CS109 SQL - Test 1

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. One question will require your answer to be in tabular format. A table has been provided for this purpose.

1. What does "SQL" stand for, and how does it relate to databases?

   Structured Query Language - Language for communicating with databases

2. If duplicate rows are stored in a table, what can you assume about the table's Keys?

   The keys are missing (there are no keys)

3. What are the three different kinds of relationships that connect entities? Describe how these relationships work and what they are used for.

   1:1, 1:M, M:M
   1:1 - Extends information about a table
   1:M - Allows for many entities to relate to a single item
   M:M - Allows for many entities to relate to many entities

4. Give an example of a table name and *at least* 3 attributes that would be used to store information about automobiles and/or their manufacturers.

   (car: *id, make_id, model_id, color1, color2, numWheels)
   (manufacturer: *id, name, founding_date, address_id)

5. Define the term "referential integrity".

   If a value exists in a foreign key field, then it needs to point to a valid primary key

6. Briefly explain the differences between Inner Join and Outer Join. Of the lesser-known join types (Equi, Natural & Cross Join) what is the join type that we have been using most in class?

   Inner join - failed connections removed from the resultset
   Outer Join - Failed connections still appear in resultset. The missing information is padded with NULLs

   Equi-Joins

7. What is the most important concept to keep in mind writing a query that joins a table with itself?

   Table Aliases

8. Given the two SQL queries below, provide example data that might be returned when they are run. Your answer *must* contain all columns, adhere to any ordering requirements, and return a different number of rows as a result of the differing join types.

```
SELECT id, first, last,
    pet.name, pet.age
FROM person
INNER JOIN pet
    ON (pet.owner_id = person.id)
ORDER BY pet.age DESC, pet.name
```

id	first	last	name	age
6	Sally			6
5	Joe			5
4	Rachel			4
3	Bob			3
2	Joe			2
1	Clarence			1

```
SELECT id, first, last,
    pet.name, pet.age
FROM person
LEFT JOIN pet
    ON (pet.owner_id = person.id)
ORDER BY pet.age ASC, pet.name
```

id	first	last	name	age
1	Clarence			1
2	Joe			2
3	Bob			3
4	Rachel			4
5	Joe			5
6	Sally			6
7	Ryan		NULL	NULL

Boolean Expressions (2pts each)

Boolean expressions can be either True or False. Read the following expressions and circle which value what they equate to, keeping in mind their order of operations.

1. **(** **True** False **) =** 'Antelope' < 'Gorilla'

2. **(** True **False** **) =** '1985-01-01' = '1985-05-12**'**

3. **(** **True** False **) =** 45 <= 47 AND 34 = 34

4. **(** True **False** **) =** NOT (True) OR ('UAA' > 'UAF')

5. **(** True **False** **) =** (3 < 5 OR 5 = 5) AND NOT ('SQL' >= 'BOOL')

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 retrieves the requested information. 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. Display only the names of all courses in the class table. Ensure that there are no duplicates in the results.

```
SELECT DISTINCT name
FROM class
```

2. Display all classes that are at least 3.0 credits. The results should be ordered by the number of credits from highest to lowest.

```
SELECT *
FROM class
WHERE credits >= 3.0
ORDER BY credits DESC
```

3. Display only the first and last name of all persons who are also students.

```
SELECT first, last
FROM person
INNER JOIN person_student ON (person_student.person_id = person.id)
```

4. Display only the course name and full day ('Monday', not 'Mon') of all classes in class_schedule.

(Multiple accepted solutions here)

```
SELECT class.name, ct_day.name
FROM class
INNER JOIN class_schedule AS sched ON (sched.class_id = class.id)
INNER JOIN ct_day ON (ct_day.id = sched.day_id)
```

5. Display all classes and the students that are registered for them, *if any*.

(Multiple accepted solutions here)

```
SELECT *
FROM class
LEFT JOIN class_registration AS reg ON (reg.class_id = class.id AND role_id = 'S')
LEFT JOIN person ON (person.id = reg.person_id)
```

Extra Credit (10pts)

Display all students that are registered for a class. If the student is out of state, also display their registered address information. Note: This means that if a student is not registered for a class they should not be displayed, and if the student is not considered out of state then their address should be empty.

```
SELECT *
FROM person
INNER JOIN class_registration AS reg ON (reg.person_id = person.id)
INNER JOIN person_student ON (person_student.person_id = person.id)
LEFT JOIN address ON (address.id = person.address_id AND out_of_state = 1)
WHERE role_id = 'S'
```