First Exam CS333

Name____Ellen Stanton___ **Due at 8:30 a.m. on 2/11/2019**

GROUND RULES:

1. Allowable sources: You may use anything in our text or in links I have sent you or in our notes.
   If you have any questions about this exam, please ask me.
   You may not discuss the exam with anyone else.

2. Submission: Please **type** your responses and give me a physical copy or send it to me by email.
   If you send it to me by email I will send you a 'got it' reply ---- and it is **your responsibility** to be sure that I got the reply.

3. Except in truly extraordinary circumstances there **your exam must be submitted by the due date and time specified.**

Honor Code:

In this exam we are imagining a database for Fly Us Airlines.

Fly Us is a small airline with relatively small planes – so that each flight needs a pilot, a co-pilot and two cabin attendants.

Pilots and co-pilots have the same licenses, and Fly Us hires only pilots/co-pilots who already have their licenses for all the types of planes which the airline uses.

Their routes include a handful of cities, and they have multiple flights between some of these cities on a given day.

Grandma Fly does the assignment of personnel to flights, but she tries to have everyone end up in their home base at the end of the day to minimize hotel etc. costs. She also took a database course in college, so she has designed the following tables:.

PILOTS(ID, PilotFamilyName, PilotGivenName, PilotLicense, LicenseExpires, DateLastPhysical)

ATTENDANTS(ID, AttendantFamilyName, AttendantGivenName, DateLastPhysical)

EMPLOYEES(Role, ID, HireDate, DateLastRaise, Salary)

AIRPORTS(AirportCode, CityName, NumberGatesAtAirpot)

CIITIES(CityName, OvernightCostPersonnelForThisCity)

SCHEDULE(FlightNumber, Origin, Destination, ScheduledDepartureTime, ScheduledArrivalTime)

FLIGHTS(FlightNumber, Date, PlaneID, ActualDepartureTime, ActualArrivalTime, Pilot_ID, CoPilot_ID, Attendant1_ID, Attendant2_ID)

ASSIGNMENTS(Date, AssignedRole, ID)

PLANES(PlaneID, Model, LastMajorMaintenanceDate, LastMinorMaintenanceDate)

NOTES:

- Airport code is a 3-letter code assigned by some international organization.(IATA)
  For example, Logan Airport in Boston is BOS.
  A city may have more than one airport – for example , both LaGuardia (LGA) and JFK(JFK) are in NYC.
  So, CityName is not an alternate key for AIRPORTS.

- An airline buys/is assigned a certain number of 'gates' at an airport. That limits how many planes they can have at the terminals at a given time.

- The SCHEDULE table is what Fly Us plans to do each day; the FLIGHTS table describes what actually happened on each day.

- In the FLIGHTS table you may assume that the pilot is the more senior (employed longer) of the pilot/co-pilot and that Attendant 1 is the more senior of Attendants 1 and 2.

- In the EMPLOYEES table the possible values for 'Role' are 'pilot', 'attendant', 'counter', 'baggage', 'office' (where counter is the people who work at the ticket counter and gate; baggage is the baggage handlers and baggage truck drivers, and office is the back office workers.)
  I have not included the tables for baggage handlers, counter people, and office workers, but you may assume that they exist.

  This also means that pilots and co-pilots have the same role ('pilot') in the EMPLOYEES table.

- The ASSIGNMENTS table specifies who works in what assigned roles each day. AssignedRole distinguishes between pilots and co-pilots and between Attendants 1 and 2.
  For example, a person may be a pilot on one flight and a co-pilot on another flight on the same day.

  Both the pilot and co-pilot must be in the PILOTS table. Similarly for attendants.

- We will work with a simplified model, where the each pilot has only one license that covers the various planes that Fly Us uses. (i.e. the planes are similar enough to be included in the same license.)
- The ID columns in the PILOTS and ATTENDANTS tables are not only primary keys for those tables but also reference the ID column in the EMPLOYEES table.

Write code to do the following, and paste your code right below the question:
Each problem is worth 8 points, so there are 112 points on the test.

A. Find a list of all flights which go (daily) from Boston to NYC

```
SELECT *
  FROM SCHEDULE
  WHERE Origin IN (SELECT AirportCode
                   FROM AIRPORTS
                   WHERE CityName = 'Boston')
   AND Destination IN (SELECT AirportCode
                       FROM AIRPORTS
                       WHERE CityName = 'NYC');
```

B. Find out how many flights leave each day from each city, alphabetized by the city.

```
SELECT CityName, COUNT(DISTINCT FlightNumber) AS NumFlightsPerCity
  FROM SCHEDULE JOIN AIRPORTS
   ON SCHEDULE.Origin = AIRPORTS.AirportCode
 GROUP BY CityName
 ORDER BY CityName;
```

C. As commented in our book and as noted on
https://dev.mysql.com/doc/refman/5.7/en/datetime.html the format for a date is inside quotes and of the form year-month-day. For example, Commencement 2018 will be on '2018-05-17'
The advantage of this format is that sorting by the SQL format also gives chronological sorting.

Write a query to show all the names of all the pilots and co-pilots who worked on 2/2/18.

```
SELECT PilotGivenName, PilotFamilyName, PILOTS.ID
  FROM PILOTS JOIN ASSIGNMENTS
     ON PILOTS.ID = ASSIGNMENTS.ID
  WHERE (AssignedRole = 'pilot' OR AssignedRole = 'co-pilot') AND Date =
'2018-02-02';
```

D. Using a subquery and wildcards, write a query to find the names of all the pilots who worked in January 2018. NOTE: HERE I AM NOT INCLUDING Co-pilots. Do not repeat pilot names.

```
SELECT PilotGivenName, PilotFamilyName, DISTINCT ID
  FROM PILOTS
  WHERE ID IN (SELECT Pilot_ID
               FROM FLIGHTS
```

```
WHERE Date LIKE '2018-01-__');
```

E. Produce a table, ordered by date, which shows how many people worked in each role for the month of November 2000. You may treat pilot and copilot as separate roles. You should order by roles.

```
SELECT Date, AssignedRole, COUNT(DISTINCT ID)
  FROM ASSIGNMENTS
  WHERE Date LIKE '2000-11-__'
  GROUP BY AssignedRole, Date
  ORDER BY Date, AssignedRole;
```

F. Produce a list of those flights on 12/25/17 who departed more than 15 minutes behind schedule.
(You may read on-line sources for how to hanndle times for this question.)
If you google MySQL time functions you will come to
https://dev.mysql.com/doc/refman/5.7/en/date-and-time-functions.html

There is a function TIMEDIFF(argument1, argument2) described at
https://dev.mysql.com/doc/refman/5.7/en/date-and-time-functions.html#function_timediff

```
SELECT *
  FROM FLIGHTS JOIN SCHEDULE
   ON FLIGHTS.FlightNumber = SCHEDULE.FlightNumber
 WHERE TIMEDIFF(ActualDepartureTime,ScheduledDepartureTime) > 001500
  AND FLIGHTS.Date='2017-12-25';
```

G. What foreign keys are in the FLIGHTS table?

The foreign keys in the FLIGHTS table are:

- **FlightNumber (primary key to SCHEDULE)**
- **PlaneID (primary key to PLANES)**
- **Pilot_ID (primary key to PILOTS)**
- **CoPilot_ID (primary key to PILOTS)**
- **Attendant1_ID (primary key to ATTENDANTS)**
- **Attendant2_ID (primary key to ATTENDANTS)**

H. What if any alternate keys are there in the PILOTS table?

- **PilotLicense**

.

I. The ASSIGNMENTS table is redundant. Write a SQL statement to generate its data from the other tables.

```
SELECT Date, Role, ID
  FROM FLIGHTS, EMPLOYEES
  WHERE EMPLOYEES.ID=FLIGHTS.Pilot_ID OR EMPLOYEES.ID=FLIGHTS.CoPilot_ID
    OR EMPLOYEES.ID=FLIGHTS.Attendant1_ID OR
EMPLOYEES.ID=FLIGHTS.Attendant2_ID;
```

I'm aware that this result doesn't distinguish between Pilot and Co-Pilot or between Attendant1 and Attendant2. However, to do so would require comparing the HireDate in the EMPLOYEE table for the IDs to determine which employee is more senior and then assigning them roles. I believe that would require IF statements or some way to access the column

J . What are all the functional dependencies in AIRPORTS?

- **AirportCode ---> (CityName, NumberGatesatAirport)**
  - **AirportCode ---> CityName**
  - **AirportCode ---> NumberGatesatAirport**

K. .Is the EMPLOYEES table in 3NF? If it is explain why, and if not turn it into 3NF.

This table is not even in 2NF. First of all, HireDate definitely only depends on ID, and not Role, so it's not dependant on the whole key, as the key is (ID, Role). In fact, it's debatable whether or not DateLastRaise and Salary depend on Role either (we'd need an expert from Fly Us to give us that information). In terms of 3NF, if raises are done regularly, DateLastRaise might depend on HireDate, which is not a key and therefore a transitive dependency.

Because we're lacking some information from Fly Us, I will assume the following functional dependencies:
ID → HireDate

- **Role won't determine HireDate, just ID will**

ID → Role

- **Each employee can only have one role**

(HireDate,Role)-->(DateLastRaise, Salary)

- **First of all, I'm assuming that DateLastRaise is determined by HireDate and Role, since raises happen periodically after an employee is hired, and the length of that period depends on their role.**
- **Second, I'm assuming that HireDate and Role determine Salary. Role would determine the starting salary, and Role and HireDate together would determine the period and amount of increases to that salary in the form of raises.**

(ID, Role) → (DateLastRaise, HireDate, Salary)

- **since these columns together are a primary key, they determine all other columns**

Here's the data turned into BCNF:
EMPLOYEE(<u>ID</u>, *HireDate, Role*)
PAY_INFO(<u>HireDate</u>, <u>Role</u>, DateLastRaise, Salary)

L. Is the CITIES table in 3NF? If it is explain why, and if not turn it into 3NF.

CITIES is in 3NF already because the only non-key column, OvernightCostPersonnelForThisCity, is dependent on the key (CityName), the entirety of CityName, and nothing else but CityName.

M. Is PILOTS in BCNF? If it is explain why, and if not turn it into BCNF.

PILOTS is in BCNF. It doesn't have any overlapping candidate keys (neither candidate key- PilotLicense and ID- is composite so that would be impossible), and it's in 3NF. We know it's in 3NF because there aren't any transitive dependencies- no non-key attributes are determinants.

N . Do any of the tables have MVDs? If so, identify them and suggest an alternate design to get rid of the MVDs.

None of the tables have MVDs. I found some possibilities, such as the ASSIGNMENTS table, since each Date can have multiple AssignedRoles which each have multiple IDs to them, but it wouldn't make sense to break these up. If there were MVDs, I would suggest putting them in their own table and adding a referential integrity constraint to explain how they relate to the original table.