CMPT 354
Assignment 2 Key

Total marks: 70 2000-1
Due: February 16, 2000 by 20:30 Instructor: G. Louie

1. Get a copy of the sample *pubs* database using the TA help sheet. The help sheet is found in \\LARCH\notes\cmpt354\SQL Server Help Sheet.pdf in the CSIL Lab or on the CMPT 354 website. Make the following changes to the database:

   a) (4 marks) In the *employee* table, add a new attribute: *supervisor*. An employee's supervisor is himself or herself also an employee, but you do not need to check on this as a constraint. The top-level employee's supervisor is the person himself/herself.

      Update the *employee* table with some supervisor names and hand in a copy of the new *employee* table definition and the list of the *employee* table's contents.

   b) (6 marks) Add a new table to the database, *customers*. Customers order books. A customer may make many orders. Customer information should include at least the customer name, phone number, street address, and city. You should define some constraints when creating the table, such as phone numbers must be in the form '(xxx) xxx-xxxx'. Add 2 - 3 additional attributes that you feel should be part of the customer entity.

      Hand in a printout of the table that you created, along with a *concise* explanation of your design and any assumptions that you made.

2. Suppose you are given the following schema:

   employee(emp_id, name, salary)
   flights(flight_no, from, to, distance, depart_time, arrival_time)
   aircraft(aircraft_id, manufacturer, model, range)
   certified(emp_id, aircraft_id)

   The *certified* relation indicates which employee(s) is/are certified to fly which aircraft. For each of the following expressions, give the equivalent SQL statement.

   For example, the following SQL statement would be used to find the names of employees who are certified to fly aircraft manufactured by 'Boeing':

   ```
   SELECT name
   FROM aircraft, certified, employee
   WHERE aircraft.aircraft_id = certified.aircraft_id
   AND aircraft.manufacturer = 'Boeing' AND employee.emp_id = certified.emp_id
   ```

a) (3 marks) Find the flight numbers of all the flights originating from Vancouver which depart after “13:00”.

```
select flight_no
from flights
where from = ‘Vancouver’ and depart_time > ’13:00’
```

Note that any reasonable assumption made about how the time is expressed is acceptable and full marks should be given.

b) (3 marks)

$\Pi_{emp\_id}(\sigma_{manufacturer\ =\ \text{'Boeing'}}(aircraft \bowtie certified))$

```
SELECT emp_id
FROM aircraft, certified
WHERE aircraft.aircraft_id = certified.aircraft_id
AND manufacturer = ‘Boeing’
```

c) (4 marks)

$\{t \mid \exists\ a \in aircraft\ \exists\ f \in flights\ (t[aircraft\_id] = a[aircraft\_id]$
$\quad \wedge\ f[from] = \text{"Vancouver"} \wedge f[to] = \text{"Tokyo"}) \wedge a[range] > f[distance]\}$

```
SELECT aircraft_id
FROM aircraft, flights
WHERE from = ‘Vancouver’ AND to = ‘Tokyo’
AND range > distance
```

d) (4 marks)

$\{t \mid \exists\ e \in employee\ \exists\ f \in flights\ \exists\ c \in certified\ \exists\ a \in aircraft$
$\quad (t[flight\_no] = f[flight\_no] \wedge a[range] > f[distance]$
$\quad \wedge\ e[salary] > 100{,}000 \wedge a[aircraft\_id] = c[aircraft\_id]$
$\quad \wedge\ e[emp\_id] = c[emp\_id])\}$

```
SELECT flight_no
FROM aircraft, certified, employee, flights
WHERE aircraft.aircraft_id = certified.aircraft_id
```

```
AND employee.emp_id = certified.emp_id
AND range > distance AND salary > 100,000
```

e) (5 marks)

$\{<n> \mid \exists\ e, a, r, a2, m\ (<e, a> \in certified \wedge <a, r> \in aircraft$
$\wedge <e, n> \in employee \wedge r > 3000 \wedge \neg (<e, a2> \in certified$
$\wedge <a2, m> \in aircraft \wedge m = \text{"Boeing"}))\}$

Note: make reasonable assumptions as what the domain variables refer to based on the schema.

```
SELECT E.name
FROM certified C, employee E, aircraft A
WHERE A.aircraft_id = C.aircraft_id
AND E.emp_id = C.emp_id
AND A.range > 3000
AND E.emp_id NOT IN ( SELECT C2.emp_id
         FROM Certified C2, Aircraft A2
         WHERE C2.aircraft_id = A2.aircraft_id
         AND A2.manufacturer = 'Boeing' )
```

f) (5 marks)

$\Pi_{emp\_id}(employee) - (\Pi_{e2.emp\_id}(employee \bowtie_{employee.salary > e2.salary} \rho_{e2}(employee)))$

```
SELECT emp_id
FROM employee
WHERE salary =
       ( Select MAX(E2.salary)
         FROM employee E2 )
```

3. Let $R = (A, B, C)$ and r_1 and r_2 both be relations on schema R. If there are 200 tuples in r_1 and 350 tuples in r_2, give the minimum and maximum sizes (in tuples) of the resulting relation produced by the following expressions.

a) (2 marks) $r_1 \cup r_2$

minimum is 350 tuples, maximum is 550 tuples (200 + 350)

b) (2 marks) $r_1 \cap r_2$

minimum is 0 tuples, maximum is 200 tuples

c) (2 marks) $r_1 - r_2$

minimum is 0 tuples, maximum is 200 tuples

d) (2 marks) $r_1 \times r_2$

exactly 70,000 (200 * 350) tuples

e) (2 marks) $\sigma_{A = \text{'Boeing'}}(r_1)$

minimum is 0 tuples, maximum is 200 tuples.

f) (2 marks) $\Pi_A (r_1)$

minimum is 1 tuple (remember projections eliminate duplicate tuples), maximum is 200 tuples.

g) (2 marks) $r_1 \div r_2$

minimum is 0 tuples, maximum is 0 tuples

h) (2 marks) $r_2 \div r_1$

minimum is 0 tuples, maximum is 0 tuples

i) (2 marks) $\Pi_{AB}(r_1) \bowtie \Pi_{BC}(r_2)$

minimum is 0 tuples (*i.e.* none of the B values in r_1 are equal to the B values in r_2), maximum is 200 tuples (*i.e.* all of the B values in r_1 have corresponding B values in r_2).

4. Suppose you are given the following schema:

   authors(<u>author_id</u>, last_name, first_name)
   books(<u>book_id</u>, title, num_pages, author_id)
   branches(<u>branch_id</u>, branch_name, address, phone_no)
   copies(<u>copy_id</u>, branch_id, book_id, cost)
   loans(<u>loan_id</u>, copy_id, borrower_id, due_date)
   borrowers(<u>borrower_id</u>, name, member_since)

   For each of the following SQL statements, describe, in plain English, what the statements is attempting to accomplish:

   a) (3 marks)
   ```
   select title
   from book B
   where  (select count(*)
           from copy C
           where C.book_id = B.book_id) < 10
   ```

   This query finds the titles of books that have fewer than 10 copies.

   b) (3 marks)
   ```
   select name
   from borrowers P
   where member_since >= 1998 and not exists
          (select *
           from books B
           where not exists
                 (select *
                  from copies C, loans L
                  where B.book_id = C.book_id
                  and C.copy_id = L.copy_id
                  and L.borrower_id = P.borrower_id))
   ```

   This query finds the names of borrowers who have been members since 1998 and have read all the books.

   c) (3 marks)
   ```
   select L.branch_id, L.name, average(pages) pagecount
   from branches L, books B
   where exists
          (select *
           from copies C
           where L.branch_id = C.branch_id
   ```

```
            and C.book_id = B.book_id)
    group by L.branch_id
    order by pagecount
```

Find the average number of pages in the books for each branch and order the result by increasing average page count.

Now give the equivalent SQL statements for the following queries:

d) (3 marks)
For each book-branch combination, find the number of copies of the book available at the branch. Give the branch name, book title, and count in the results.

```
select branch_name, title, count(*)
from branches L, copies C, books B
where L.branch_id = C.branch_id and C.book_id = B.book_id
group by L.branch_name, B.title
```

e) (3 marks)
Find the names of the branches that have no books available for loan.

```
select branch_name
from branches B
where not exists
        (select *
         from copies C
         where C.branch_id = B.branch_id)
```

f) (3 marks)
Find the names of branches that hold a copy of a book written by ‘Korth’.

```
select branch_name
from branches
where branch_id in
        (select branch_id
         from branches L, copies C, books B, authors A
         where L.branch_id = C.branch_id and C.book_id = B.book_id
         and B.author_id = A.author_id and A.last_name = ‘Korth’)
```