CSCI 332 Database Concepts
G. Pothering
Fall 2020
Assignment 3 - Tuple Calculus -Solutions

Complete the following exercises from Chapter 5 of your textbook, page 140.

1. Exercise 5.12 parts (b), (c), (e) and (g). This problem uses the Hotel database, which has the structure given on page 118 of the textbook and is the same one you used in Assignment 1. We reproduce the schema here:

```
Hotel(hotelNo,hotelName,city)
Room(roomNo,hotelNo,type,price)
Booking(hotelNo,guestNo,dateFrom,dateTo,roomNo)
Guest(guestNo,guestName,guestAddress)
```

   b. List all single rooms with a price below £20 per night.

$$\{r \mid r \in Room \wedge r[price] < 20\}$$

   c. Lists the names and cities of all guests.

   Note: Here by "city" we shall mean city where a guest is staying, rather than the city the guest is from since we have no way to extract just the city subtext from the address attribute of Guest.

$$\{g[guestName], h[city] \mid g \in Guest \wedge h \in Hotel \wedge \exists b(b \in Booking \wedge g[guestNo] = b[guestNo] \wedge b[hotelNo] = h[hotelNo])\}$$

   e. List all guests currently staying at the Grosvenor Hotel.

$$\{g \mid g \in Guest \wedge \exists h(h \in Hotel \wedge h[hotelName] = \text{'Grosvenor Hotel'} \wedge \exists b(b \in Booking \wedge g[guestNo] = b[guestNo] \wedge b[hotelNo] = h[hotelNo] \wedge b[dateFrom] \leq currDate \wedge b[dateTo] \geq currDate))\}$$

   [where you see ***currDate*** above, a date of the form DD-MMM-YYYY may be used]

   g. List the guest details and dates they stayed (guestNo,guestName,and guestAddress) of all guests who have ever stayed at the Grosvenor Hotel.

$$\{g,b[dateFrom],b[dateTo] \mid g \in Guest \wedge b \in Booking \wedge g[guestNo] = b[guestNo] \wedge \exists h(h \in Hotel \wedge h[hotelName] = \text{'Grosvenor Hotel'} \wedge b[hotelNo] = h[hotelNo])\}$$

2. Still using the Hotel database from problem 1, express the following queries in tuple calculus:

   a. Get a listing of the names of hotels that have a hotel in Cambridge.

$$\{h \mid h \in Hotel \wedge h[city] = \text{'Cambridge'}$$

   b. Get a listing of the names of hotels that do not have a hotel in Cambridge.

$$\{h \mid h \in Hotel \wedge \neg(\exists h1(h1 \in Hotel \wedge h1[city] = \text{'Cambridge'} \wedge h[hotelNo] = h1[hotelNo]))$$

c. Get a listing of the guest numbers of guests who have had a booking with every hotel in Stratford.

{g[guestNo] | g ∈ Guest ∧ ∀h((h ∈ Hotel ∧ h[city] = 'Stratford') →
∃b (b ∈ Booking ∧ h[hotelNo] = b[hotelNo] ∧
g[guestNo] = b[guestNo]))}

3. Exercise 5.10 parts (a), (b) and (c). Use the Hotel database as described on page 118. Again, this is the same one you used in Assignments 1 and 2. Here by "describe the relation" you are to describe he query using language like that in problems 1. and 2.above. Terms like "projection," "selection," "join," etc. may not appear in your answer.

a. {h[hotelName] | h ∈ Hotel ∧ h[city] = 'London')}

Get the names of all hotels in London.

b. {h[hotelName] | h ∈ Hotel ∧ ∃r(r ∈ Room ∧ h[hotelNo]= r[hotelNo] ∧
r[price] > 50}

Give the names of all hotels that have rooms whose price is more than £50 (the £ symbol is not required)

c. {h[hotelName] | h ∈ Hotel ∧ ∃b (b ∈ Booking ∧ ∃g(g ∈ Guest ∧
h[hotelNo]= b[hotelNo] ∧b[guestNo] = g[guestNo] ∧
g[guestName] = 'John Smith'))}

Give the names of any hotels who had a guest named John Smith

4. Problem 5.9 on page 140, but only give the tuple calculus equivalent of the relational algebra expressions in parts (a),(b),(c) and (f) from Problem 5.8.

a. $\pi_{hotelNo}(\sigma_{price>50}(Room))$

{r[hotelNo] | r ∈ Room ∧ r[price] > 50}

b. $\sigma_{Hotel,hotelNo=Room.hotelNo}(Hotel \times Room)$

{h,r | h∈ Hotel ∧ r ∈ Room ∧ h[hotelNo] = r[hotelNo]}

c. $\pi_{hotelName}(Hotel \mid\times\mid_{Hotel.hotelNo=Room.roomNo}(\sigma_{cprice>50}(Room))$

{h[hotelName] | h ∈ Hotel ∧ ∃r(r ∈ Room ∧ r[price] > 50 ∧
h[hotelNo] = r[hotelNo])}

f. $\pi_{guestName,hotelName}(Booking \mid\times\mid_{Booking.guestNo=Guest.guestNo} Guest) \div \pi_{hotelNo}(\sigma_{city='London'}(Hotel)$

```
{g[guestNo] | g ϵ Guest ∧ ∀h((h ϵ Hotel ∧ h[city] = 'London') →
                  ∃b(b ϵ Booking ∧ h[hotelNo] = b[hotelNo] ∧
                  g[guestNo] = b[guestNo]))}
```