The University of British Columbia  
Computer Science 304

Midterm Examination  
March 17, 2010

Time: 50 minutes  
Instructor: Rachel Pottinger

Instructor: Rachel Pottinger

Name ANSWER KEY ____________________________Student No______________
(PRINT) (Last) (First)

Signature ______________________________________

This examination has 3 doublesided pages.

Check that you have a complete paper.

This is a closed book, closed notes exam. No books or other material may be used.

Answer all the questions on this paper.

Give very short but precise answers.

State any assumptions you make

Work fast and do the easy questions first. Leave some time to review your exam at the end.

Question  Mark  Out of

| 1.a   | 5      |
| 1.b   | 5      |
| 2.a   | 5      |
| 2.b   | 5      |
| 2.c   | 5      |
| 2.d   | 5      |
| TOTAL |         |

Out of 30
All queries for this exam use the same schema as in the SQL tutorials:

- authors (au_id, au_lname, au_fname, phone, address, city, state, zip)
- titleauthors (au_id, title_id, au_ord, royaltyshare)
- sales (sonum, stor_id, ponum, sdate)
- salesdetails (sonum, qty_ordered, qty_shipped, title_id, date_shipped)
- editors (ed_id, ed_lname, ed_fname, ed_pos, phone, address, city, state, zip, ed_boss)
- titleditors (ed_id, title_id, ed_ord)
- titles (title_id, title, type, pub_id, price, advance, ytd_sales, contract, notes, pubdate)
- publishers (pub_id, pub_name, address, city, state)

The schema will be repeated on following pages for easy reference.

Foreign Keys are shown in the following diagram, where the referring attribute is marked by a + and the referencing attribute is marked by a ↷ (e.g., au_id in titleauthors references au_id in authors).
The schema again:

- authors(au_id, au_lname, au_fname, phone, address, city, state, zip)
- titleauthors(au_id, title_id, au_ord, royaltyshare)
- sales(sonum, stor_id, ponum, sdate)
- salesdetails(sonum, qty_ordered, qty_shipped, title_id, date_shipped)
- editors(ed_id, ed_lname, ed_fname, ed_pos, phone, address, city, state, zip)
- titleditors(ed_id, title_id, ed_ord)
- titles(title_id, title, type, pub_id, price, advance, ytd_sales, contract, notes, pubdate)
- publishers(pub_id, pub_name, address, city, state)

1. {10 marks} Relational Algebra. For each query return EXACTLY the following:
   
a. Find the first name of all of the authors who have publishers in the city “Boston”
   
   \[
   \pi_{au\_fname}(\pi_{au\_id, au\_fname}(authors) \bowtie_{titleauthors} \bowtie_{titles} \bowtie_{\sigma_{city = 'Boston'}(publishers)})
   \]
   
   Common error: if you do a natural join and do not do a projection you will accidentally perform a join on the city, address, and state of authors and publishers

b. Return the last names of the authors and the editors of the book titled “You Can Combat Computer Stress!” Your answer should be a single list of the last names.

   \[
   \pi_{au\_lname}(authors \bowtie_{titleauthors} \bowtie_{\sigma_{title = 'You Can Combat Computer Stress!'(titles)}) \\
   \cup \pi_{ed\_lname}(editors \bowtie_{titleditors} \bowtie_{\sigma_{title = 'You Can Combat Computer Stress!'(titles)})
   \]
   
   Note: the attribute names do NOT need to be the same to be unioned (though doing so wouldn’t hurt), but they must be projected before the union can occur.
2. The schema again:
   authors( au_id, au_lname, au_fname, phone, address, city, state, zip)
   titleauthors( au_id, title_id, au_ord, royaltyshare)
   sales( sonum, stor_id, ponum, sdate)
   salesdetails( sonum, qty_ordered, qty_shipped, title_id, date_shipped)
   editors( ed_id, ed_lname, ed_fname, ed_pos, phone, address, city, state, zip)
   titleditors( ed_id, title_id, ed_ord)
   titles( title_id, title, type, pub_id, price, advance, ytd_sales, contract, notes, pubdate)
   publishers( pub_id, pub_name, address, city, state)

3. {20 marks} SQL Queries. For each query return EXACTLY the following:
   a. “List the last names of all authors who have a letter 'k' in their last name?” If a last name occurs more than once, only list it once

   **Answer:**
   ```sql
   SELECT DISTINCT au_lname
   FROM authors
   WHERE au_lname LIKE '%k%' or au_lname LIKE '%K%'
   Tuples:
   Karsen
   Locksley
   Yokomoto
   This is question 8j from the first SQL tutorial.
   Common error: You have to check for both the capital and lowercase K.
   Also need to have distinct
   
   b. List editor phone numbers and how many editors share that number, but don't list those lines where there is only one editor with that number.

   ```sql
   SELECT phone, COUNT(*)
   FROM editors
   GROUP BY phone
   HAVING COUNT(*) > 1;
   
   Tuples:
   301 468-3909       2
   
   This is question 9 from the second SQL tutorial. Note that count of pretty much anything else will work, too.
The schema again:

- authors( au_id, au_lname, au_fname, phone, address, city, state, zip)
- titleauthors( au_id, title_id, au_ord, royaltyshare)
- sales( sonum, stor_id, ponum, sdate)
- salesdetails( sonum, qty_ordered, qty_shipped, title_id, date_shipped)
- editors( ed_id, ed_lname, ed_fname, ed_pos, phone, address, city, state, zip)
- titleditors( ed_id, title_id, ed_ord)
- titles( title_id, title, type, pub_id, price, advance, ytd_sales, contract, notes, pubdate)
- publishers( pub_id, pub_name, address, city, state)

**c. How many editors have not edited a book?**

```
SELECT COUNT(*)
FROM editors e
WHERE e.ed_id not in
  (SELECT ed_id
   FROM titleditors)

Note: count of anything should work

COUNT(*)
---------
3
```

**d. Find the names of all publishers who have had more than 200 books ordered (note that a publisher may publish more than one book)**

```
CREATE VIEW manybooks as
SELECT p.pub_id
FROM publishers p, titles t, salesdetails s
WHERE p.pub_id = t.pub_id and s.title_id = t.title_id
GROUP BY p.pub_id
HAVING SUM(s.qty_ordered) > 200

SELECT p.pub_name
FROM manybooks m, publishers p
WHERE p.pub_id = m.pub_id
Answer:

-----------------------------------------------
New Age Books
Common mistakes: (1) as mentioned during the exam, you need to sum up the books and then (2) trying to select the pub_name in the same query as the group by, which is not legal.
```