Announcements
Joining Tables
Reminder: John the Patriotic Dog Breeder

create table parents as
select "abraham" as parent, "barack" as child union
select "abraham" , "clinton" union
select "delano" , "herbert" union
select "fillmore" , "abraham" union
select "fillmore" , "delano" union
select "fillmore" , "grover" union
select "eisenhower" , "fillmore";

Parents:

<table>
<thead>
<tr>
<th>Parent</th>
<th>Child</th>
</tr>
</thead>
<tbody>
<tr>
<td>abraham</td>
<td>barack</td>
</tr>
<tr>
<td>abraham</td>
<td>clinton</td>
</tr>
<tr>
<td>delano</td>
<td>herbert</td>
</tr>
<tr>
<td>fillmore</td>
<td>abraham</td>
</tr>
<tr>
<td>fillmore</td>
<td>delano</td>
</tr>
<tr>
<td>fillmore</td>
<td>grover</td>
</tr>
<tr>
<td>eisenhower</td>
<td>fillmore</td>
</tr>
</tbody>
</table>
Joining Two Tables

Two tables A & B are joined by a comma to yield all combos of a row from A & a row from B

```sql
create table dogs as
    select "abraham" as name, "long" as fur union
    select "barack"  , "short"    union
    select "clinton" , "long"     union
    select "delano"  , "long"     union
    select "eisenhower", "short"  union
    select "fillmore", "curly"    union
    select "grover", "short"      union
    select "herbert", "curly";

create table parents as
    select "abraham" as parent, "barack" as child union
    select "abraham"  , "clinton"    union
    ...;
```

Select the parents of curly-furred dogs

```sql
select parent from parents, dogs
    where child = name and fur = "curly";
```
Aliases and Dot Expressions
Joining a Table with Itself

Two tables may share a column name; dot expressions and aliases disambiguate column values

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

[table] is a comma-separated list of table names with optional aliases

Select all pairs of siblings

```sql
select a.child as first, b.child as second
from parents as a, parents as b
where a.parent = b.parent and a.child < b.child;
```

<table>
<thead>
<tr>
<th>First</th>
<th>Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>barack</td>
<td>clinton</td>
</tr>
<tr>
<td>abraham</td>
<td>delano</td>
</tr>
<tr>
<td>abraham</td>
<td>grover</td>
</tr>
<tr>
<td>delano</td>
<td>grover</td>
</tr>
</tbody>
</table>
Example: Grandparents

Which select statement evaluates to all grandparent, grandchild pairs?

1. `select a.grandparent, b.child from parents as a, parents as b
   where b.parent = a.child;`

2. `select a.parent, b.child from parents as a, parents as b
   where a.parent = b.child;`

3. `select a.parent, b.child from parents as a, parents as b
   where b.parent = a.child;`

4. `select a.grandparent, b.child from parents as a, parents as b
   where a.parent = b.child;`

5. None of the above
Joining Multiple Tables

Multiple tables can be joined to yield all combinations of rows from each

```
create table grandparents as
    select a.parent as grandog, b.child as granpup
    from parents as a, parents as b
    where b.parent = a.child;
```

Select all grandparents with the same fur as their grandchildren

```
select grandog from grandparents, dogs as c, dogs as d
    where grandog = c.name and
        granpup = d.name and
        c.fur = d.fur;
```

Which tables need to be joined together?
Example: Dog Triples
Fall 2014 Quiz Question (Slightly Modified)

Write a SQL query that selects all possible combinations of three different dogs with the same fur and lists each triple in *inverse* alphabetical order.

```sql
create table dogs as
    select "abraham" as name, "long" as fur union
    select "barack", "short" union
    ...
;
create table parents as
    select "abraham" as parent, "barack" as child union
    select "abraham", "clinton" union
    ...
;
```

Expected output:

```
delano|clinton|abraham
grover|eisenhower|barack
```

(Demo)
Numerical Expressions
Numerical Expressions

Expressions can contain function calls and arithmetic operators

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

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

Combine values: +, -, *, /, %, and, or

Transform values: abs, round, not, -

Compare values: <, <=, >, >=, <>, !=, =

(Demo)
String Expressions
String Expressions

String values can be combined to form longer strings

```
sqlite> select "hello," || " world";
hello, world
```

Basic string manipulation is built into SQL, but differs from Python

```
sqlite> create table phrase as select "hello, world" as s;
sqlite> select substr(s, 4, 2) || substr(s, instr(s, " ")+1, 1) from phrase;
low
```

Strings can be used to represent structured values, but doing so is rarely a good idea

```
sqlite> create table lists as select "one" as car, "two,three,four" as cdr;
sqlite> select substr(cdr, 1, instr(cdr, ",")-1) as cadr from lists;
two
```

(Demo)
Database Management Systems
Database Management System Architecture

Architecture of a Database System by Hellerstein, Stonebreaker, and Hamilton
Query Planning

The manner in which tables are filtered, sorted, and joined affects execution time.

Select the parents of curly-furred dogs:

```sql
select parent from parents, dogs
where child = name and fur = "curly";
```

Join all rows of parents to all rows of dogs, filter by \texttt{child = name} and \texttt{fur = "curly"}.

Join only rows of parents and dogs where \texttt{child = name}, filter by \texttt{fur = "curly"}.

Filter dogs by \texttt{fur = "curly"}, join result with all rows of parents, filter by \texttt{child = name}.

Filter dogs by \texttt{fur = "curly"}, join only rows of result and parents where \texttt{child = name}.