Joining Tables

**Create Table: dogs**

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

**Create Table: parents**

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

**Select Parents of Curly-Furred Dogs**

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

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.

**Create Table: dogs**

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

**Create Table: parents**

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

**Select the Parents of Curly-Furred Dogs**

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

Joining a Table with Itself

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

**Select Parents of Curly-Furred Dogs**

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

Aliases and Dot Expressions

**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;
```

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 a.child = b.parent;`
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.

```sql
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.

Which tables need to be joined together?

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

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

Expected output:

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

Numerical Expressions

Expressions can contain function calls and arithmetic operators.

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

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

Transform values: abs, round, not, -

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

String Expressions

String values can be combined to form longer strings.

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

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

```sql
sqlite> select create table phrase as select "hello, world" as s;
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.

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

(Demo)