [name], [expression] as [name], ...
```

```
select [columns] from [table] group by [expression] having [expression];
```

The number of groups is the number of unique values of an expression

```
select legs, max(weight) from animals group by legs;
```

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

```
[expression] as [name], [expression] as [name], ...
```

```
select [columns] from [table] group by [expression] having [expression];
```

The number of groups is the number of unique values of an expression

```
select legs, max(weight) from animals group by legs;
```

**animals:**

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>

legs=4
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

```sql
select [columns] from [table] group by [expression] having [expression];
```

The number of groups is the number of unique values of an expression

```sql
select legs, max(weight) from animals group by legs;
```

animals:

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>

legs=4

legs=2
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

```
[expression] as [name], [expression] as [name], ...
```

```
select [columns] from [table] group by [expression] having [expression];
```

The number of groups is the number of unique values of an expression

```
select legs, max(weight) from animals group by legs;
```

<table>
<thead>
<tr>
<th>legs</th>
<th>max(weight)</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>

animals:

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

```
[expression] as [name], [expression] as [name], ...
```

```
select [columns] from [table] group by [expression] having [expression];
```

The number of groups is the number of unique values of an expression

```
select legs, max(weight) from animals group by legs;
```

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>

(Demo)
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group

\[
\text{[expression] as [name], [expression] as [name], ...}
\]

\[
\text{select [columns] from [table] group by [expression] having [expression];}
\]

animals:

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group.

```
select [columns] from [table] group by [expression] having [expression];
```

A `having` clause filters the set of groups that are aggregated.
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group

```
[expression] as [name], [expression] as [name], ...
```

```
select [columns] from [table] group by [expression] having [expression];
```

A `having` clause filters the set of groups that are aggregated

```
select weight/legs, count(*) from animals group by weight/legs having count(*)>1;
```

**animals:**

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group

```sql
select [columns] from [table] group by [expression] having [expression];
```

A `having` clause filters the set of groups that are aggregated

```sql
select weight/legs, count(*) from animals group by weight/legs having count(*)>1;
```

<table>
<thead>
<tr>
<th>animals:</th>
<th>weight/legs=5</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>kind</td>
<td>legs</td>
</tr>
<tr>
<td>----------</td>
<td>------</td>
</tr>
<tr>
<td>dog</td>
<td>4</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
</tr>
</tbody>
</table>
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group

```
[expression] as [name], [expression] as [name], ...
```

```
select [columns] from [table] group by [expression] having [expression];
```

A `having` clause filters the set of groups that are aggregated

```
select weight/legs, count(*) from animals group by weight/legs having count(*) > 1;
```

animals:

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>

weight/legs=5
weight/legs=2
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group

```sql
select [columns] from [table] group by [expression] having [expression];
```

A having clause filters the set of groups that are aggregated

```sql
select weight/legs, count(*) from animals group by weight/legs having count(*)>1;
```

**animals:**

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group.

\[
\text{select [columns] from [table] group by [expression] having [expression];}
\]

A `having` clause filters the set of groups that are aggregated.

\[
\text{select weight/legs, count(*) from animals group by weight/legs having count(*)>1;}
\]

<table>
<thead>
<tr>
<th>animals:</th>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>weight/legs=5</td>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>weight/legs=2</td>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>weight/legs=2</td>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>weight/legs=3</td>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group

```
select [columns] from [table] group by [expression] having [expression];
```

A having clause filters the set of groups that are aggregated

```
select weight/legs, count(*) from animals group by weight/legs having count(*)>1;
```

**animals:**

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>

weight/legs=5
weight/legs=2
weight/legs=2
weight/legs=3
weight/legs=5
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group

```sql
SELECT [columns] FROM [table] GROUP BY [expression] HAVING [expression];
```

A **having** clause filters the set of groups that are aggregated

```sql
SELECT weight/legs, count(*) FROM animals GROUP BY weight/legs HAVING count(*)>1;
```

**animals:**

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group

```
select [columns] from [table] group by [expression] having [expression];
```

A `having` clause filters the set of groups that are aggregated

```
select weight/legs, count(*) from animals group by weight/legs having count(*)>1;
```

<table>
<thead>
<tr>
<th>weight/legs</th>
<th>count(*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

Animals:

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group

```
select [columns] from [table] group by [expression] having [expression];
```

A having clause filters the set of groups that are aggregated

```
select weight/legs, count(*) from animals group by weight/legs having count(*)>1;
```

animals:

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>weight/legs</th>
<th>count(*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>weight/legs=5</td>
<td>2</td>
</tr>
<tr>
<td>weight/legs=2</td>
<td>2</td>
</tr>
<tr>
<td>weight/legs=2</td>
<td>2</td>
</tr>
<tr>
<td>weight/legs=3</td>
<td></td>
</tr>
<tr>
<td>weight/legs=5</td>
<td></td>
</tr>
<tr>
<td>weight/legs=6000</td>
<td></td>
</tr>
</tbody>
</table>
Selecting Groups

Rows in a table can be grouped, and aggregation is performed on each group:

```
[expression] as [name], [expression] as [name], ...
```

```
select [columns] from [table] group by [expression] having [expression];
```

A `having` clause filters the set of groups that are aggregated:

```
select weight/legs, count(*) from animals group by weight/legs having count(*) > 1;
```

<table>
<thead>
<tr>
<th>weight/legs</th>
<th>count(*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>2</td>
</tr>
</tbody>
</table>

```
animals:
```

<table>
<thead>
<tr>
<th>kind</th>
<th>legs</th>
<th>weight</th>
</tr>
</thead>
<tbody>
<tr>
<td>dog</td>
<td>4</td>
<td>20</td>
</tr>
<tr>
<td>cat</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>ferret</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td>parrot</td>
<td>2</td>
<td>6</td>
</tr>
<tr>
<td>penguin</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>t-rex</td>
<td>2</td>
<td>12000</td>
</tr>
</tbody>
</table>
Select Grammar
SQL Select Statements
SQL Select Statements
SQL Select Statements

WITH
  RECURSIVE
    common-table-expression

SELECT
  DISTINCT
    result-column
    ALL

FROM
  table-or-subquery

WHERE
  expr

GROUP BY
  expr

HAVING
  expr

ORDER BY
  ordering-term

LIMIT
  expr

OFFSET
  expr

VALUES
  ( expr , )

compound-operator

http://www.sqlite.org/lang_select.html
SQL Select Statements

SELECT
  DISTINCT
  ALL
FROM
  table-or-subquery
  join-clause
WHERE
  expr
GROUP BY
  expr
HAVING
  expr
ORDER BY
  ordering-term
LIMIT
  expr
OFFSET
  expr
VALUES
  expr
compound-operator

http://www.sqlite.org/lang_select.html
SQL Select Statements

http://www.sqlite.org/lang_select.html
SQL Select Statements

http://www.sqlite.org/lang_select.html
SQL Select Statements

http://www.sqlite.org/lang_select.html
SQL Select Statements

http://www.sqlite.org/lang_select.html
SQL Select Statements

http://www.sqlite.org/lang_select.html
SQL Select Statements

WITH
  | RECURSIVE
  | common-table-expression

SELECT
  | DISTINCT
  | ALL

FROM
  | table-or-subquery
  | join-clause

WHERE
  | expr

GROUP
  | BY
  | expr

HAVING
  | expr

VALUES
  | { expr }

ORDER
  | BY
  | ordering-term

LIMIT
  | expr

OFFSET
  | expr

http://www.sqlite.org/lang_select.html
SQL Select Statements

http://www.sqlite.org/lang_select.html