NEVER ASK THIS MANY QUESTIONS AGAIN. Ask 1/2 at most

UNIVERSITY OF VICTORIA
Faculty of Engineering
Department of Computer Science

CSC 370 (Database Systems)
Instructor: Daniel M. German

Final Exam
Aug 15, 2012

Duration: 2 hrs

This is a closed-book exam.

This examination paper consists of **9** pages and **5** sections. Please bring any discrepancy to the attention of an invigilator. The number in parenthesis at the start of each question is the number of points the question is worth.

Answer all questions.

Please write your answers clearly.

For instructor's use:

	Score
1 (27)	
2 (10)	
3 (10)	
4 (9)	
5 (14)	
Total (70)	

For this exam, consider the following schema of a simple university database. It includes information about instructors, students, and the courses offered. Feel free to remove this page from the exam.

- The **Students** table contains the id of the student, his/her name, age, and gpa.

```
Students(sid: integer, sname: string,
         age: integer, gpa: real)
```

  - Primary Key: **sid**

- The **Instructors** table contains information about instructors of the courses: their id, name and department they belong to.

```
Instructors(iid: string, iname: string, dept: string)
```

  - Primary Key: **sid**

- The **Courses** table contains information about courses: their id, their name, the department that offers it, the id of its instructor, and the maximum number of students who can take it (**maxenrol**).

```
Courses(cid: string, cname: string,
        dept: string, iid: string,
        maxenrol:integer
)
```

  - Primary Key: **cid**
  - Foreign Key: **iid** references **Instructors**

- The table **Enrolled** contains what students are registered to which courses, and the grade they receive (NULL if they have not received one yet).

```
Enrolled(sid: integer, cid: string,
         grade: integer)
```

  - Primary Key: (**sid,cid**)
  - Foreign Key: **sid** references **Students**
  - Foreign Key: **cid** references **Courses**

1. **Relational Algebra and SQL**

For each of the following questions, provide a relational algebra expression to answer them, and its equivalent SQL query:

(a) [3] List the **iid** of instructors that are not teaching any course.

$\Pi_{iid}\, I - \Pi_{iid}\, C$

```
select iid from Instructors
  EXCEPT
select iid from Courses.
```

(b) [3] List the **sId** of those students who are older than student with **sId** = 12345;

$\Pi_{sid}\ \sigma_{age > \gamma_{max(age)} S}\ S$

```
select sid from students where
  age > ( select max(age) from students)
```

(c) [3] List the **cId** of all courses that have at least one student without a grade (its grade is NULL). Make sure you list each course only once (you cannot use DISTINCT).

$\gamma_{cid}\ C \bowtie (\sigma_{grade\ is\ NULL}\ E)$

```
Select cid from
Courses natral join (select * from
 Enrolled where grade is NULL) as A
group by cid
```

(d) [3] List the **sId** of students who are taking exactly 3 courses.

$\Pi_{sid}\ \sigma_{count=3}\ \gamma_{sid,\ count(*)\Rightarrow count}\ E$

```
select sid from Enrolled
group by sid
having count(*) = 3;
```

(e) [3] For every student **sId** in the database, list the number of courses they are enrolled in (make sure you include students who are not taking any courses, i.e. the number of courses they are enrolled in is zero).

$\gamma_{sid,\ count(*)}(S \overset{o}{\bowtie} E)$ can be left join

```
select sid, count(*) from
S Natural full join E
```

(f) [3] List the **sId**s of the students who are enrolled in the most number of courses (there might be one or more student).

$N = \gamma_{sid,\ count(*)\rightarrow c}\ E$ count per student

$M = \gamma_{max(c)\rightarrow c}\ N$ max number of courses by one student

$\Pi_{sid}\ N \bowtie M$

```
WITH N AS ( select sid, count(*) as c from enrolled
            group by sid),
WITH M AS ( select max(c) as c from N)
SELECT sid from N NATURAL JOIN M;
```

(g) [3] What is the average **age** of the students who are taking at least one course? Make sure you average each student only once.

$\gamma_{avg(age)}\ \sigma_{sid\ IN\ (\pi sid\ E)}\ S$

```
SELECT avg(age) FROM Students
where sid in (select sid in Enrolled);
```

(h) [3] List the **sid** of students that are taking **cid** *csc370* and *seng401*?

$\pi_{sid}\ \sigma_{cid='csc370'}\ E\ \cap\ \pi_{sid}\ \sigma_{cid='seng401'}\ E$

```
select sid from Enrolled where cid = 'csc370'
INTERSECT
select sid from Enrolled where cid = 'seng401';
```

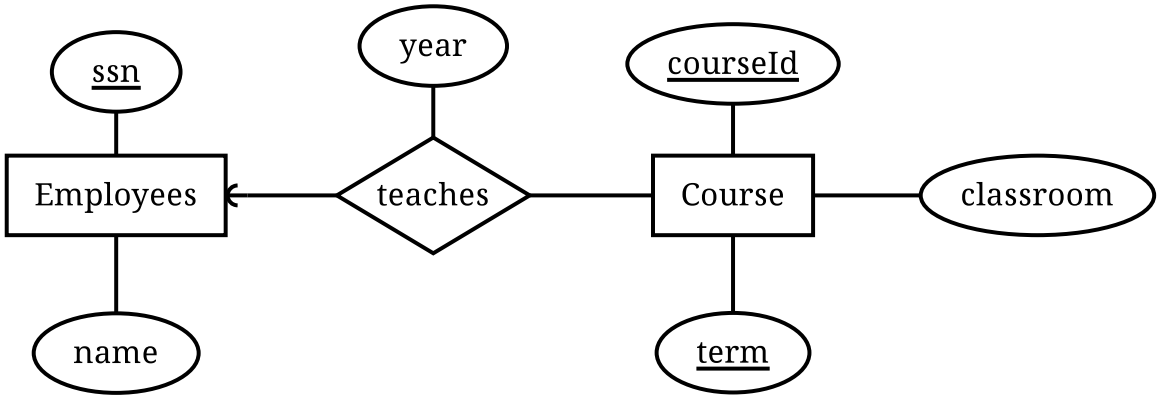
(i) [3] Assume a relation $R(a, b)$. Write a query that projects the values of a, but each value of a is output only once (duplicates are removed). You can not use DISTINCT.

$\gamma_a\ R$

```
select a from R group by a;
```

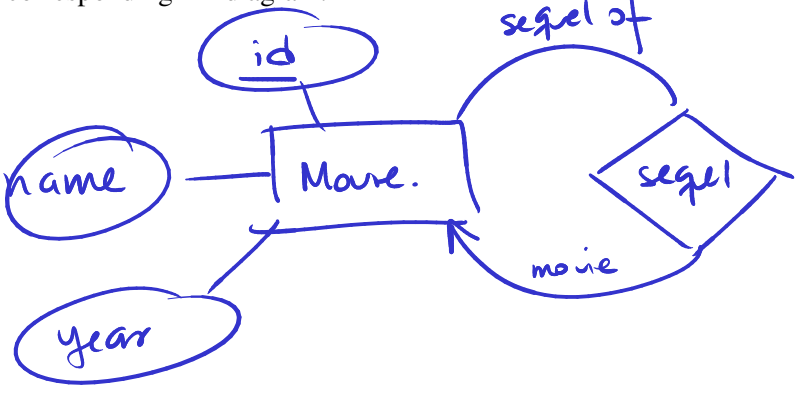
2. **Relational Model**

(a) [6] Convert the following ER-diagram to `CREATE TABLE` statements. Make sure you include all the necessary constraints. For the sake of simplicity, you can assume all the attributes are type `CHAR(100)`.




```
CREATE TABLE Employees (
    ssn char(100)
    name char(100),
    PRIMARY KEY (ssn));
CREATE TABLE Course_Teaches (
    courseid char(100),
    term char(100),
    classroom char(100)
    year int,
    instructor char(100) NOT NULL,
    PRIMARY KEY (courseid, term)
    FOREIGN KEY (instructor) references employees );
```

(b) [4] We are currently designing a database. We have one entity: a *movie*, with three attributes: *id*, and *name*, and *year*. There is also a relationship between two movies, called *Sequel*. A movie can be a sequel of at most one movie. A movie can have any number of sequels. Draw the corresponding ER diagram.



3. **Security**

Our university database has the following users: Alice (who is the owner of the table *Students*), Peter, John, and Ben.

(a) [3] Ben will be the owner of the table *Enrolled*. List the minimum privileges that he will need to obtain in order to be able to create such table. Explain your answer.

CREATE on DB
REFERENCES on student.

(b) [2] Peter is expected to execute the query `SELECT * from Enrolled`, what are the minimal set of privileges he needs? select on enrolled

(c) [3] Alice runs the following commands:

```
GRANT SELECT ON Students TO John with GRANT Option;

GRANT SELECT ON Students TO Peter;
```

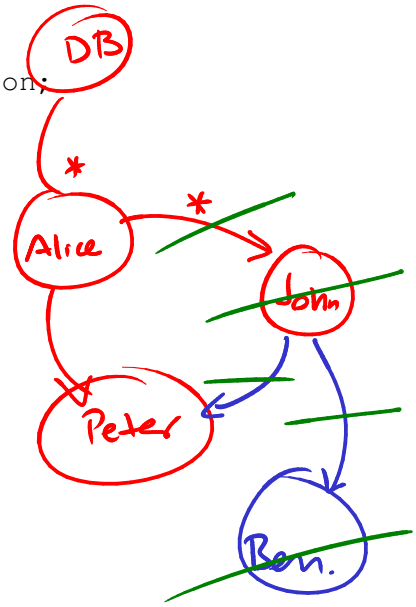
John then runs the following commands:

```
GRANT SELECT ON Students TO Peter;
GRANT SELECT ON Students TO Ben;
```

Finally Alice decides to run:

```
REVOKE SELECT on Students FROM John CASCADE;
```

Draw the resulting grant diagram.



(d) [2] After all these commands are executed, which users have the SELECT privilege on Students?

Alice and Peter

4. **Transactions**

Assume that the Courses database is currently in use at our university and that transactions exist to modify the database (add, remove, edit remove students, courses and instructors). These transactions always run as serializable. Many instances of the same transaction can happen at the same time. For each of the following transactions and isolation level, explain your answers.

- The impact of the specified isolation level will have on the query.
- Under the indicated isolation level, is the query always serializable? Why? (In other words, will the effect of this transaction be the same as if no other transactions run at the same time?)

(a) [4] Return the number of available spaces in a course (**maxEnrol** minus the number of students enrolled in that course). Run as `READ UNCOMMITTED`.

1) We will get an imprecise # of spaces available if students register / drop at the same time.

2) No. It is not serializable.

(b) [5] Change the name of an existing course (updates **cname**). The transaction does not need to read the current name (i.e. the new name does not depend upon the previous name). Assume only this type of transaction can change the name of a course, and multiple instances of this transaction can exist. Run as `READ COMMITTED`.

1) Since the new name of the course does not depend on other tuples (it does not get generated with the output of a query) we are not concerned with other trans. in progress, except if they also modify the same course name. If that is the case this transaction reads committed, so it waits for the other to complete.

2) Yes

5. **Indexes**

The **Students** table contains one million records. It is stored as an unsorted data file. Student's **sids** range from 1 to 1,000,000. The table is stored in a dense B+tree index on **sid** with a height of 3. Each data block can hold at most 100 records. There is no waste in the data file.

The table has a secondary index on **age** (unclustered B+tree). 4% of the records of **Students** have **age** equal NULL. The remaining 96% contain values within the range 16-39 (it includes both 16 and 39). Assume **age** is uniformly distributed within this range. The index data blocks can hold at most 500 records. On average, only 2/3 of the space in the index blocks is used.

For each of the following queries.

(a) [2] What is the number of blocks read in a sequential scan? Show all your work.

$$B(R) = \frac{|R|}{100} = \frac{10^6}{10^2} = 10^4$$

(b) For each of the following queries, give two potential access paths. Calculate the cost (in number of block reads) of each path. Show all your work.

i. [6] select * from Students where sid = 12345;

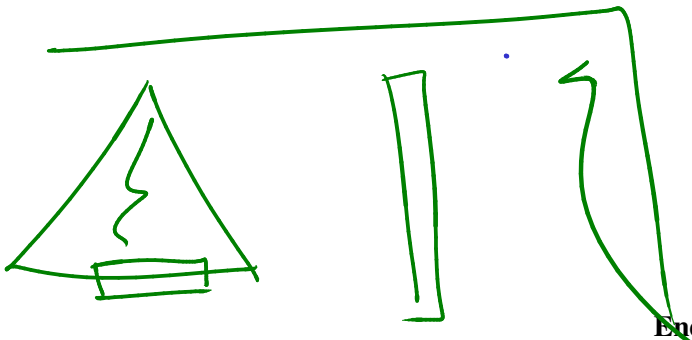
use seq. scan.

use index. One matching tuple.

so $h+1 = 4$ (sid is index)

ii. [6] select * from Students where age = 20;

seq scan



use index.

Matching tuples: $\begin{matrix} 39 \\ -\ 16 \\ \hline 23+1 \end{matrix}$ } 24 values + NULLs

24 values in 96% tuples.

age = 20 $\Rightarrow \frac{96\%}{24} = 4\%$

selectivity of age = 20 4%

$\Rightarrow$ #matching tuples $= 10^6 \cdot 0.04 = 4 \cdot 10^4$ tuples

$h-1$ + blocks + #matching tuples
index

blocks in index matching tuples

$\frac{40,000}{500 \cdot \frac{2}{3}} = \frac{1.2 \cdot 10^5}{10^3} = 1.2 \cdot 10^2$

height $= \lceil \log_{333} 10^6 \rceil = 3$.

$\Rightarrow 3 - 1 + 120 + 4 \cdot 10^4 = 40,122$ blocks.

End of examination
Total pages: 9
Total marks: 70