Reminder: Use Discord for voice chat with the course staff. Write to @discuss in the #discuss-queue channel on Discord at any time, and a member of the course staff will join your group’s voice channel.

Pick someone in your group to join Discord. It’s fine if multiple people join, but one is enough.

Now switch to Pensieve:

- **Everyone**: Go to discuss.pensieve.co and log in with your @berkeley.edu email, then enter your group number. (Your group number is the number of your Discord channel.)

Once you’re on Pensieve, you don’t need to return to this page; Pensieve has all the same content (but more features). If for some reason Pensieve doesn’t work, return to this page and continue with the discussion.

Post in the #help channel on Discord if you have trouble.

**Pro tip**: Any of you can type a question into your group’s Discord channel’s text chat with the @discuss tag, and a member of the course staff will respond.

**Getting Started**

If you have only 1 or 2 people in your group, you can join the other group in the room with you.

Everybody say your name, and then share your favorite restaurant, cafe, or boba shop near campus. (Yes, Kingpin Donuts counts as a restaurant.)

**Select Statements**

A SELECT statement describes an output table based on input rows. To write one: 1. Describe the input rows using FROM and WHERE clauses. 2. Format and order the output rows and columns using SELECT and ORDER BY clauses.

```
SELECT (Step 2) FROM (Step 1) WHERE (Step 1) ORDER BY (Step 2);
```

Step 1 may involve joining tables (using commas) to form input rows that consist of two or more rows from existing tables.

The WHERE and ORDER BY clauses are optional.

**Pizza Time**

The pizzas table contains the names, opening, and closing hours of great pizza places in Berkeley. The meals table contains typical meal times (for college students). A pizza place is open for a meal if the meal time is at or within the open and close times.
CREATE TABLE pizzas AS
    SELECT "Artichoke" AS name, 12 AS open, 15 AS close UNION
    SELECT "La Val's" , 11 , 22 UNION
    SELECT "Sliver" , 11 , 20 UNION
    SELECT "Cheeseboard" , 16 , 23 UNION
    SELECT "Emilia's" , 13 , 18;

CREATE TABLE meals AS
    SELECT "breakfast" AS meal, 11 AS time UNION
    SELECT "lunch" , 13 UNION
    SELECT "dinner" , 19 UNION
    SELECT "snack" , 22;

Q1: Open Early
You'd like to have pizza before 13 o'clock (1pm). Create a opening table with the names of all pizza places that open before 13 o'clock, listed in reverse alphabetical order.

opening table:

<table>
<thead>
<tr>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sliver</td>
</tr>
<tr>
<td>La Val’s</td>
</tr>
<tr>
<td>Artichoke</td>
</tr>
</tbody>
</table>

-- Pizza places that open before 1pm in alphabetical order
CREATE TABLE opening AS
    SELECT name FROM pizzas WHERE open < 13 ORDER BY name DESC;

To order by name in reverse alphabetical order, write ORDER BY name DESC.

Q2: Study Session
You're planning to study at a pizza place from the moment it opens until 14 o'clock (2pm). Create a table study with two columns, the name of each pizza place and the duration of the study session you would have if you studied there (the difference between when it opens and 14 o'clock). For pizza places that are not open before 2pm, the duration should be zero. Order the rows by decreasing duration.

Hint: Use an expression of the form MAX(_, 0) to make sure a result is not below 0.

study table:

<table>
<thead>
<tr>
<th>name</th>
<th>duration</th>
</tr>
</thead>
<tbody>
<tr>
<td>La Val's</td>
<td>3</td>
</tr>
<tr>
<td>Sliver</td>
<td>3</td>
</tr>
<tr>
<td>Artichoke</td>
<td>2</td>
</tr>
</tbody>
</table>
$$\begin{array}{ll}
\text{name} & \text{duration} \\
\text{Emilia's} & 1 \\
\text{Cheeseboard} & 0 \\
\end{array}$$

-- Pizza places and the duration of a study break that ends at 14 o'clock
CREATE TABLE study AS
  SELECT name, MAX(14 - open, 0) AS duration FROM pizzas ORDER BY duration DESC;

To order by decreasing duration, first name the column with SELECT ..., ... AS duration ..., then ORDER BY duration DESC.

Q3: Late Night Snack
What’s still open for a late night snack? Create a late table with one column named status that has a sentence describing the closing time of each pizza place that closes at or after snack time. Important: Don’t use any numbers in your SQL query! Instead, use a join to compare each restaurant’s closing time to the time of a snack. The rows may appear in any order.
late table:

$$\begin{array}{l}
\text{status} \\
\text{Cheeseboard closes at 23} \\
\text{La Val’s closes at 22} \\
\end{array}$$

-- Pizza places that are open for late-night-snack time and when they close
CREATE TABLE late AS
  SELECT name || " closes at " || close AS status FROM pizzas, meals WHERE meal="snack"
  AND time<=clos;

To compare a pizza place’s close time to the time of a snack: - join the pizzas and meals tables using FROM pizzas, meals - use only rows where the meal is a "snack" - compare the time of the snack to the close of the pizza place.

Use name || " closes at " || close to create the sentences in the resulting table. The || operator concatenates values into strings.

Q4: Double Pizza
If two meals are more than 6 hours apart, then there’s nothing wrong with going to the same pizza place for both, right? Create a double table with three columns. The first column is the earlier meal, the second column is the later meal, and the name column is the name of a pizza place. Only include rows that describe two meals that are more than 6 hours apart and a pizza place that is open for both of the meals. The rows may appear in any order.
double table:
<table>
<thead>
<tr>
<th>first</th>
<th>second</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>breakfast</td>
<td>dinner</td>
<td>La Val’s</td>
</tr>
<tr>
<td>breakfast</td>
<td>dinner</td>
<td>Sliver</td>
</tr>
<tr>
<td>breakfast</td>
<td>snack</td>
<td>La Val’s</td>
</tr>
<tr>
<td>lunch</td>
<td>snack</td>
<td>La Val’s</td>
</tr>
</tbody>
</table>

```sql
-- Two meals at the same place
CREATE TABLE double AS
SELECT a.meal AS first, b.meal AS second, name
FROM meals AS a, meals AS b, pizzas
WHERE open <= a.time AND a.time <= close AND
     open <= b.time AND b.time <= close AND
     b.time > a.time + 6;
```

Use `FROM meals AS a, meals AS b, pizzas` so that each row has info about two meals and a pizza place. Then you can write a `WHERE` clause that compares both `a.time` and `b.time` to `open` and `close` and each other to ensure all the relevant conditions are met.

**Document the Occasion**

Please all fill out the attendance form (one submission per person per week).

**Important:** Please help put the furniture in the room back where you found it before you leave. Thanks!

If you finish early, maybe go get pizza together…

*Note: This worksheet is a problem bank—most TAs will not cover all the problems in discussion section.*