Submission Instructions:

Due Date: Tuesday, October 9, 2007 1:00 p.m.

- Answer both of the questions provided in the assignment.
- The assignment must be **typed.** Handwritten assignments will **not** be marked.
- If you are working in a team of two, submit only **one** assignment. (If two assignments are submitted, the assignment with the lower mark will be considered.)
- Include a cover page. A template for the cover page is available on the Assignment One web page.
- All pages must be **numbered** and **stapled** together. Do not put them in an envelope. Attach all pages well. Only those pages actually stapled to the front page will be marked – the markers cannot be responsible for pages that go astray. For this reason – don't use a paperclip, fold your pages together at the corner, or attach them by any other creative means.
- Late assignments must be emailed to your instructor (either jm@cs.toronto.edu, Professor Mylopoulos; or faye@cs.toronto.edu. Professor Baron). Acceptable electronic formats for submission are PDF and Postscript. Any other format will not be marked.
- Assignments cannot be more than three days late – assignments received after 1:00 p.m. Friday, October 12, 2007, will not be considered. A 10% penalty (five marks) will be applied for each late day.

Question 1 – The Relational Model and DDL [20 marks]

Le Challow Stores is building a database to manage purchasing, inventory control and sales. They must capture the following kinds of information:

- Le Challow Stores has company locations. Each location has a location ID, address, city, province, telephone number, and location type.
- Location types may be one of office, warehouse or store.
- There are employees. Each employees has a base location, last name, first name, job type (sales, purchasing, receiving, office), address, city, province, telephone number, social insurance number, pay type (salary which is annual rate, and hourly which is paid for each hour worked), and pay rate.
- There are clothing items that will be sold in the stores. Each item has a unique ID, a short description, long description, a clothing category (shirt, dress, sweater, scarf, etc.), intended sex of wearer (women, men), a vendor ID, cost price, sale price, and a discontinued flag to indicate the product may no longer be purchased.
- Clothing items also have sizes and colours, where applicable. They require a record for each size in which the clothing item is manufactured, and a record for each colour in which it is manufactured.
- Purchase orders are used to order new items from vendors. Each order indicates the location to which the items must be shipped, a unique purchase order number for reference, the vendor, the order date, the employee who placed the purchase. If a purchase order is cancelled, it is deleted from the system. All records dependent on the purchase order are also deleted.
- Each purchase order has order item details. For each item ordered we must specify the purchase order ID, the clothing item ID, size, colour, quantity, and status (on order, received). Note that there must

be a separate order item detail for each size and colour combination ordered. If an order item is deleted, all items dependent on this item must be deleted.

- There is a distribution record for each order item detail. The purchaser predetermines the stores to which stores the ordered items will be sent, the status of the distribution (sent, received, and none) and how many of each size and colour. Distribution details are determined by the buyer at the time of purchase. A single purchase order item received at the warehouse may be distributed to ten different locations.
- Inventory is managed for each location. Inventory information is kept for each different clothing item by item ID, size, and colour. The quantity of each item received, sold, and on-hand (in the store at the time a manual inventory is taken) is maintained.
- There are Vendors. For each Vendor, we maintain information about their name, address, city, province or state, country, primary contact name, primary contact telephone number, relationship rating (e.g., reliable, usually late, excellent, closed).
- There are Sales in all of the stores. For each sale we record the ID, location, date and time, form of payment (cash, debit, credit card), credit card type and number (if applicable), and approval or authority number received when debits and credit cards are approved for the purchase amount.
- For each sale tax information is recorded (in a separate table?) which contains the sale ID, tax type, (GST, PST), the amount charged, the province code (if it is a PST record), and when the tax has been remitted to the appropriate government.
- Finally, there are Sales Items. These are the details about individual clothing items that are sold to the customer. They include the Clothing item ID, size, colour, quantity, sale price, and Sale ID.

Define a relational schema that can accommodate the above information in DDL. Remember: Enforce all key and integrity constraints.

- Provide default values where appropriate.
- Avoid duplicating information (except for foreign keys used to link relations.)
- Indent your DDL statements consistently for readability.
- You may make text fields any (reasonable) size you want.
- If you are using a descriptive field that occurs in more than one tuple, you may want to encode it in a separate table.

Possible Solution:

```
CREATE TABLE Locations        ( locationID     INTEGER PRIMARY KEY,
                                address        CHAR(50) NOT NULL,
                                city           CHAR(30) NOT NULL,
                                province       CHAR(20) NOT NULL,
                                telephone      CHAR (20)NOT NULL,
                                locationType   CHAR(01) NOT NULL REFERENCES
                                                    LocationTypes (code) ON
                                                    UPDATE CASCADE ON DELETE
                                                    RESTRICT
                              )

CREATE TABLE LocationTypes    ( code           CHAR (01) PRIMARY KEY,
                                description    CHAR (10) NOT NULL
                              )
```

```
CREATE TABLE Employees        ( ID             INTEGER PRIMARY KEY,
                                location       INTEGER NOT NULL REFERENCES
                                                    Locations (locationID),
                                lName          CHAR (20) NOT NULL,
                                fName          CHAR (20) NOT NULL,
                                jobType        CHAR (1) NOT NULL REFERENCES
                                               JobTypes ( code) ON DELETE
                                               RESTRICT, ON UPDATE CASCADE,
                                address        CHAR(50) NOT NULL,
                                city           CHAR(30) NOT NULL,
                                province       CHAR(20) NOT NULL,
                                telephone      CHAR (20)NOT NULL,
                                SIN            CHAR (9) NOT NULL,
                                payType        CHAR (1) NOT NULL REFERENCES
                                               PayTypes ( code) ON DELETE
                                               RESTRICT, ON UPDATE CASCADE,
                                payRate        NUMERIC (9,2) NOT NULL
                              )

CREATE TABLE JobTypes         ( code           CHAR (1) PRIMARY KEY,
                                desc           CHAR (30) NOT NULL)
                              )

CREATE TABLE PayTypes         ( code           CHAR (1) PRIMARY KEY,
                                desc           CHAR (30) NOT NULL)
                              )

CREATE TABLE ClothingItems    ( ID             INTEGER PRIMARY KEY,
                                shortDesc      CHAR (20) NOT NULL,
                                longDesc       VARCHAR (100) NOT NULL,
                                cat            INTEGER NOT NULL REFREENCES
                                               ClothingCats (ID),
                                sex            CHAR (01) NOT NULL CHECK ((sex
                                               = ‘M’) OR (sex = ‘F’)),
                                vendorID       INTEGER NOT NULL REFERENCES
                                               Vendors(ID),
                                costPrice      NUMERIC (7,2) NOT NULL,
                                salePrice      NUMERIC (7,2) NOT NULL,
                                discontinued   BOOL DEFAULT FALSE NOT NULL
                              )

CREATE TABLE ClothingCats     ( ID             INTEGER PRIMARY KEY,
                                desc           CHAR (20) NOT NULL
                              )

CREATE TABLE ClothingColours  ( ID             INTEGER PRIMARY KEY,
                                clothingID     INTEGER NOT NULL REFERENCES
                                               ClothingItems (ID),
                                colour         CHAR (20) NOT NULL
                              )

CREATE TABLE ClothingSizes    ( ID             INTEGER PRIMARY KEY,
                                clothingID     INTEGER NOT NULL REFERENCES
```

```
                                                   ClothingItems (ID),
                                  size             INTEGER NOT NULL
                                )

CREATE TABLE PurchaseOrders     ( poID             INTEGER PRIMARY KEY,
                                  shipToLoc        INTEGER NOT NULL REFERENCES
                                                   Locations (locationID),
                                  orderDate        TIMESTAMP,
                                  employeeID       INTEGER NOT NULL REFERENCES
                                                   Employees (ID) ON DELETE
                                                   RESTRICT ON UPDATE CASCADE
                                )

CREATE TABLE
     PurchaseOrderItems         ( poItemID         INTEGER PRIMARY KEY,
                                  poID             INTEGER NOT NULL REFERENCES
                                                   PurchaseOrders (poID) ON
                                                   UPDATE CASCADE ON DELETE
                                                   CASCADE,
                                  clothingID       INTEGER NOT NULL REFERENCES
                                                   ClothingItems (ID),
                                  colourID         INTEGER NOT NULL REFERENCES
                                                   ClothingColours (ID),
                                  sizeID           INTEGER NOT NULL REFERENCES
                                                   ClothingSizes (ID),
                                  quantity         INTEGER NOT NULL,
                                  status           INTEGER NOT NULL REFERENCES
                                                   OrderStatusCodes (ID)
                                )

CREATE TABLE
     OrderStatusCodes           ( ID               INTEGER PRIMARY KEY,
                                  desc             CHAR (20) NOT NULL
                                )

CREATE TABLE
     ItemDistributions          ( ID               INTEGER PRIMARY KEY,
                                  locID            INTEGER NOT NULL REFERENCES
                                                   Locations (locationID),
                                  POiTEMid         INTEGER NOT NULL REFERENCES
                                                   PurchaseOrderItems (poItemID)
                                                   ON UPDATE CASCADE ON DELETE
                                                   CASCADE,
                                  status           INTEGER NOT NULL REFERENCES
                                                   DistributionStatusCodes (ID)
                                )

CREATE TABLE
     DistributionStatusCodes    ( ID               INTEGER PRIMARY KEY,
                                  desc             CHAR (20) NOT NULL
                                )

CREATE TABLE InventoryItems     ( clothingID       INTEGER NOT NULL REFERENCES
                                                   ClothingItems (ID),
                                  colourID       INTEGER NOT NULL REFERENCES
                                                   ClothingColours (ID),
                                  sizeID         INTEGER NOT NULL REFERENCES
                                                   ClothingSizes (ID),
                                  qtyOnHand      INTEGER NOT NULL,
                                  qtySold        INTEGER NOT NULL,
                                  qtyReceived    INTEGER NOT NULL,
                                  PRIMARY KEY    (clothingID, colourID, sizeID)
                                )

CREATE TABLE Vendors         ( ID             INTEGER PRIMARY KEY,
                                  name           CHAR (50) NOT NULL,
                                  address        CHAR(50) NOT NULL,
                                  city           CHAR(30) NOT NULL,
                                  provinceState  CHAR(20) NOT NULL,
                                  country        CHAR(20) NOT NULL,
                                  contactName    CHAR(30) NOT NULL,
                                  telephone      CHAR (20)NOT NULL,
                                  rating         INTEGER NOT NULL REFRENCES
                                                   VendorRelationships (ID)
                                )

CREATE TABLE
     VendorRelationships     ( ID             INTEGER PRIMARY KEY,
                                  desc           CHAR (40) NOT NULL
                                )

CREATE TABLE Sales           ( ID             INTEGER PRIMARY KEY,
                                  saleDate       TIMESTAMP,
                                  fop            INTEGER NOT NULL REFRENCES
                                                   PaymentTypes (ID),
                                  fopNumber      CHAR (20) NOT NULL,
                                  fopAuthCode    CHAR (20) NOT NULL,
                                )

CREATE TABLE PaymentTypes
                                ( ID             INTEGER PRIMARY KEY,
                                  desc           CHAR (40) NOT NULL
                                )

CREATE TABLE TaxItems        ( saleID         INTEGER NOT NULL REFERENCES
                                                   Sales (ID),
                                  taxType        CHAR (01) NOT NULL CHECK
                                                   ((taxType = ‘P’) OR (taxType =
                                                   ‘F’)),
                                  province       CHAR(2) NOT NULL REFERENCES
                                                   ProvinceCodes(code),
                                  remitDate      TIMESTAMP,
                                  amount         NUMERIC (7,2) NOT NULL,
                                  PRIMARY KEY    (saleID, taxType)
                                )
```

```
CREATE TABLE ProvinceCodes     ( code          CHAR (02) NOT NULL,
                                 desc          CHAR (20) NOT NULL
                               )

CREATE TABLE SalesItems        ( saleID        INTEGER NOT NULL REFERENCES
                                               Sales (ID),
                                 clothingID    INTEGER NOT NULL REFERENCES
                                               ClothingItems (ID),
                                 colourID      INTEGER NOT NULL REFERENCES
                                               ClothingColours (ID),
                                 sizeID        INTEGER NOT NULL REFERENCES
                                               ClothingSizes (ID),
                                 QTY           INTEGER NOT NULL,
                                 PRICE         NUMERIC (6,2) NOT NULL,
                                 PRIMARY KEY   (saleID, clothingID, colourID,
                                               sizeID)
                               )
```

Question 2 – Relational Algebra [30 marks]

For this problem we use the schema of the Ontario Hospital Health Network database. It includes the following relations. Keys are underlined and each tuple is described:

- Hospital(hospitalID, hName, hAddress, hCity) A hospital including the hospital ID, name, address and city of that hospital.
- Room(roomID, hospitalID, rNumber, rSize) A hospital room including the hospital ID, room number, and room capacity – the number of patients it accommodates.
- Employee(employeeID, eFirstName, eLastName, eRole, eSalary, eAddress, eCity, eSIN) An employee including their employee ID number, first and last names, role in the hospital (or occupation type, such as doctor, technician, etc.), salary, address, city and Social Insurance Number.
- Department(departmentID, hospitalID, dName, dHead) a department in a hospital including the hospital ID, the department ID, the name of the department, and the employee ID of the head of the department.
- EmployeeDepartment(employeeID, departmentID, edStartDate, edEndDate) the departments to which an employee belongs. An employee may work in more than one department. This tuple contains the employee ID, the department ID, and the dates that they started and finished employment in that department.
- Patient(patientID, pFirstName, pLastName, pSex, pOHIP, pOther, pDOB, pAddress, pCity, pProvince, pCountry) A patient in the hospital system including the patient ID, first name, last name, sex (M or F), OHIP number, other form of payment if not on OHIP ,date of birth, residence address including the address, city, province, and country.
- PatientAppointment(patientID, departmentID, employeeID, paDate, paTime, paReason) An appointment for a patient, including the department ID of the hospital department, the employee ID of the person whom they are seeing (could be technician), the date, the time, and the purpose of the visit.
- PatientRoomAssignment(patientID, roomID, prInDate, prOutDate, prDoctor) A room that is assigned to a patient who needs to stay in the hospital - including the ID of the patient and room, the date that they were admitted, the date that they left the hospital, and the employee ID of their attending physician during their stay.

The following inclusion dependencies (foreign keys) hold: Room(hospitalID) $\subseteq$ Hospital(hospitalID)

Department(hospitalID) $\subseteq$ Hospital(hospitalID)

Department(dHead) $\subseteq$ Employee(employeeID)

EmployeeDepartment(departmentID) $\subseteq$ Department(departmentID) PatientAppointment(patientID) $\subseteq$ Patient(patientID)

PatientAppointment(departmentID) $\subseteq$ Department(departmentID) PatientAppointment(employeeID) $\subseteq$ Employee(employeeID) PatientRoomAssignment (patientID) $\subseteq$ Patient(patientID) PatientRoomAssignment (roomID) $\subseteq$ Room(roomID)

Write the following queries in Relational Algebra. You are not allowed to use the renaming or relations (as described in the text), and the extended relational algebra. You may use renaming of attributes, as well as numerical and date comparisons (e.g., R.A > 5, or R.A ≥ YYYY-MM-DD). You may specify a date as YYYY-MM-DD and time as HH:MM (using a 24 hour clock). Also assume that you can subtract one date from another to get the number of days. So if a first date minus a second date equals 5, then the first date if five days after the second date. Use CurrentDate to represent today's date and be sure to clearly state any assumptions that you make.

1. Find the first name, last name, hospital, department name, date, and time of all patients who had an appointment with Dr. Joe Smith. (assume there is only one Dr. Joe Smith)
2. Find the hospital name and department name for all departments whose head is also head of another department.
3. Find the first name and last name of all Doctors whose patients are never in the hospital for less than 7 days. (They must have had patients admitted in the hospital at some time.)
4. Find the first and last name of all patients who have exactly two doctors in the hospital system. (Note that doctors are specified in both appointments and room assignment.)
5. Find the hospital and department name for the department at which there were no appointments from June 3, 2007 to June 8, 2007.
6. Find the hospital, room number, and patient first and last names of all semi-private (size = 2) rooms which were occupied by both a male and female patient both of whom were over 50 on August 13, 2007.

A possible solution

For simplicity by

- H we refer to relation Hospital;
- R we refer to relation Room;
- E we refer to relation Employee;
- D we refer to relation Department;
- ED we refer to relation EmployeeDepartment;
- P we refer to relation Patient;
- PA we refer to relation PatientAppointment;
- PRA we refer to relation PatientRoomAssignment;

1. $\pi_{pFirstName,\ pLastName,\ hName,\ dName,\ paDate,\ paTime}(\sigma_{eFirstName = 'Joe' \land eLastName='Smith'}(P \bowtie PA \bowtie E \bowtie D \bowtie H))$

2. We join relation D with itself on attribute dHead and select tuples with different departmentIDs.

$\pi_{hName,dName}((\sigma_{departmentID \neq dID}(D \bowtie \rho_{dID \leftarrow departmentID,\ hID \leftarrow hospitalID,\ dN \leftarrow dName}(D))) \bowtie H)$

3. First we find doctors who had some patients in hospital for less than 7 days:

$\pi_{prDoctor}(\sigma_{prOutDate - prInDate < 7}(PRA))$

Then we subtract the above doctors from the set of all doctors to get the ones who never had a patient in hospital for less than 7 days:

$\pi_{prDoctor}(PRA) - \pi_{prDoctor}(\sigma_{prOutDate - prInDate < 7}(PRA))$

Now we join the above relation with E to get the names:

$\pi_{eFirstName,eLastName}(\rho_{prDoctor \leftarrow employeeID}(E) \bowtie (\pi_{prDoctor}(PRA) - \pi_{prDoctor}(\sigma_{prOutDate - prInDate < 7}(PRA))))$

4. First we create a temporary relation containing all the patients and their doctors. This information can be found in both PRA and PA relations:

$R1 = \pi_{patientID,prDoctor}(PRA) \cup \rho_{prDoctor \leftarrow employeeID}(\pi_{patientID,employeeID}(\sigma_{dRole='Doctor'}(E \bowtie PA)))$

We next find patients who have at least two doctors:

$R2 = \pi_{patientID}(\sigma_{prDoctor \neq pD}(R1 \bowtie \rho_{pD \leftarrow prDoctor}(R1)))$

Now we find patients with at least three doctors:

$R3 = \pi_{patientID}(\sigma_{prDoctor \neq pD1 \wedge prDoctor \neq pD2 \wedge pD2 \neq pD1}(R1 \bowtie \rho_{pD1 \leftarrow prDoctor}(R1) \bowtie \rho_{pD2 \leftarrow prDoctor}(R1)))$

Patients with *exactly* two doctors are those who are in R2 but not in R3:

$\pi_{pFirstName,pLastName}(P \bowtie (R2 - R3))$

5. We first find departments which had some appointments between June 3, 2007 and June 8, 2007:

$R1 = \pi_{departmentID}(\sigma_{paDate > '2007\text{-}06\text{-}02' \wedge paDate < '2007\text{-}06\text{-}09'}(PA))$

Departments with no appointment in that period can be found by subtracting R1 from the set of all departments:

$\pi_{hName,dName} (H \bowtie D \bowtie (\pi_{departmentID} (D) - R1))$

6. We first select rooms of size 2:

$R1 = \pi_{roomID} (\sigma_{rSize = 2} (R))$

Then we find rooms (and patients) in R1 in which there was a female patient on August 13, 2007 with over 50 years of age:

$R2 = \pi_{roomID,patientID} (\sigma_{prInDate < 2007\text{-}08\text{-}13 \wedge prOutDate > 2007\text{-}08\text{-}13 \wedge pSex='F' \wedge pDOB < 1957\text{-}08\text{-}13} (P \bowtie PRA \bowtie R1))$

Next we find rooms (and patients) in R1 in which there was a male patient on August 13, 2007 with over 50 years of age:

$R3 = \pi_{roomID,patientID} (\sigma_{prInDate < 2007\text{-}08\text{-}13 \wedge prOutDate > 2007\text{-}08\text{-}13 \wedge pSex='M' \wedge pDOB < 1957\text{-}08\text{-}13} (P \bowtie PRA \bowtie R1))$

Rooms that were occupied by both a male and a female over 50 on August 13, 2007 are the ones that are both in R2 and R3.

$R4 = \pi_{roomID} (R2) \cap \pi_{roomID} (R3)$

Now we find names of the patients in R2 or R3 that occupied a room in R4:

$\pi_{hName,rNumber,pFirstName,pLastName} (H \bowtie R \bowtie P \bowtie ((R2 \cup R3) \bowtie R4))$