Samuel Steinberg

CS 465

January 31st, 2020

Homework #3

1. a) This will produce a relation of hotel ID numbers (hotelNo) where the price for a room from the Room table is greater than 50 pounds. Essentially this produce a relation of hotelNo's with a room price greater than 50 pounds.
   c) This will produce a natural join of Hotel and Room with a price of a room being greater than 50 pounds. This will produce a relation containing all hotel names with a room price greater than 50 pounds.
   f) This will produce a join of Booking and Guest and will essentially produce a relation containing the names of guests and hotel ID's for all tuples of guests with hotels booked in London.

2. b. Relational Algebra: $\Pi_{\text{roomNo}}(\sigma_{\text{price}} < 20 \ \wedge \ \sigma_{\text{type}} = 'single')$
   Tuple Relational Calculus: $\{R \mid \text{Room}(R) \ \wedge \ R.\text{price} < 20 \ \wedge R.\text{type} = 'single'\}$

   c. Relational Algebra: $\Pi_{\text{guestName,guestAddress}}(\text{Guest})$
   Tuple Relational Calculus: $\{G.\text{guestName}, G.\text{guestAddress} \mid \text{Guest}(G)\}$

   d. Relational Algebra:
   $\Pi_{\text{price,type}}(\text{Room} \ \bowtie_{\text{Room.hotelNo=Hotel.hotelNo}} (\sigma_{\text{hotelName='GrosvenorHotel'}}(\text{Hotel})))$

   Tuple Relational Calculus:
   $$\{R.\text{price}, R.\text{type} \mid \text{Room}(R) \wedge (\exists H)(\text{Hotel } H) \wedge (R.\text{hotelNo} = H.\text{hotelNo}) \wedge (H.\text{hotelName} = 'GrosvenorHotel')\}$$
   e. Relational Algebra:
   $$\text{Guest} \ \bowtie_{\text{Guest.guestNo=Booking.guestNo}} (\sigma_{\text{dateFrom} \le \text{CURDATE()}} \wedge \sigma_{\text{dateTo} > \text{CURDATE()}} (\text{Booking} \ \bowtie_{\text{Booking.hotelNo=Hotel.hotelNo}} (\sigma_{\text{hotelName} = 'Grosvenor\ Hotel'}(\text{Hotel}))))$$

   Tuple Relational Calculus:
   $$\{G \mid \text{Guest}(G) \ ((\exists B)(\exists H)(\text{Booking}(B) \wedge \text{Hotel}(H) \wedge (B.\text{dateFrom} \le \text{CURDATE()}) \wedge (B.\text{dateTo} > \text{CURDATE()}) \wedge (G.\text{guestNo} = B.\text{guestNo}) \wedge (B.\text{hotelNo} = H.\text{hotelNo}) \wedge (H.\text{hotelName} = 'Grosvenor\ Hotel')))\}$$

f. Relational Algebra:

$$\text{Room} \bowtie \left(\sigma_{\text{hotelName} = \text{'Grosvenor Hotel'}}(\text{Hotel})\right) ⟕ \Pi_{\text{guestName,hotelNo}} (\text{Guest} \bowtie_{\text{Guest.guestNo=Booking.guestNo}} (\sigma_{\text{dateFrom} \leq \text{CURDATE()}} \wedge \sigma_{\text{dateTo} > \text{CURDATE()}} \left(\text{Booking} \bowtie_{\text{Booking.hotelNo=Hotel.hotelNo}} \left(\sigma_{\text{hotelName='Grosvenor Hotel'}}(\text{Hotel})\right)\right))$$

Note to TA: ⟕ was the closest symbol to the textbooks LEFT OUTER JOIN as I could find.

Tuple Relational Calculus:

$$\{R, G.\text{guestName} \mid \text{Room}(R)\ ((\exists H)\left(\text{Hotel}(H) \wedge (H.\text{hotelNo} = R.\text{hotelNo}) \wedge \left(H.\text{hotelName} = \text{'Grosvenor Hotel'}\right)\right)) \vee (\text{Guest}(G)((\exists B)(\exists H)(\text{Booking}(B) \wedge \text{Hotel}(H) \wedge (H.\text{hotelNo} = B.\text{hotelNo}) \wedge (h.\text{hotelName} = \text{'Grosvenor Hotel'}) \wedge (G.\text{guestNo} = B.\text{guestNo}) \wedge (B.\text{dateFrom} \leq \text{CURDATE()} \wedge B.\text{dateTo} > \text{CURDATE()}\}$$

3. a. This would produce a relation containing all hotel names in the city of London.
   b. This would produce a relation containing all hotel names with room prices greater than 50 pounds.
   c. This would produce a relation containing all of the hotel names where guest John Smith has stayed. This would essentially return all hotel names where guest John Smith has had bookings.
   d. This would produce a relation containing the hotel name and guest name where there exist two bookings for the guest at the same hotel.

4.

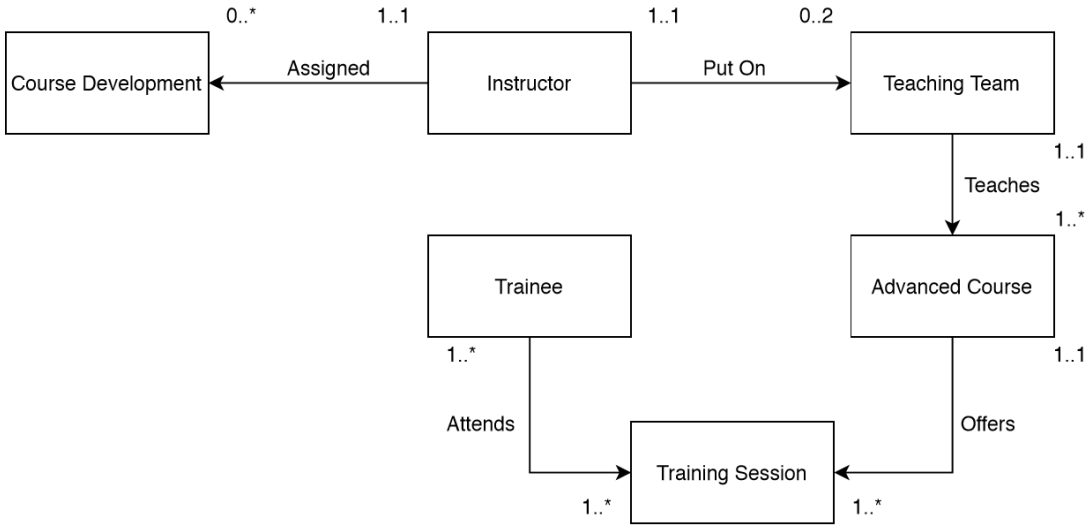
a. Main Entity Types:
   i. Instructor
   ii. Trainee
   iii. Advanced Technology Course
   iv. Training Session
   v. Course Development
   vi. Teaching Team

b.

Relationship Types and Multiplicity

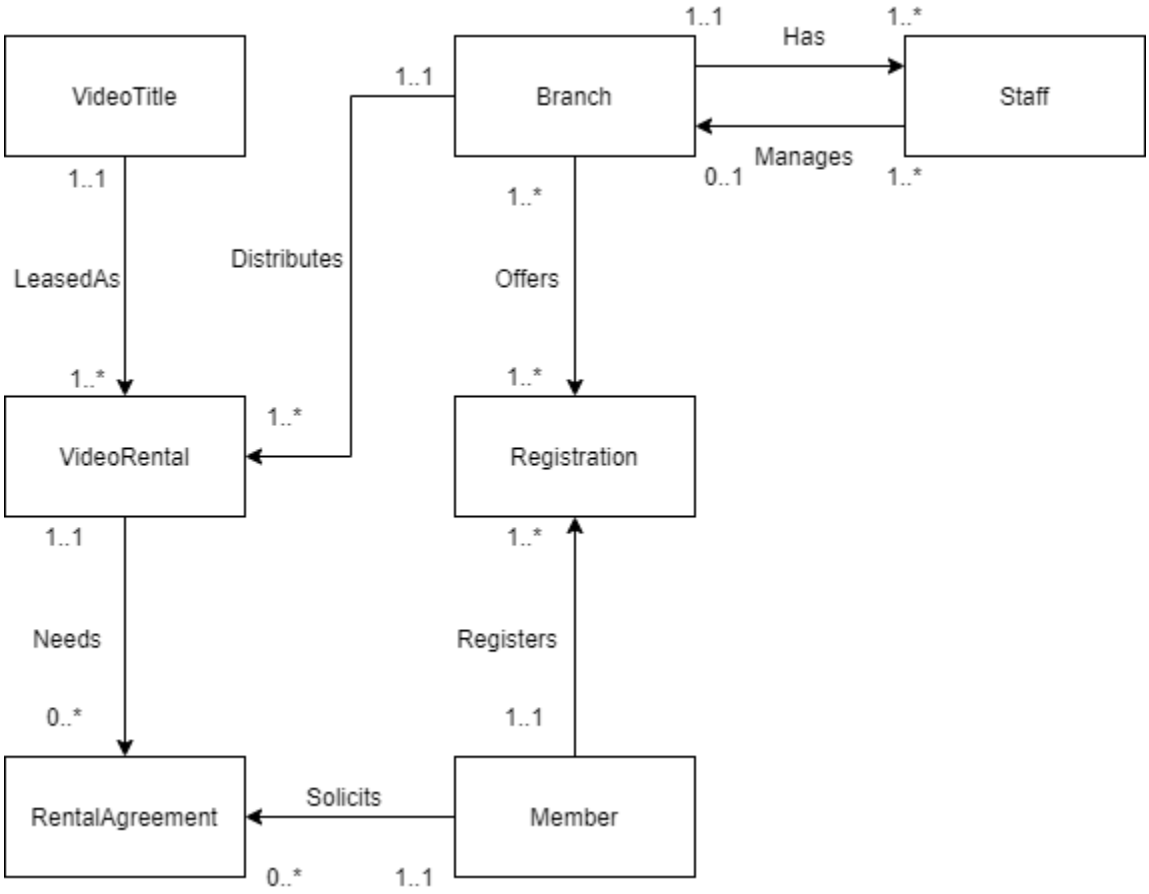
LHS Multiplicity	LHS Entity	Relationship	RHS Entity	RHS Multiplicity
1..*	Trainee	Attends	Training Session	1..*
1..1	Instructor	Put On	Teaching Team	0..2
1..1	Instructor	Assigned	Course Development	0..*
1..1	Advanced Course	Offers	Training Session	1..*
1..1	Teaching Team	Teaches	Advanced Course	1..*

c. (Made on draw.io)



5.

a. (Made on draw.io)



b. **Red denotes primary keys**

i. Branch (branchNo, address, phoneNumber, manager)
ii. Staff (staffNo, name, position, salary)
- Name contains first name and last name

iii. VideoTitle (catalogNo, title, category, dailyRentFee, purchaseCost, actorNames, directorName)
iv. VideoRental (videoNum, category, rentalStatus, catalogNo, title, dailyRentFee, purhcaseCost, actorNames, directorName)
v. Member (memberNo, firstName, lastName, address, registrationDate, branchRegistered)
vi. Registration (memberNo, branchNo)
vii. RentalAgreement(rentalNo, firstName, lastName, memberNo, videoTitle, dailyRental, dateFrom, dateTo)

c. Primary Keys:

i. Branch Entity
- Primary Key: branchNo
- Candidate Key: branchNo, address

ii. Staff Entity

- Primary Key: staffNo
- Candidate Key: staffNo, name

iii. VideoTitle
- Primary Key: catalogNo
- Candidate Key: catalogNo, {title, category}

iv. VideoRental
- Primary Key: videoNum
- Candidate Key: videoNum, {catalogNo, title}

v. Member
- Primary Key: memberNo
- Candidate Key: memberNo, {firstName, lastName}

vi. Registration
- Primary Key: None, this is a weak entity type.
- Candidate Key {memberNo, branchNo}

vii. RentalAgreement
- Primary Key: rentalNo
- Candidate Key: rentalNo, {firstName, lastName, memberNo}