**UNIVERSITY OF VICTORIA**
**Faculty of Engineering**
**Department of Computer Science**

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

**Final-midterm Exam**
**April 4, 2014**

**Duration: 50 minutes**

**This is a closed-book exam.**

This examination paper consists of **7** 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 (5) | |
| 2 (4) | |
| 3 (6) | |
| 4 (6) | |
| 5 (6) | |
| Total (27) | |

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 (**sid**), his/her name (**sname**), and birthday.

  ```
  Students(sid: integer, sname: string,
           birthday: date)
  ```

  - Key: **sid**

- The **Instructors** table contains information about instructors of the courses: their id (**iid**), and name (**iname**). An instruct can teach many different courses.

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

  - Key: **iid**

- The **Courses** table contains information about courses: their id (**cid**), their name (**cname**), the id of its instructor (**iid**), and the maximum number of students who can take it (**maxenrol**). Every **iid** in this table is also found in the table **Instructors**. Every course must have an instructor. Students can register in as many courses as they want. Courses can have as many students as needed.

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

  - Key: **cid**

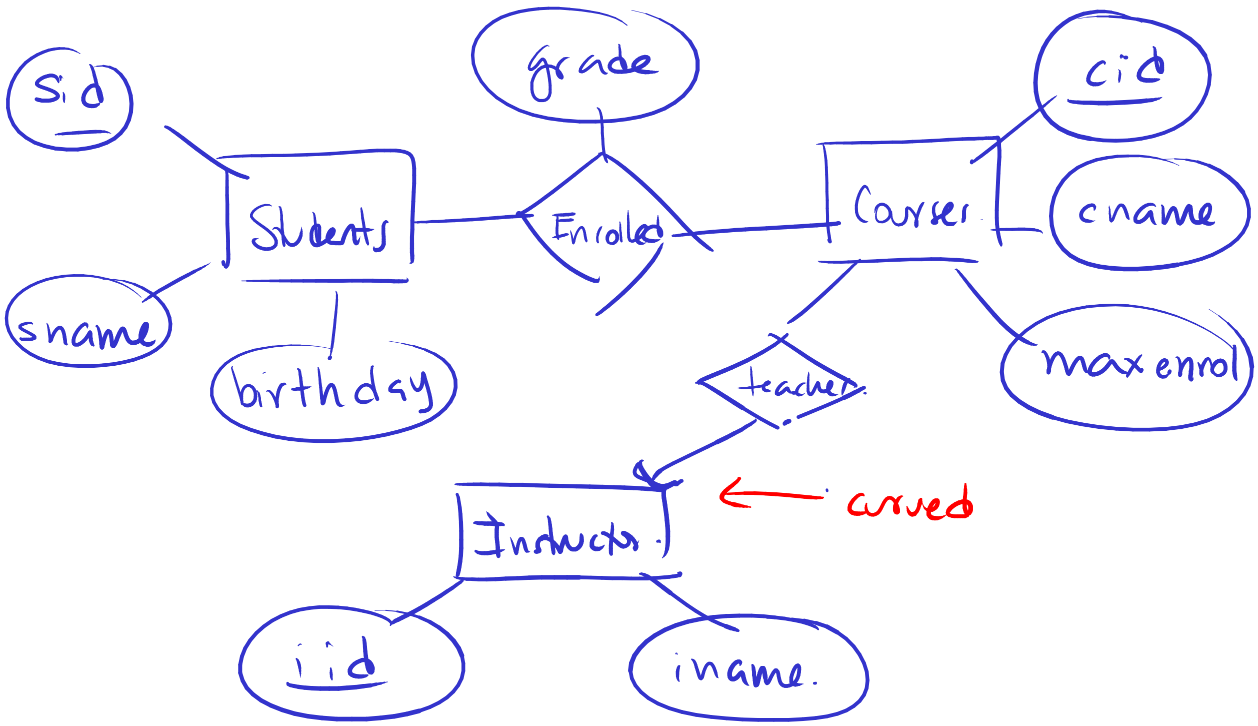
- The table **Enrolled** contains what students are registered to which courses, and the grade they receive (NULL if they have not received one yet). A student can only register once to any given course, but he/she can register to as many courses as necessary. Neither **sid** nor **cid** can be NULL. Every **sid** in this table is also found in the table **Students**, and every **cid** in this table is also found in the table **Courses**.

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

  - Key: (**sid,cid**)

1. [5] **Entity-Relationship**

   Create an entity-relationship diagram that reflects the University database (see page 2).



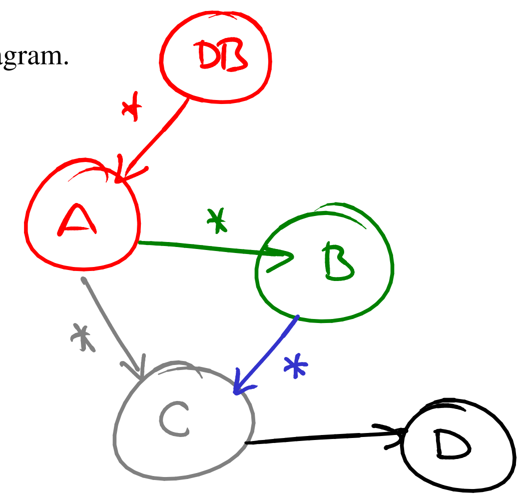
2. **Security**

Assume we have table R and users A, B, C and D. Assume the following sequence of statements (executed in the order given):

A is creator

- user A executes *GRANT select on R to B with grant option*
- user A executes *GRANT select on R to C with grant option*
- user B executes *GRANT select on R to C with grant option*
- user C executes *GRANT select on R to D*

(a) [2] Draw the resulting grant diagram.




(b) [2] User A executes *revoke select on R from C restrict*. After this command is executed, what is the resulting grant diagram?

Same. The command will fail because C has children!

3. **Transactions**

Assume the following transactions:

- $T_1$ = R(A,x), x+= 10, W(A,x), R(B,x), x+=20, W(B,x)
- $T_2$ = R(A,x), x+= 5, W(A,x)

(a) [2] Is the following schedule serializable? Explain your answer.

| $T_1$ | $T_2$ |
|---|---|
| R(A, x) | |
| x+=10 | |
| W(A,x) | |
| | R(A,x) |
| R(B,x) | |
| | x+= 5 |
| | W(A,x) |
| x+20; | |
| | Commit |
| W(B,x) | |
| Commit | |

Yes

Effect of Serial $T_1T_2$ or $T_2T_1$ $\Rightarrow$ A + 15, B + 20

Effect of this schedule $\Rightarrow$ A + 15, B + 20

$\Rightarrow$ equiv. to a Serial Schedule.

(b) [2] In the previous schedule replace the *commit* of $T_1$ with *abort*. Is the schedule serializable? Explain your answer.

No

Effect of Serial Schedule.

$T_1T_2$, $T_2T_1$ = $T_2$ $\Rightarrow$ A+5, B

Effect of this one: A+15, B Diff. than

(c) [2] In schedule above (question a) replace the *commit* of $T_2$ with *abort*. Is the schedule serializable? Explain your answer.

Yes.

Effect of serial: $T_1T_2$, $T_2T_1$ = $T_1$ $\Rightarrow$ A + 10, B + 20

Effect of schedule: A+10, B+20.

$\Rightarrow$ Serializable.

4. **Indexing and Query Evaluation**

Assume that we have a table R(a, b, c) with primary key a. Assume that the table contains $10^6$ tuples. The data file contains 100 tuples per block.

(a) [2] Assume that there is only a sparse index on $a$. The index has on average 1,000 values per index block. We need to evaluate the query $\sigma_{a>10}R$. We are told that 1,500 tuples match this query. What is the value that best approximates the cost of this query in terms of block reads?

i. 3
ii. 17
iii. 152
iv. 1503
v. 5

h

$\frac{1500}{100} = 15$ blocks

$h = \lceil \log_{1000} 10^4 \rceil = \left\lceil \frac{\log 10^4}{\log 1000} \right\rceil = 2$

$15 + 2 = 17$

(b) [2] Assume that there is only a dense index on $b$. Assume that the index has a height of 3. Assume that the index has on average 500 key values per block. We must evaluate the query $\sigma_{b>100}R$. What is the minimum number of tuples that this query should match such that the cost of the sequential scan is cheaper than the cost of using the index? Choose the best approximation:

i. $\geq$ 1,000
ii. $\geq$ 10,000
iii. $\geq$ 100,000
iv. $\geq$ 50,000
v. $\geq$ 5,000

$\sigma_{b>100} R$

Cost of seq scan $= \frac{|B|}{100}$

Cost. Seq scan $= \frac{10^6}{10^2} = 10^4$

Cost of Index > Cost Seq. Scan.

(c) [2] Assume that we compute $R \bowtie S$ using a block-based nested loop join. Assume $B(R)$ is much larger than $B(S)$, and $B(S)$ is much larger than M (M is the amount of memory available to do the join). What is the difference in the cost of this join (in terms of block reads) if we choose $S$ as the outer table instead of $R$? Choose the value that best approximates.

i. 0
ii. $B(S)$
iii. $|B(R) - B(S)|$
iv. $B(S) + B(R)$
v. $B(R)$

Cost of join

$C_1 = B(R) + \frac{B(R) \cdot B(S)}{M-1}$

or

$C_2 \quad B(S) + \frac{B(R) \cdot B(S)}{M-1}$.



$C_1 - C_2 = B(R) - B(S)$

:

5. **SQL**

Answer the following questions in both relational algebra and SQL. Your relational algebra should match your SQL.

(a) List the course id, course name, the number of students in enrolled in the course, and the number of students without a grade of the course with the maximum number of students enrolled in it (there might be more than one such courses). The result should contain four columns.

$S = \gamma_{cid,\ count(*) \to c,\ count(*) - count(grade) \to no}\ E$

$M = \gamma_{max(c) \to c}\ S$

$\Pi_{cid,\ cname,\ c,\ no}\ (C \bowtie S \bowtie M)$

with S as ( select cid, count(*) as c, count(*) - count(grade) as no from E group by cid)

M as (select max(c) as c from S)

select cid, cname, c, no from C NATURAL JOIN S NATURAL JOIN M;

(b) [3] The average grade of a student is the average of the grades she/he has received in the courses she/he has enrolled in. Write a query that computes the difference between the best average grade and the worst average grade in the database. The result should have one tuple and one value. For example, if the best average grade is 9.9 and the worst average grade is 5, the result would be 4.9.

$A = \gamma_{sid,\ avg(grade) \to a}\ E$

$Max = \gamma_{max(a) \to max}\ A$

$Min = \gamma_{min(a) \to min}\ A$

$\Pi_{max - min}\ (Max \times Min)$

With A as (select avg(grade) as a from E group by sid),

Min AS (select min (a) as min from A),

Max as (select max (a) as max from A)

select max - min from MAX, Min;

(c) [3] The antijoin between two tables R and S, written $R \triangleright S$ is defined as those tuples in R for which there is no tuple in S that is equal on the common attributes of R and S. For instance, if $R(a, b)$ and $S(b, c)$, for every tuple r in $R \triangleright S$ there is no tuple s in $S$ that satisfies r.b = s.b. Write a query that computes $Students \triangleright Enrolled$.

$\sigma_{b\ NOT\ IN\ (\Pi_b S)}\ R.$

select * from R

where b not in (select b from S)

**End of examination**
**Total pages: 7**
**Total marks: 27**