Name: _____

CS109 SQL - Test 2

Short Answer (5pts each)

For the following questions, provide a short written answer of around 3-5 sentences. A simple 'yes' or 'no' will not suffice.

1. In what situation might COUNT produce a different number of rows than the actual count of rows in the resultset? How would you format your function call to return this alternate number?

   When counting rows with Nulls in them.
   Examples: COUNT(address_id), COUNT(bio)

2. What is the name for the concept of forcefully converting a value from one data type to another? When might this be useful?

   Casting. This might be useful when working with Dates/Times or when stripping off decimal points such as Float -> Int.

3. In your own words, describe the difference between aggregate functions, row-level functions and calculated fields.

   Aggregates count the entire column, return one row (the calculation result)
   Row-level function modify the row data in-place
   Calculated fields are arithmetic based: + - / % *

4. Provide an example of the result of a query without grouping that includes a COUNT aggregate function with a non-aggregate field. Why is this result invalid?

   River Tam has 2 addresses
   The database fills in the non-aggregate field 'first' with a dummy value that is not aggregated or a group label. This value as a result of being non-grouped and a non-aggregate applies to only a single row...but grouped columns and aggregates apply to multiple rows. This causes the value to invalidate the entire row.

5. What is the difference between the WHERE and HAVING clauses? What should HAVING be used to filter?

   WHERE filters the results from the FROM and the JOINS -- Use it to filter non-aggregates
   HAVING filters the results *after* the GROUP BY has grouped the results

   HAVING must b used with the GROUP BY and should only group aggregates

6. Compare the persistence of data between temporary tables and views. Which of the two would be better for a daily report that gathers statistics from a highly used database?

   Temporary Tables live for the duration of the connection; they store their data
   Views live permanently, the don't store any data

   Views are best for a daily report because their definitions are permanent and their data isn't. This means that the daily report can use the view without having to recreate it every day, and the view when run will always use the most recent data in the database because it has to gather that data every time it is run.

Query Interpretation (5pts each)

1. Given the SQL query below regarding airline flight times, explain in plain english what statistics it is calculating. Your explanation must take the called function and grouping into account as to how the data is being categorized.

```
SELECT airline.name, loc1.name AS from, loc2.name AS to,
        AVG(flight.duration) AS length
FROM flight
INNER JOIN airline ON (flight.airline_id = airline.id)
INNER JOIN location AS loc1 ON (flight.from_id = loc1.id)
INNER JOIN location AS loc2 ON (flight.to_id = loc2.id)
GROUP BY airline.name, loc1.name, loc2.name
```

   Display the average flight time for each airline for each departure location for each destination.

   This might be beneficial for an airline comparing their average flight times between locations with competing airlines' average flight times between those same locations.

2. Using the SQL query from the previous question, provide example data that might be returned when it is run. Your answer *must* contain all columns and correctly list data according to the column grouping.

name	from	to	length
Alaska Airlines	Anchorage	Seattle	4
Delta	Anchorage	Seattle	3.5
Delta	Seattle	Anchorage	6
Horizon	Anchorage	Anchorage	0.5
Frontier	Anchorage, AK	Montreal, CA	14

SQL Queries (10pts each)

The following queries reference data that is in the example database provided on a separate handout. Using the database, write down a single query that performs the requested operations. If the question specifies the results to be ordered or specific columns to be selected it is expected that this be part of the query.

1. Create a temporary table containing all faculty in every department. After the query but in the *same* transaction, drop the table that you created.

```
CREATE TEMPORARY TABLE dept_fac AS
SELECT *
FROM person_faculty
INNER JOIN department ON department.id = person_faculty.department_id;

DROP TABLE dept_fac;
```

2. Create two views: one that displays all students registered for a class and a second that displays all faculty registered for a class. The names for these views must conform to best practices.

```
CREATE VIEW students_v AS
SELECT *
FROM person
INNER JOIN class_registration reg ON reg.person_id = person.id
WHERE role_id = 'S';

CREATE VIEW faculty_v AS
SELECT *
FROM person
INNER JOIN class_registration reg ON reg.person_id = person.id
WHERE role_id = 'F'
```

3. Display the number of faculty in each department that are actively teaching a class. You may assume that the temporary table and views from the prior questions still exist.

```
SELECT department.name, COUNT(faculty_v.person_id) AS count
FROM dept_fac
INNER JOIN faculty_v ON faculty_v.person_id = dept_fac.person_id
GROUP BY department.name
```

4. Display the average number of credits currently being taken in each student's resident city. You may assume that the temporary table and views from the prior questions still exist. Note that there are some city names that appear in multiple states; this must be accounted for.

```
SELECT state, city, AVG(credits)
FROM students_v
INNER JOIN address ON address.id = students_v.address_id
INNER JOIN class ON class.id = students_v.class_id
GROUP BY state, city
```

5. Display all course names along with their meeting times. The meeting times must be in a single column according to the following format:
"[Day] at [Start Hour]:[Start Minutes] - [End Hour]:[End Minutes]"
For example: "Tue at 15:30 - 16:45".

```
SELECT class.name, day_id || STRFTIME(' at %H:%M', from_time) || STRFTIME(' - %H:%M',
to_time)
FROM class
INNER JOIN class_schedule ON class_schedule.class_id = class.id
```

6. Given a column 'dob' that contains your date of birth in a table named 'person', display your current age. You may calculate on the year and ignore your birth month & day.
Note that this will not need use of any WHERE clause, joins or grouping.

```
SELECT STRFTIME('%y') - STRFTIME('%y', dob)
FROM person
```

Extra Credit (10pts)

Write a query that displays the name of a single random class that meets 30 minutes into the hour and meets at least twice per week.
Note that a correct query will end up using all statement clauses taught so far: Select, From, Join, Where, Group By, Order By, Having & Top/Limit.

```
SELECT class.name
FROM class
INNER JOIN class_schedule ON class_schedule.class_id = class.id
WHERE STRFTIME('%M', from_time) = 30
GROUP BY class.id
HAVING COUNT(class_schedule.id) >= 2
ORDER BY RANDOM()
LIMIT 1
```