

CSE 344 Midterm Exam

Friday July 21, 2017

Name: _____

Student Number: _____

Problem	Points
1	36
2	20
3	28
4	16
Total	100

- **Do not open** the test until instructed to do so.
- Please read all instructions carefully. You may ask the instructor clarifying questions during the exam.
- This is a closed book exam, but you are allowed one (double sided) page of notes.
- Please silence all cell phones and place them off the table.
- There are 4 questions each with multiple parts. If you get stuck on a question move on and come back to it later. Partial solutions will be graded for partial credit.
- You have 60 min to work on this exam.
- If you need, scratch paper is provided at the front of the room. Good Luck!

This exam uses relations from a database that a gym company uses to track membership and usage between multiple gyms in many different cities. The relations in this database are:

Gym (gid, name, city)

Member (mid, name, is_student, birthdate, city)

Visits (timestamp, mid, gid)

The underlined attributes are primary keys for each relation. The attributes *mid* and *gid* in Visits reference the keys in Member and Gym. Some assumptions you can make about the table are:

- Max length of Member.name and Member.city is 50 characters.
- Member.is_student is a Boolean field (not an Int that can be 0 or 1).
- Member.birthdate is a Date field.
- Visit.timestamp is a Datetime field with the timestamp of the visit.

Question 1: SQL Statements (36 Points)

Part A: Schema Creation (10 Points)

- a) Write the SQL Create statement for Member and Visits.

Part B: Queries (26 Points)

Each item below describes a query to perform on the database using the schema from the previous page. Provide an SQL statement for each query. For reference we provide the SQLite date functions:

```
date(timestamp/date) => "YYYY-MM-DD"  
date('now') => "2017-07-21" (current date)  
year(timestamp/date) => "YYYY"
```

The following SQL would find all gym visits that occurred in 2017 or on July 4, 2016:

```
SELECT * from Visits where date(timestamp) = "2016-07-04" OR  
year(timestamp) = "2017";
```

- a) Find members who have attended two different gyms on the same day. Return only unique member ids in any order. (10 points)

- b) Find non-student members who have visited any gym less than 10 times. For each member return their name, number of times they visited a gym and first date they visited the gym (3 attributes). Include members who have never visit a gym. Order by decreasing visit count (high → low) and then by increasing name (A → Z). (10 points)

- c) Indicate which of the following queries correctly finds all members who did not visit any gyms outside of their city: (All queries are syntactically correct). (6 points)

Q1: `SELECT DISTINCT M.mid FROM Member M
WHERE NOT EXISTS (
 SELECT * FROM Visits V, Gym G
 WHERE V.mid = M.mid AND V.gid = G.gid
 AND G.city <> M.city
);`

Q2: `SELECT DISTINCT M.mid FROM Member M, Visits V
WHERE M.mid = V.mid
AND NOT EXISTS (
 SELECT * from Gym G
 WHERE V.gid = G.gid AND G.city != M.city
)`

Q3: `SELECT DISTINCT M.mid
FROM Member M
LEFT OUTER JOIN Visits V ON M.mid = V.mid
LEFT OUTER JOIN Gym G ON V.gid = G.gid AND M.city <>
G.city
GROUP BY M.mid
HAVING count(*) = 0`

Q4: `SELECT DISTINCT M.mid
FROM Member M
LEFT OUTER JOIN Visits V ON M.mid = V.mid
LEFT OUTER JOIN Gym G ON V.gid = G.gid AND M.city <>
G.city
GROUP BY M.mid
HAVING count(G.gid) = 0`

Answer with any subset of Q1, Q2, Q3, Q4: _____

Question 2: Cost Estimation and Indexes (20 Points)

The following database statistics exist for the Member Table

$T(\text{Member}) = 500$

$V(\text{Member}, \text{city}) = 10$

$B(\text{Member}) = 100$

$V(\text{Member}, \text{is_student}) = 2$

Reminder: The number of tuples returned by $\sigma_{a=?}(R)$ is equal to $T(R) * 1/V(R,a)$

Part A: Indexes (10 Points)

Consider the following simple SFW query:

```
SELECT name FROM Member
WHERE city = "Seattle WA" AND is_student = True;
```

Estimate cost in disk I/Os of the query in the following conditions (assume all indexes are used):

- There is no Index
- There is an unclustered index Member(City)
- There is a clustered index Member(City)
- There is an unclustered index Member(is_student)
- There is an unclustered index Member(City,is_student)

Part B: Joins (10 Points)

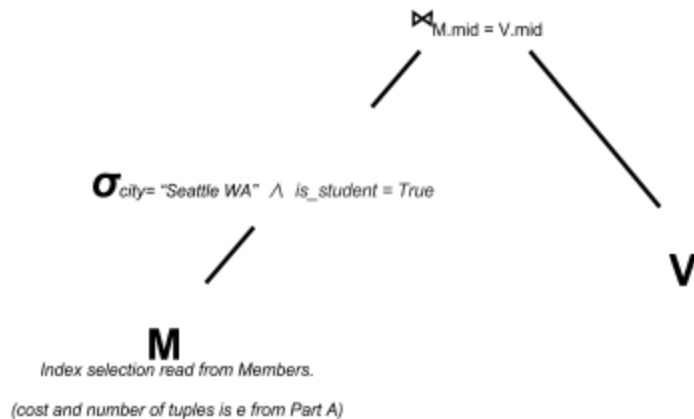
We now want to count how many times students in Seattle went to a gym with this query:

```
SELECT name, count(*) FROM Member M, Visits V
WHERE M.mid = V.mid AND city = "Seattle WA" AND is_student =
True
GROUP BY name;
```

The Relation Algebra for this join statement is below. However, multiple types of join algorithms can be used. Estimate the cost for a nested loop join with and without an index on Visits(mid). In both cases the Members table will be scanned using the Member(City,is_student) index from the previous question. No temporary tables are written to disk. The cost of reading M is therefore your answer from e in Part A - if part e was wrong you will not lose points again here.

B(Visits) = 400
T(Visits) = 5000

V(Visits,mid) = 500



a) What is the cost of a Nested Loop join with V

b) What is the cost of a Nested Loop join with Visits using an unclustered index Visits(mid)

Question 3: Relational Calculus and Datalog (28 Points)

For this section we will use these simplified versions of the gym relations:

`Member (mid, name, city)`

`Visits (ts, mid, gid)`

`Gym (gid, city)`

Part A: Relational Calculus (8 Points)

State in English what each of the following relational calculus queries return:

a) $P(x) = \exists a. \exists b. \exists g. \exists c. (\text{Member}(x, a, c) \wedge \text{Visits}(b, x, g) \wedge \text{Gym}(g, c))$

b) $P(x) = \exists a. \exists c. \text{Member}(x, a, c) \wedge (\forall g. (\text{Gym}(g, c) \Rightarrow \exists b. \text{Visits}(b, x, g)))$

Part B: Datalog (10 Points)

- a) Write a Datalog program that finds the names of member pairs who only visit gyms on the same day. (Note: this can be any gym but must be the same day). Assume that the function `Date(ts)` returns "YYYY-MM-DD".

Part C: Relational Algebra (10 Points)

- a) Write a Relational Algebra expression in the form of a logical query plan (i.e., draw a tree) that is equivalent to the query below. Your plan does not have to be “optimal”: however, points will be taken off for overly complex solutions.

```
SELECT M.name, count(*) AS cnt
FROM Member M, Visits V, Gym G
WHERE M.mid = V.mid AND G.gid = V.gid
      AND G.city = M.city
GROUP BY M.name
HAVING count(*) > 20;
```


Question 4: Short Answer (16 Points)

- a) You and a friend go to the gym from Question 1. You arrive at the same time and sign in together. However, the following error is raised:

Error: Duplicate entry '1500602520' for key 'PRIMARY'

What did the DBA do wrong when creating the tables to cause this error? How would you fix the table schema?

- b) Signing someone in at the gym is now taking a very long time. You are hired to optimize the database. Inspecting the schema you find many indexes on all the fields, including a clustered index on Visit(mid). Why would this slow down creating a new Visit? What information would you need to decide which indexes to keep and which to remove?

- c) Describe the difference between a logical query plan and a physical query plan. At what stage of query execution is each used?

- d) Split the following five query languages into two groups based on expressive power. All languages in each group should have equivalent expressive power. The expressive power of all languages in Group Two should be greater than those in Group One.

Relational Algebra (RA)
extended RA
Relational Calculus (RC)
SQL
Datalog_⊆ (no recursion)

Group One



Group Two