Import and Browse

Review data

bp_stages is a chart based on a graphic

hrs_clin is clinical data
patient id (anonymized)
some interesting things to note here.
female is a boolean
age is a number
depress_dx is a 0/1, whether someone has been diagnosed
bp_dx is a yes/no, whether someone has been diagnosed
bp_rx is yes/no, whether someone has been prescribed bp medication
systolic, diastolic readings
and an id linking back to the bp_stages (more on that later)

hrs_genetic
patient id
polygen_map (genetic markers for mean arterial pressure)
polyten_dep (genetic markers for depression)

Data Cleanup

Modify the data types

discuss integer, numeric boolean, text

Querying

All:

SELECT * FROM hrs_clin

One column

SELECT bp_dia FROM hrs_clin

More than one

SELECT id, bp_dia, bp_sys FROM hrs_clin
DISTINCT

SELECT bp_stage_id FROM hrs_clin
SELECT DISTINCT bp_stage_id FROM hrs_clin

ORDER BY

SELECT DISTINCT bp_stage_id FROM hrs_clin
ORDER BY bp_stage_id

NOTE: DESC for descending order, ASC for ascending (default if you don’t specify it)

Exercise:

How many distinct ages are in the data set?
How many distinct values for female?

Example: what is our age range?

SELECT DISTINCT age FROM hrs_clin_sql
ORDER BY age DESC

This is a problem because of data issues!
SELECT MAX(age), MIN(age) FROM hrs_clin_sql
ORDER BY age DESC

we will get to this after filtering...
https://github.com/sqlitebrowser/sqlitebrowser/issues/195

Filtering

SELECT * FROM hrs_clin
WHERE female = 'yes'

Exercise: Select all rows where bp is at stage 2

inequalities

SELECT * FROM hrs_clin WHERE bp_sys > 120

More than one condition in a filter

SELECT * FROM hrs_clin WHERE bp_sys > 120 AND bp_dia > 80
you can use an OR clause as well

SELECT * FROM hrs_clin WHERE bp_sys > 120 OR bp_dia > 80

Exercise:

Find all rows with low blood pressure

SELECT * FROM hrs_clin WHERE bp_sys <100 AND bp_dia < 60

Combining boolean clauses

Mixing AND and OR

SELECT * FROM hrs_clin
WHERE
female = 'no' OR bp_stage_id = 4
AND bp_sys > 200

Now add parentheses around

SELECT * FROM hrs_clin
WHERE
(female = 'no' OR bp_stage_id = 4)
AND bp_sys > 200

Note that the parentheses *can* matter

Try counting the results, look at them. what’s going on? The first one is applying the OR clause to the entire remainder of the query. With parentheses, it is only applied to a joint condition

Solution, and advice - use parentheses. Sometimes they won’t matter, sometimes they will (remember PEMDAS). It makes it readable, and you won’t have to work through your own tricky logic. Remember, data can change, so no difference with one data set can become a difference one more data arrives!

Yes, also, you may note that there is a redundant condition in the query above!

Exercise:

Find everyone with really low diastolic and systolic, or really high diastolic and systolic
SELECT * FROM hrs_clin
WHERE
(bp_dia < 50 AND bp_sys < 100)
OR
(bp_dia > 110 AND bp_sys > 180)

Add'l: do the parenthese make any difference? in the results? to you?

What if I wrote this?

SELECT * FROM hrs_clin
WHERE
bp_dia < 50
AND bp_sys < 100 OR bp_dia > 110
AND bp_sys > 180

It’s about readability, not just what the computer does.

IN

SELECT * FROM hrs_clin
WHERE bp_stage_id
IN (0, 2, 5)

AGGREGATIONS

SELECT female, COUNT(id)
FROM hrs_clin
GROUP BY female

SELECT female, AVG(age)
FROM hrs_clin
GROUP BY female

Can you get the AVG systolic by female?

SELECT female, AVG(bp_sys)
FROM hrs_clin
GROUP BY female

(uncategorized is low! huh. I don’t really know what to make of that)

Exercise: Is it true for diastolic?
You can get more than one aggregation at a time

SELECT age, COUNT(id), AVG(bp_dia)
FROM hrs_clin
GROUP BY age

HAVING

SELECT age, COUNT(id), AVG(bp_dia)
FROM hrs_clin
GROUP BY age
HAVING count(id) > 50

You can group by more than one thing too

SELECT female, age, count(id)
FROM hrs_clin
GROUP BY female, age

ORDERING

SELECT female, age, count(id)
FROM hrs_clin
GROUP BY female, age
ORDER BY count(id) DESC

Exercise

Count the number of people who have or have not been diagnosed with bp

SELECT bp_rx, count(bp_rx)
FROM hrs_clin
GROUP BY bp_rx

Count the number of people who have been diagnosed with bp by female(y/n)

SELECT female, bp Rx, count(bp_rx)
FROM hrs_clin
GROUP BY female, bp_rx
Feeling irritated with the null values?

There are things you can do.

```sql
SELECT female, bp_rx, count(bp_rx)
FROM hrs_clin
WHERE female != ''
AND bp_rx != ''
GROUP BY female, bp_rx

But the blank should probably be a NULL

Data Cleaning

UPDATE hrs_clin SET bp_rx = NULL where bp_rx = ''

JOINS

To join on stages through a common element

```sql
SELECT * FROM hrs_clin,
bp_stages
ON hrs_clin.bp_stage_id = bp_stages.id
```

You can now select and filter (and group) as if it were one table

```sql
SELECT age, bp_sys, bp_dia FROM hrs_clin,
bp_stages
ON hrs_clin.bp_stage_id = bp_stages.id
where high_blood_pressure = 'yes'
```

AGGREGATIONS

```sql
SELECT high_blood_pressure, count(hrs_clin.id) FROM hrs_clin,
bp_stages
ON hrs_clin.bp_stage_id = bp_stages.id
GROUP BY high_blood_pressure
```

(note what happens if you don’t use the alias)

**Exercise**: get the average systolic and diastolic for people who have high blood pressure.
**Exercise:** How many people who have high blood pressure have been diagnosed?

```sql
SELECT high_blood_pressure, bp_dx, count(hrs_clin.id) FROM hrs_clin, bp_stages
ON hrs_clin.bp_stage_id = bp_stages.id
WHERE bp_dx != ''
GROUP BY high_blood_pressure, bp_dx
```

Check with smaller independent queries like this:

```sql
select count(1) FROM hrs_clin where bp_dx = 'yes' and bp_stage_id > 2;
```

**Joining multiple tables**

```sql
SELECT * FROM hrs_clin
JOIN bp_stages
ON hrs_clin.bp_stage_id = bp_stages.id
JOIN hrs_genetic
ON hrs_clin.id = hrs_genetic.id
```