CSE 414 Midterm Exam

February 7, 2014

Name ____________________________________

Question 1	/ 15
Question 2	/ 45
Question 3	/ 15
Question 4	/ 25
Total	/ 100

The exam is open textbook but otherwise you may not use any other references. No computers, electronics devices, phones of the smart or not-so-smart variety, telegraphs, telepathy, mirrors, smoke signals, or other contraptions permitted.

The exam lasts 50 min. Please budget your time so you get to all questions.

Please wait to turn the page until everyone has their exam and you are told to begin.

Relax. You are here to learn.

The first several questions deal with a database that stores information about airlines, flights, and pilots. The database consists of the following tables:

Airline(code, name)
Pilot(id, name, flight_hours, airline_code)
Flight(airline_code, number, from_city, to_city, depart_time, arrive_time, day, month, year)
Flown(pilot_id, airline_code, number)

The underlined attributes are keys for each relation. The tables contain the following information:

- *Airline* stores airline names and the two-letter unique code assigned to each airline. Examples: ('UA', 'United Airlines'), ('AA', 'American Airlines'), ('AS', 'Alaska Airlines').

- *Pilot* contains information about individual pilots. id is a unique integer assigned to each pilot, name is a string giving the pilot's name, flight_hours is an integer giving the total number of hours the pilot has flown during his or her career, and airline_code is the two-letter code of the airline that currently employs the pilot. (e.g., 'AA', 'AS', 'UA', etc.). For this exam, we assume that airline_code is not NULL, i.e., every pilot currently works for some airline.

- *Flight* gives information about each unique flight in the database. A flight has an airline_code ('AA', 'UA', etc.), an integer flight number, two three-letter codes giving the origin and destination cities (such as 'SEA' for Seattle, 'LAX' for Los Angeles International, etc.), departure and arrival times as integers giving 24-hour clock time (e.g., 1015 means 10:15 am, 1830 means 18:30 or 6:30 pm), and integers giving the day, month, and year of the flight (e.g., 7, 2, 2014 for Feb. 7, 2014). A single flight like 'AA' 125 may have many occurrences with different dates. The key for this table consists of the airline_code, flight number, and month, day and year attributes.

- *Flown* records which pilots have flown which flights. For example, the entry (1234, 'UA', 120) means that pilot with id 1234 has flown United Airlines flight 120 at least once. An individual pilot may appear many times in this table, once for each different (airline_code, number) pair like 'UA' 120 that the pilot has flown, and a flight may have had many different pilots. Further, a pilot may currently be employed by one airline but have worked for different airlines in the past, so this table may contain entries for that pilot with different airline_codes other than the one the pilot currently works for.

The following attributes are foreign keys in each table:

- Pilot.airline_code, Flight.airline_code, and Flown.airline_code are foreign keys that reference Airline.code
- Flown.pilot_id is a foreign key referencing Pilot.id.

Flown.airline_code and Flown.number together refer to Flight.airline_code and Flight.number, but we will not attempt to enforce foreign key constraints on these attributes.

Answer the questions about this database on the following pages. You may remove this page from the test for reference if that is convenient.

Question 1. (15 points) SQL tables. Write the SQL commands needed to create the Airline and Pilot tables described on the previous page. Be sure to include the correct names and types for all attributes, and any key or foreign key constraints.

Question 2. (45 points) SQL queries and indexes. Write SQL queries to retrieve the requested information from airline database tables described previously. The queries you write must be proper SQL that would be accepted by SQL Server or any other SQL implementation. You should not use incorrect SQL, even if sqlite might produce some sort of answer from the buggy SQL.

(a) (15 points) Give the total number of pilots who flew on at least one flight prior to year 2000.

(b) (15 points) Give the full airline names (not code), flight numbers, and departure times for all flights from 'SEA' (Seattle) to 'HNL' (Honolulu) that depart on or after 10 am (time 1000) on March 18, 2014 (3/18/2014).

(continued next page)

Question 2. (cont.) (c) (15 points) A *senior pilot* is a pilot who has 2000 or more total hours of flight time. List the names of all airlines that currently employ 10 or more senior pilots.

Question 3. (15 points) Suppose we have a large number of queries like the one in part (b) of the previous question ("list all flights from one city to another city on a given date on or after a particular time"). Suggest two indexes that would be most effective in speeding up these queries. Give a brief justification for your answers.

Question 4. (25 points) Relational algebra and query plans. Consider the following schema for the relations R, S, and T to answer part (a) and (b).

R(a, b, c)
S(d, e, f)
T(g, h)

(a) (15 points) Draw a tree giving a relational algebra query plan corresponding to the SQL query below. (Recall that the main relational algebra operators are ⋈, join; σ, select; Π, project; γ, grouping and aggregation; δ, duplicate elimination; and –, difference or subtract.)

```
SELECT R.a, count(*)
FROM R, S, T
WHERE R.a = S.d AND S.f = T.g
GROUP BY R.a
HAVING count(*) > 3;
```

Question 4 (cont.) (b) (10 points) Suppose relations R, S, and T contain the following data:

a	b	c
A	4	6
B	7	3
C	2	5

d	e	f
A	3	1
B	10	8
C	5	14
A	11	7

g	h
1	1
15	2
1	3
1	4
7	5
1	10

What output is produced by the query in part (a) when it is executed with this data? (Query repeated for convenience)

```
SELECT R.a, count(*)
FROM R, S, T
WHERE R.a = S.d AND S.f = T.g
GROUP BY R.a
HAVING count(*) > 3;
```