University of Toronto
CSC343, Summer 2019

Assignment 1:
Yuchen Fan 1003800265
Junming Zhang 1003988982

Unary operators on relations:

- $\Pi_{x,y,z}(R)$
- $\sigma_{condition}(R)$
- $\rho_{New}(R)$
- $\rho_{New(a,b,c)}(R)$

Binary operators on relations:

- $R \times S$
- $R \bowtie S$
- $R \bowtie_{condition} S$
- $R \cup S$
- $R \cap S$
- $R - S$

Logical operators:

- $\vee$
- $\wedge$
- $\neg$

Assignment:

- $New(a, b, c) := R$

Stacked subscripts:

- $\sigma_{this.something > that.something \wedge \\ this.otherthing \leq that.otherthing}$

Below is the text of the assignment questions; we suggest you include it in your solution. We have also included a nonsense example of how a query might look in LaTeX. We used `\var` in a couple of places to show what that looks like. If you leave it out, most of the time the algebra looks okay, but certain words, *e.g.*, "Offer" look horrific without it.

The characters "\\" create a line break and "[5pt]" puts in five points of extra vertical space. The algebra is easier to read with extra vertical space. We chose "–" to indicate comments, and added less vertical space between comments and the algebra they pertain to than between steps in the algebra. This helps the comments visually stick to the algebra.

Part 1: Queries

1. Report the name of the Patron that has given the highest rating to a restaurant. If there are ties, report all of them.

   – Patrons who are rating.
   $RatingPatrons := Patron \bowtie Rating$

   – Patrons who did not give top rating.
   $NotTopRating(PID, fname, lname, rname) :=$
   $\Pi_{R1.PID,R1.fname,R1.lname,R1.rname}\sigma_{R1.rating<R2.rating}[(\rho_{R1}RatingPatron)\times(\rho_{R2}RatingPatron)]$

   – Name of Patron with highest rating.
   $HighestRating(fname, lname) :=$
   $\Pi_{fname,lname}[\Pi_{PID,fname,lname,rname}RatingPatrons - NotTopRating]$

2. Report the name of the restaurant for which the highest number of reservations were made. If there are ties report all of them.

   Cannot be expressed.
   No attribute says anything about number of reservations (need aggregation).

3. Report the PID(s) of the Patrons(s) who reserved a spot at a restaurant, but did not order anything.

   – Reservation with ordering something
   $ReservationWithOrder(RID, PID) := \Pi_{RID,PID}(Reservation \bowtie Order)$

   – PID of Patron who reserved a spot without ordering
   $ReservationWithoutOrder(PID) := \Pi_{PID}[\Pi_{RID,PID}Reservation - ReservationWithOrder]$

4. Report the name(s) of the Patrons(s) who have made a reservation to the restaurant named 'Boston Pizza' and ordered 3 of a dish called 'Margherita Pizza'.

   – RID and Patrons' name of Reservation at Boston Pizza
   $ReservationAtBP(RID, fname, lname) :=$
   $\Pi_{RID,fname,lname}\sigma_{rname=\text{“}BostonPizza\text{”}}(Reservation \bowtie Patron)$

   – Orders in Margherita Pizza
   $OrderInMargheritaPizza(RID, number) :=$
   $\Pi_{RID,number}\sigma_{dish.name=\text{`}MargheritaPizza\text{'}}(Order \bowtie Dish)$

– name of Patrons who order 3 dish of Margherita Pizza at Boston Pizza

$OrderWith3MP(fname, lname) :=$

$\Pi_{fname,lname}\sigma_{number=3}(ReservationAtBP \bowtie OrderInMargheritaPizza)$

5. Report the owner of the restaurant with the highest average rating. If there are ties, report all of them.

Cannot be expressed

Aggregation is required for obtaining the average.

6. Report the capacities of the restaurants from which patrons have so far only ordered foods with a 'gluten-free' dietary restriction.

– RID and DID of order with 'gluten-free'

$OrderWithGF(RID, DID) := \Pi_{RID,DID}\sigma_{dietary='gluten-free'}(Dish \bowtie Order)$

– RID and DID of order without 'gluten-free'

$OrderWithoutGF(RID, DID) := \Pi_{RID,DID}\sigma_{dietary\neq'gluten-free'}(Dish \bowtie Order)$

– Name of restaurant with gluten-free order

$RestaurantWithOrderGF(rname) :=$

$\Pi_{rname}(Reservation \bowtie OrderWithGF)$

– Name of restaurant without gluten-free order

$RestaurantWithoutOrderGF(rname) :=$

$\Pi_{rname}(Reservation \bowtie OrderWithoutGF)$

– Name of restaurant with only gluten-free order

$RestaurantOnlyOrderGF(name) :=$

$RestaurantWithOrderGF - RestaurantWithoutOrderGF$

– Capacity of restaurant with only gluten-free order

$OnlyOrderGF(capacity) := \Pi_{capacity}(Restaurant \bowtie RestaurantOnlyOrderGF)$

7. Report the restaurant owner for which the very earliest reservation out of all the reservations in the database was made. Report any ties.

– RID and name of reservation which is not earliest

$NotEarliest(RID, rname) :=$

$\Pi_{R1.RID,R1.name}\sigma_{R1.date>R2.date}[(\rho_{R1}Reservation) \times (\rho_{R2}Reservation)]$

– RID and Restaurants' Name of earliest reservation

$Earliest(name) :=$

$\Pi_{rname}(\Pi_{RID,rname}Reservation - NotEarliest)$

– Owner of restaurant with earliest reservation

$EarliestRestaurant(Owner) :=$

$\Pi_{Owner}(Earliest \bowtie Restaurant)$

8. Report the PID(s) of the Patrons who have made reservations to the restaurant named 'Red Lobster' on their birthday.

– Rename Patron

$\rho_P Patron$

– Rename Reservation

$\rho_R Reservation$

$ReservationAtRedLobsterOnBirthday(PID) :=$

$\Pi_{P.PID}\sigma_{R.rname='RedLobster'}(P \bowtie_{P.PID=R.PID \wedge P.birthday=R.date} R)$

9. Consider all patrons that have made reservations to at least two different restaurants. For each of those patrons, report their name, and the names and ratings of all of the restaurants they went to (not ones they rated without actually going to).

– PID of Patrons went to at least two restaurant and name of restaurants.

$AtLeastTwoRestaurant(PID, rname1, rname2) :=$

$\Pi_{R1.PID,R1.rname,R2.rname}\sigma_{R1.PID=R2.PID \wedge R1.rname \neq R2.rname}[(\rho_{R1}Reservation) \times (\rho_{R2}Reservation)]$

– Name of Patrons with name of restaurant

$PatronsAtLeastTwo(PID, fname, lname, rname1, rname2) :=$

$\Pi_{PID,fname,lname,rname1,rname2}Patron \bowtie AtLeastTwoRestaurant$

– spliting name of restaurants to patron.

$PatronAndRestaurant1(PID, fname, lname, rname) :=$

$\Pi_{PID,fname,lname,rname1}PatronsAtLeastTwo$

$PatronAndRestaurant2(PID, fname, lname, rname) :=$

$\Pi_{PID,fname,lname,rname2}PatronsAtLeastTwo$

– name and rating of restaurants to patron

$PatronRestaurantRating1(PID, fname, lname, rname1, rating1) :=$

$\Pi_{PID,fname,lname,rname,rating}[PatronAndRestaurant1 \bowtie Rating]$

$PatronRestaurantRating2(PID, fname, lname, rname2, rating2) :=$
$\Pi_{PID,fname,lname,rname,rating}[PatronAndRestaurant2 \bowtie Rating]$

– Union two Patrons with name of restaurant
$WentAtLeastTwo(fname, lname, rname1, rating1, rname2, ranting2) :=$
$\Pi_{P1.fname,P1.lname,P1.rname1,P1.rating1,P2.rname2,P2.rating2}$
$[(\rho_{P1} PatronRestaurantRating1) \bowtie (\rho_{P2} PatronRestaurantRating2)]$

10. Report the name of all Restaurants that had reservations made on every day that someone made a reservation at the restaurant named 'Pickle Barrel'.

– Date of reservation in "Pickle Barrel"
$PB\_Date(date) := \Pi_{date}\sigma_{rname='PickleBarrel'} Reservation$

– Name and Date of restaurant in the reservation
$Restaurant\_Name(name, date) := \Pi_{rname,date} Reservation$

– All Name of restaurant should have reservation in the day with 'Pickle Barrel'.
$Name\_ShouldBeen(name, date) := (\Pi_{name} Restaurant\_Name) \times PB\_Date$

– All Name of restaurant which do not always have reservation in the same day with 'Pickle Barrel'.
$NotAlwaysSame(name, date) := Name\_ShouldBeen - Restaurant\_Name$

– All Name of restaurant which have reservation in every single day when 'Pickle Barrel' has the reservation.
$HasEveryDay(name) := \Pi_{name} Name\_ShouldBeen - \Pi_{name} NotAlwaysSame$

Part 2: Integrity Constraints

Express the following integrity constraints with the notation $R = \emptyset$, where R is an expression of relational algebra. You are welcome to define intermediate results with assignment and then use them in an integrity constraint.

1. A restaurant owner can only own one restaurant.

– Two Owners of two different restaurant name
$Owners(owner1, owner2) :=$
$\Pi_{R1.owner,R2.owner}[(\rho_{R1} Restaurant) \bowtie_{R1.name \neq R2.name} (\rho_{R1} Restaurant)]$

– one owner and one restaurant

$\sigma_{owner1=owner2} Owners = \emptyset$

2. Patrons who did not make a reservation for a restaurant cannot review it.

   – PID and Name of Patron made reservation

   $MadeReservation(PID, rname) := \Pi_{PID,rname} Reservation$

   – PID and Name of Patron reviewed

   $Reviewed(PID, name) := \Pi_{PID,rname} Rating$

   – no reservation no review

   $Reviewed - MadeReservation = \emptyset$

3. A Patron cannot make multiple reservations in one day for a restaurant that has a capacity less than 100.

   – Name of Restaurants with capacity < 100

   $LessCapacity(rname) := \Pi_{name} \sigma_{capacity<100} Restaurant$

   – Reservation with less capacity restaurant

   $ReservationWithLessCapacity = Reservation \bowtie LessCapacity$

   – Rename Reservation with less capacity restaurant

   $\rho_{R1} ReservationWithLessCapacity$

   $\rho_{R2} ReservationWithLessCapacity$

   – no multiple reservation

   $\sigma_{R1.RID \neq R2.RID \wedge R1.PID=R2.PID \wedge R1.rname=R2.rname \wedge R1.date=R2.date}(R1 \times R2) = \emptyset$