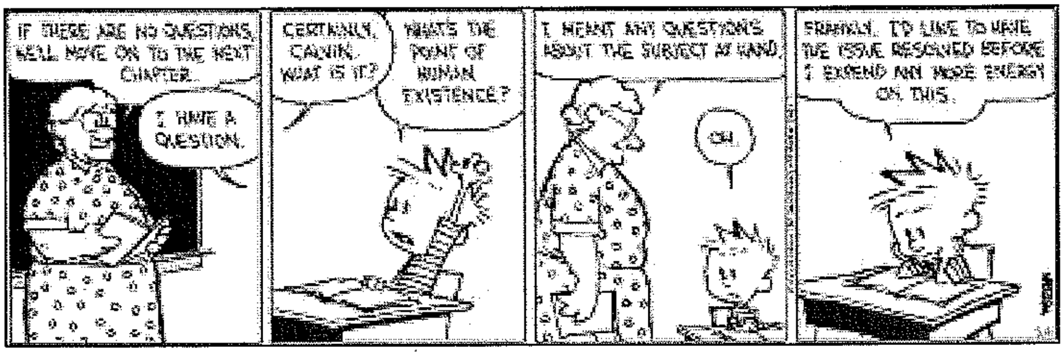
Endterm Exam
CS 122A
Winter 2015

Max. Points: 100

(Please read the instructions carefully)

Instructions:

- The total time for the exam is 120 minutes; be sure to budget your time accordingly.
- The exam is closed book and closed notes but "open cheat sheet".
- Read each question first, in its entirety, and then carefully answer each part of the question.
- If you don't understand something, ask the instructor for clarification.
- If you still find ambiguities in a question, note the interpretation you are taking.
- The very last page contains the familiar Deliber schema that virtually all of the exam questions will be based on; feel free to tear it off for easier reference during the exam.


IF THERE ARE NO QUESTIONS, WE'LL MOVE ON TO THE NEXT CHAPTER.
I HAVE A QUESTION.
CERTAINLY, CALVIN. WHAT IS IT?
WHAT'S THE POINT OF HUMAN EXISTENCE?
I MEANT ANY QUESTIONS ABOUT THE SUBJECT AT HAND.
OH.
FRANKLY, I'D LIKE TO HAVE THE ISSUE RESOLVED BEFORE I EXPEND ANY MORE ENERGY ON THIS.

NAME: WRIGHT, ANNE SEARS **STUDENT ID:** |

QUESTION	***TOPIC***	***POINTS***	***SCORE***
1	**Business Rules**	**10**	10
2	**Queries & Updates**	**40**	40
3	**Physical DB Design**	**15**	15
4	**Index Innards**	**15**	15
5	**Pot Pourri**	**20**	20
TOTAL	**All**	**100**	100

Question 1: Business Rules (10 points)

(10 pts) For each of the following Deliber business rules, each of which you're just hearing about now, circle the SQL feature that should be used to enforce it by adding its use to the Deliber schema attached at the end of the exam. If (and only if!) a given rule cannot be handled in SQL, your circled answer should be "Application", i.e., saying that this rule would need to be enforced somewhere higher up in the web application code. If there are several ways to enforce a given rule, choose the one that seems like the best (simplest and most natural) fit.

(a) (2 pts) A new table called Accidents is being added to keep track of car accidents involving Deliber drivers. A given (bad!) driver may appear in multiple rows of this table. The driver id in a row of this new table should always refer to a registered Deliber driver.

Primary key [✓ Foreign key] Not null Unique Check Trigger Application

(b) (2 pts) Social security number (ssn) is a candidate key for Drivers.

Primary key Foreign key Not null [✓ Unique] Check Trigger Application

(c) (2 pts) Each dish must have a price.

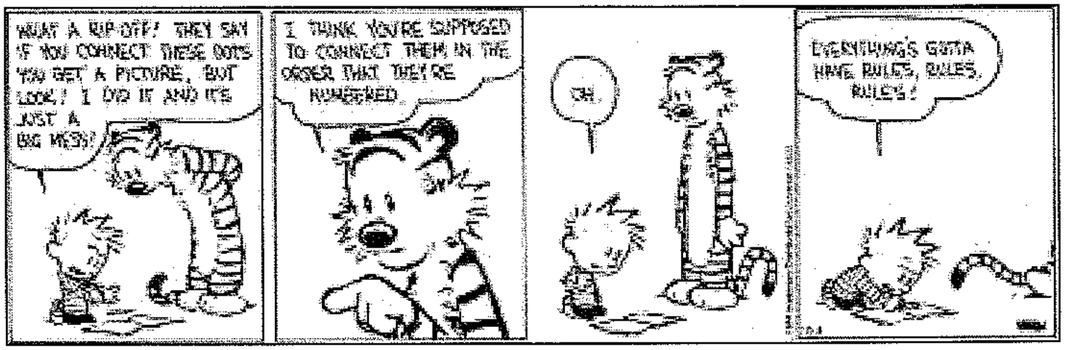
Primary key Foreign key [✓ Not null] Unique Check Trigger Application

(d) (2 pts) The total amount in Orders should automatically be kept current and should equal the sum of its dish prices times their quantities plus a 15% markup for taxes and the Deliber delivery service.

Primary key Foreign key Not null Unique Check [✓ Trigger] Application

(e) (2 pts) Rating values in customer reviews are on a scale of 1 to 10.

Primary key Foreign key Not null Unique [✓ Check] Trigger Application


WHAT A RIP-OFF! THEY SAY IF YOU CONNECT THESE DOTS YOU GET A PICTURE, BUT LOOK! I DID IT AND IT'S JUST A BIG MESS!
I THINK YOU'RE SUPPOSED TO CONNECT THEM IN THE ORDER THAT THEY'RE NUMBERED.
OH.
EVERYTHING'S GOTTA HAVE RULES, RULES, RULES!

SCORE: 10

Question 2: Queries & Updates (40 points)

Express each of the following queries (or other database actions) using SQL against the Deliber schema.

(a) (10 pts) Print a list of the Deliber drivers whose phone number is in area code 408. (Assume that the phone number format has the area code as the first three characters, e.g., a local number at UCI might be '9498240123'.) Your results should include their driver id, name, phone number, and, if they happen to also be a customer, their nickname; additionally, your results should be ordered by driver name.

```
SELECT U.id, U.name, U.phone_number, C.nickname
 FROM (Drivers D JOIN Users U ON D.did = U.id)
   LEFT JOIN Customers C ON C.cid = U.id
 WHERE U.phone_number like '408%' ORDER BY
                                      U.name;
```

(b) (10 pts) Create a view named Rated_restaurants with the schema (rid, rname, raddr, rev_cnt, avg_rating) that – for each restaurant that has gotten 10 or more reviews – lists the restaurant's id, name, address, review count, and average rating.

```
CREATE VIEW Rated_restaurants (rid, rname, raddr,
 rev_cnt, avg-rating)
 AS SELECT R.rid, R.name, R.address, COUNT(*), AVG(C.rating)
 FROM Restaurants R, Customers_review C
 WHERE R.rid = C.rid GROUP BY R.rid, R.name, R.address
 HAVING COUNT(*) ≥ 10;
```

SCORE: 20

Question 2: SQL Queries & Updates (continued)

Express each of the following Deliber queries (or other database actions) using the SQL language.

(c) (7 pts) Deliber is keeping its database clean by periodically ridding it of unpopular restaurants. Assuming that the Deliber database was created by simply executing the DDL statements shown on the last page of the exam, write a SQL statement (or statements) that will delete **all** of the information about restaurants that ***have never been ordered from by anyone***. Make sure your statement(s) will leave ***no*** dangling references in ***any*** tables when you are done.

```
DELETE FROM Restaurants R WHERE R.rid NOT IN
                                   (SELECT rid FROM Orders);

DELETE FROM Restaurants_cuisine      ··· (the same)

DELETE FROM Customers_review         ··· (the same)

DELETE FROM Dishes                   ··· (the same)
```

(d) (5 pts) How would you modify the DDL so that this periodically needed process gets simpler for the application developer? Give the **line number(s)** from the DDL and the (partial) SQL statement that you would **replace** them with.

Line	Replace with
22	add "ON DELETE CASCADE"
34	" the same
48	" the same

(e) (8 pts) A customer nicknamed 'careless123' lost his wallet (again!) and just got all new credit cards. Luckily they all have the same card numbers as before, but they also all have a new expiration date of December 2019 (*a.k.a.* '201912'). Write a single SQL UPDATE statement that will update all of this customer's on-file credit cards accordingly.

```
UPDATE Credit_cards
SET expr_date = '201912'
WHERE cid IN (SELECT ccid FROM Customers C
                    WHERE c.nickname = 'careless123');
```

SCORE: 20

Question 3: Physical DB Design (15 points)

Deliber is getting more and more popular, so its web site is starting to run to slowly, and you need to design a physical storage and indexing strategy that serves the following mixed query workload well. (I.e., when the application is running, all of these queries are happening – and you may assume they are all roughly of the same importance.) Assume that primary keys are already hash indexed (unclustered), so don't list those indexes here. Deliber is using an open source DBMS that supports both hashed and B+ tree indexes of both the clustered and unclustered varieties, so those are the choices available to you. Question marks stand for values that vary (i.e., different SSNs that are used as a search criterion)

1. SELECT bank_account_routing_number, COUNT(bank_account_number) FROM Drivers GROUP BY bank_account_routing_number ORDER BY bank_account_routing_number;
2. SELECT U.name, U.phone_number FROM Drivers D JOIN Users U ON (D.did = U.id) WHERE D.ssn < ? AND D.ssn > ?;
3. SELECT * FROM Drivers D WHERE D.ssn = ?;
4. SELECT * FROM Customers_review;
5. SELECT C.customer_comment FROM Customers_review C WHERE C.rating = ?;
6. SELECT * FROM Customers_review WHERE customer_comment = ?;
7. SELECT C.rating, C.customer_comment FROM Customers_review C, Restaurants R WHERE C.rid = R.rid AND R.name = ?;

(a) (6 pts) List the indexes that you recommend creating on the ***Drivers*** table. For each index that you recommend, give the column(s) that it should index on, the type of each index, and whether or not it should be clustered. (In your design, don't index a given attribute more than once.)

Indexed Column(s)	Hashed or B+ Tree?	Clustered? (Y/N)
bank_account_routing_number, bank_account_number	B+	N
ssn	B+	Y

(b) (9 pts) List the indexes that you recommend creating on the ***Customers_review*** table. For each index that you recommend, give the column(s) that it should index on, the type of each index, and whether or not it should be clustered. (In your design, don't index a given attribute more than once.)

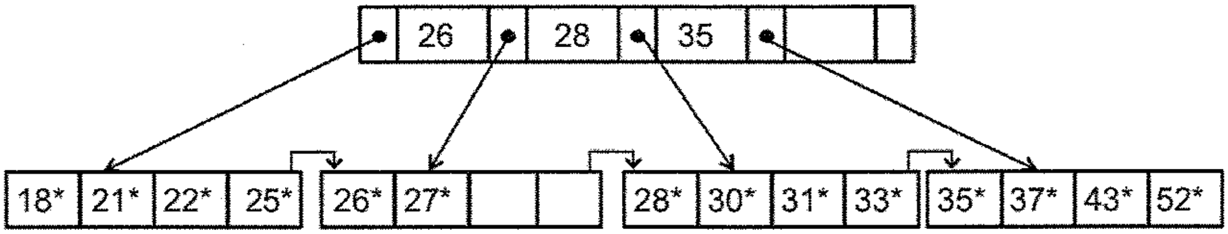
Indexed Column(s)	Hashed or B+ Tree?	Clustered? (Y/N)
rating	Hashed	●
customer_comment	Hashed	N
rid	Hashed	●

either or

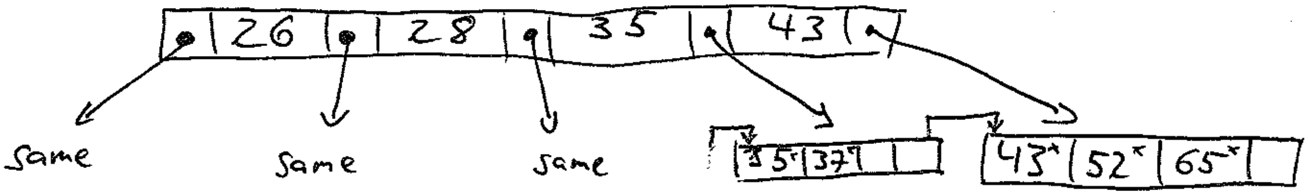
SCORE: 15

Question 4: Index Innards (15 points)

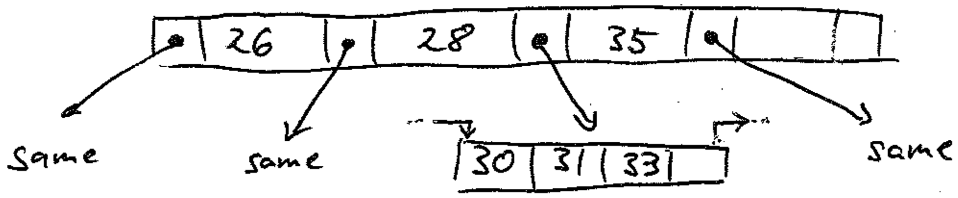
Consider the following example of a B+ tree index of order ***d***=2. Suppose that this is a dense secondary B+ tree index on Users.age, a new field that has recently been added to the Users table using an ALTER TABLE command, and assume that its leaf entries contain key/record-id-list pairs.



(a) (6 pts) Draw the B+ tree index that would result from performing the following operation on the ***original index*** pictured above: INSERT INTO Users (id, name, phone_number, age)
VALUES (301, 'I. M. Old', '4085551213', 65);



(b) (6 pts) Draw the B+ tree index that would result from performing the following operation on the ***original index*** pictured above: DELETE FROM Users WHERE age = 28;



(c) (2 pts) In real life, due to variable-length keys but fixed-size disk pages, the notion of order ***d*** for a B+ tree is replaced with a different criteria for making decisions about when to split and merge nodes. Circle the tree node condition below that replaces the test for *num_entries* = ***d*** in real B+ tree code.

Node has 2 entries   Node is full   Node has 1 entry   [✓ Node is half-full]

(d) (1 pt) Is the above index clustered or unclustered?

Clustered   Unclustered   [✓ Can't tell from this picture]

SCORE: 15

Question 5: Pot Pourri (20 points)

(a) (4 pts) Transaction management is an important feature of a DBMS. Which of the following are things about transactions that make their support desirable to an application programmer? (Put an X in the box next to each of the right answers.)

- □ They make queries more declarative and therefore programmers more productive.
- ☒ They relieve programmers of having to write complex synchronization code to handle cases that can arise when multiple users try to access the same data.
- □ They relieve programmers of having to use check constraints and foreign keys to keep the database data consistent.
- ☒ They relieve programmers of having to worry about the partial effects of updates on a database when failures such as software errors or power failures or deadlocks arise.

(b) (4 pts) As a way to remember what they are about, transactions are often said to have the "ACID" properties. ACID is a four-letter acronym that's short for Atomic, Consistent, Isolated, and Durable. Match the appropriate property letter (A, C, I, or D) in this acronym with one corresponding phrase or description below. (You should use each letter exactly once.)

A All or nothing execution.

D Once committed, always remembered.

C Programmers write programs to individually do the right thing when run alone.

I Locking is the usual way that database systems provide this property.

(c) (2 pts) For security purposes, a DBA can arrange things so that no users are authorized to access anything in a database by default. Assuming the existence of a database user named 'Driver', write a SQL statement to permit user 'Driver' to update (only!) the bank account information in the Drivers table. (*Note:* If a user tries to access data that they aren't authorized to access, the attempt to do so will fail and their statement or transaction will be aborted.)

GRANT UPDATE (bank_account_number,
bank_account_routing_number)

ON Drivers TO 'Driver'

SCORE: 10

Question 5: Pot Pourri (cont.)

Deliber is still considering migrating their web site to AsterixDB in the near future, so they have an exploratory team with a few of their top engineers building a prototype.

(d) (4 pts) Which of the following could be among the set of valid reasons given by the Deliber engineers to Deliber's management team in making the case for such a migration? (Check all that apply.)

- ☒ Their current open source relational solution will not scale to a very large user base.
- □ They want to use Pig instead of SQL for writing their queries.
- ☒ The SQL data model and its design rules are causing their application to perform too many joins, and they would like to be able to violate 1NF in certain sensible places to gain performance.
- ☒ Their previous database model is very rigid, and the "NoSQL" approach will allow them to add new information to database records without having to make ALTER TABLE / ADD COLUMN schema changes every time they have new information capture needs.

(e) (5 pts) Below is an AQL query that the exploratory team has written against their AsterixDB schema, which is the same schema you dealt with in the last homework assignment. Write a SQL query against Deliber's current relational schema that will produce the same result as this AQL query does.

```
for $r in dataset Restaurants
for $c in $r.cuisine
where $c = "Greek"
order by $r.name
return {"name": $r.name, "food": $c, "addr": $r.address}
```

```
SELECT R.name, C.cuisine_type AS food, R.address
AS addr FROM Restaurents R, Restaurants_cuisine C
WHERE R.rid = C.rid AND C.cuisine_type = 'Greek'
ORDER BY R.name
```

(f) (1 pt) Below is another AQL query (and a sample result) that the exploratory team has written against their AsterixDB schema. Can this query also be exactly expressed in SQL? Circle **YES** or **NO** below.

```
for $r in dataset Restaurants
where some $c in $r.cuisine satisfies $c = "Greek"
order by $r.name
return {"name": $r.name, "food": $r.cuisine, "addr": $r.address}
```

```
...... {"name": "Der Houstons", "food": {{"American", "Greek"}}, "addr":
"1592603 Main Street, Ste. 500, Irvine, CA, 92614-426199"} ......
```

YES [✓ **NO**]

(because of nested values)

SCORE: 10

Deliber Database Schema

```
CREATE TABLE Users ( id INT,
name    VARCHAR(40),
phone_number  VARCHAR(20),
PRIMARY KEY (id) );

CREATE TABLE Credit_cards (
card_number VARCHAR(20),
expr_date CHAR(6),  cid INT,
PRIMARY KEY (card_number),
FOREIGN KEY (cid) REFERENCES
Customers(cid) );

CREATE TABLE Restaurants (rid INT,
name VARCHAR(40),
address VARCHAR(100),
bank_account_number VARCHAR(20),
bank_account_routing_number
VARCHAR(20),
last_bank_transaction_datetime
DATETIME,
PRIMARY KEY (rid) );

CREATE TABLE Customers_review (
cid INT,  rid INT,  rating INT,
customer_comment VARCHAR(100),
PRIMARY KEY (cid,rid),
FOREIGN KEY (cid) REFERENCES
Customers(cid),
FOREIGN KEY (rid) REFERENCES
Restaurants(rid) );

CREATE TABLE Orders_Contain_Dishes (
oid INT,  rid INT,
name VARCHAR(40),
quantity INT,
PRIMARY KEY (oid, rid, name),
FOREIGN KEY (oid) REFERENCES Orders
(oid),
FOREIGN KEY (rid, name) REFERENCES
Dishes(rid, name) );

CREATE TABLE Dishes (  rid INT,
name VARCHAR(40),
price DECIMAL(6,2),
PRIMARY KEY (rid, name),
FOREIGN KEY (rid) REFERENCES
Restaurants(rid) );

CREATE TABLE Customers (  cid INT,
address VARCHAR(100), nickname VARCHAR(40),
PRIMARY KEY (cid),
FOREIGN KEY (cid) REFERENCES Users(id) );

CREATE TABLE Drivers (  did INT,
ssn CHAR(9),
bank_account_number VARCHAR(20),
bank_account_routing_number VARCHAR(20),
PRIMARY KEY (did),
FOREIGN KEY (did) REFERENCES Users(id) );

CREATE TABLE Restaurants_cuisine (  rid INT,
cuisine_type VARCHAR(20),
PRIMARY KEY (rid, cuisine_type),
FOREIGN KEY (rid) REFERENCES Restaurants(rid) );

CREATE TABLE Orders (  oid INT,
cid INT,  did INT,  rid INT,  order_datetime DATETIME,
total_amount DECIMAL(7,2),
PRIMARY KEY (oid),
FOREIGN KEY (cid) REFERENCES Customers(cid),
FOREIGN KEY (did) REFERENCES Drivers(did),
FOREIGN KEY (rid) REFERENCES Restaurants(rid) );

CREATE TABLE Orders_track_status (  oid INT,
order_status_code INT,
order_status_datetime DATETIME,
PRIMARY KEY (oid, order_status_code),
FOREIGN KEY (oid) REFERENCES Orders(oid),
FOREIGN KEY (order_status_code) REFERENCES
Orders_status_code_text(order_status_code) );

CREATE TABLE Orders_status_code_text (
order_status_code INT,
order_status_text VARCHAR(40),
PRIMARY KEY (order_status_code) );
```

SCORE: ________