University of Toronto
CSC343, Winter 2017

Assignment 1: Sample Solutions

Note that there are multiple correct answers to all of these questions. You can use query 1 as a model of good comments and formatting. We have not consistently tidied up the comments and formatting in the remaining questions. It was more important to get the solutions to you quickly.

1. Find the manufacturers who make an item whose type is a descendant of "apparel" in the subcategory hierarchy/ies. Report the manufacturer ID, name, address, and phone number.

   Impossible.

2. Let's say a "singleton order" is one that includes exactly one item. Find all gold customers who have made at least one singleton order in 2016. Report their CID, and the date and time when they made their first and their last singleton order that year.

   – Orders with one or more LineItems.
   $OnePlus(OID) :=$
   $\Pi_{OID}(Order \bowtie LineItem)$

   – Orders with two or more LineItems.
   $TwoPlus(OID) :=$
   $\Pi_{Order.OID}(\sigma_{Order.OID=L1.OID=L2.OID \atop \wedge L1.IID \neq L2.IID}(Order \times \rho_{L1} LineItem \times \rho_{L2} LineItem))$

   – Singleton orders.
   $Singleton(OID) :=$
   $OnePlus - TwoPlus$

   – Orders made by gold customers, and the customers who made them.
   $GoldOrders(CID, OID) :=$
   $\pi_{Order.CID,Order.OID}(\sigma_{membership="gold"}(Order \bowtie Customer))$

   – Gold customers who have made a singleton order in 2016.
   $GoldSingleton(CID, OID) :=$
   $\pi_{CID,OID}(\sigma_{when.year=2016}(GoldOrders \bowtie Singleton \bowtie Order))$

   – Orders by gold customers and when they were made.
   $GoldOrdersTime(OID, CID, when) :=$
   $\pi_{OID,CID,when}(GoldOrders \bowtie Order)$

   – Orders by gold customers that were not the first they made.
   $NotFirst(OID, CID, when) :=$
   $\pi_{O1.OID,O1.CID,O1.when}(\sigma_{O1.CID=O2.CID \atop \wedge O2.when<O1.when}(\rho_{O1} GoldOrdersTime \times \rho_{O2} GoldOrdersTime))$

   – Orders by gold customers that *were* the first they made.
   $First(OID, CID, when1) :=$
   $GoldOrdersTime - NotFirst$

   – Orders by gold customers that were not the last they made.
   $NotLast(OID1, CID, when) :=$
   $\pi_{O2.OID,O2.CID,O2.when}(\sigma_{O1.CID=O2.CID \atop \wedge O2.when<O1.when}(\rho_{O1} GoldOrdersTime \times \rho_{O2} GoldOrdersTime))$

   – Orders by gold customers that *were* the last they made.
   $Last(OID2, CID, when2) :=$
   $GoldOrdersTime - NotLast$

   $Answer(CID, first, last) :=$

$$\Pi_{CID,when1,when2}(GoldSingleton \bowtie First \bowtie Last)$$

3. Suppose we consider two orders to be "identical" if they contain exactly the same items (ignoring quantity). Find all pairs of customers who have made identical orders on the same day. Report each customer's CID and OID for the order that was identical. A pair could have multiple identical orders on the same day. If so, report them all.

   This solution is based on finding all the items that an order does not include. That is a lot of items, but we are not worried about efficiency.

   – A pair of orders.
   $Pairs(OID1, OID2) :=$
   $$\Pi_{O1.OID,O2.OID}(\rho_{O1}Orders \times \rho_{O2}Orders)$$

   – Item IID has been ordered by someone, but order OID does not include it.
   $DoesNotInclude(OID, IID) :=$
   $$(\Pi_{OID}Orders \times \Pi_{IID}LineItem) - (\Pi_{OID,IID}Order \bowtie LineItem)$$

   – Order OID1 includes an item that OID2 does not. (They are therefore not identical.)
   $Different(OID1, OID2) :=$
   $$\Pi_{LineItem.OID,DoesNotInclude.OID}(\sigma_{LineItem.IID=DoesNotInclude.IID}(LineItem \times DoesNotInclude))$$

   – Identical orders.
   $Identical(OID1, OID2) :=$
   $$Pairs - Different$$

   – Identical orders made on the same day.
   $IdenticalSameDay(OID1, OID2) :=$
   $$\Pi_{Identical.OID1,Identical.OID2}(\sigma_{\substack{O1.OID=Identical.OID1 \\ \wedge \\ O2.OID=Identical.OID2 \\ \wedge \\ O1.when.day=O2.when.day}}(Identical \times \rho_{O1}Order \times \rho_{O2}Order))$$

   $Answer(CID1, OID1, CID2, OID2) :=$
   $$\Pi_{O1.CID,O1.OID,O2.CID,O2.OID}(\sigma_{\substack{O1.OID=IdenticalSameDay.OID1 \\ \wedge \\ O2.OID=IdenticalSameDay.OID2 \\ \wedge \\ O1.CID<O2.CID}}(\rho_{O1}Order \times IdenticalSameDay \times \rho_{O2}Order))$$

4. Find all customers who have a silver membership, have placed at least two orders in 2014, fewer than 2 orders in 2015, and no orders at all in 2016. Report the CID.

   – All customers with a silver membership.
   $Silver(CID) :=$
   $$\Pi_{CID}(\sigma_{membership="silver"}Customer)$$

   – All customers who made at least two orders in 2014.
   $AtLeastTwo2014(CID) :=$
   $$\Pi_{O1.CID}(\sigma_{O1.CID=O2.CID \wedge O1.OID \neq O2.OID \wedge O1.when.year=2014 \wedge O2.when.year=2014}(\rho_{O1}Order \times \rho_{O2}Order))$$

   – All customers who made fewer than 2 orders in 2015.
   $AtLeastTwo2015(CID) :=$
   $$\Pi_{O1.CID}(\sigma_{O1.CID=O2.CID \wedge O1.OID \neq O2.OID \wedge O2.when.year=2015 \wedge O1.when.year=2015}(\rho_{O1}Order \times \rho_{O2}Order))$$

   $LessThanTwo2015(CID) :=$

$\Pi_{CID}Customer - AtLeastTwo2015(CID)$

– All customers who made no orders at all in 2016.
$None2016(CID) :=$
$(\Pi_{CID}Customer) - (\Pi_{CID}(\sigma_{when.year=2016}Order))$

$Answer(CID) :=$
$Silver \cap AtLeastTwo2014 \cap LessThanTwo2015 \cap None2016$

5. Let's say the "top cost" on any order is the cost of the most expensive item. (There could be several items tied for that top cost.) Among all the orders a customer places in a year, let's say their "skimpiest" order is the one whose top cost is the lowest. (There could be several orders tied for skimpiest.) For each customer who has ever placed an order, find their skimpiest order. If several orders for that customer are tied for skimpiest, report them all. Report the customer ID, order ID, and the order's top cost.

$ItemPrice(OID, IID, price) ::=$
$\Pi_{OID,IID,price}(LineItem \bowtie Item)$

$NotTopCost(OID, price) :=$
$\pi_{I1.OID,I1.price}(\sigma_{I1.OID=I2.OID \wedge I1.price<I2.price}$
$(\rho_{I1}ItemPrice \times \rho_{I2}ItemPrice))$

$TopCost(OID, price) :=$
$\Pi_{OID,price}ItemPrice - NotTopCost$

$TopCostCustomer(OID, CID, price) :=$
$\Pi_{OID,CID,price}(TopCost \bowtie Order)$

$NotSkimpy(OID, CID, price) :=$
$\Pi_{T2.OID,T2.CID,T2.price}$
$(\sigma_{T1.CID=T2.CID \wedge T1.price<T2.price}$
$(\rho_{T1}TopCostCustmer \times \rho_{T2}TopCostCustmer))$

$Answer(OID, CID, price) :=$
$TopCostCustomer - NotSkimpy$

6. Find every order that includes at least one item for which reviewers unanimously gave it a rating of 0[1] and at least one item for which reviewers unanimously gave it a rating of 5[2]. Report the customer ID, customer's last name and first name, order ID, and when the order was placed.

To find orders that unanimously had a rating of 5, find all items that were reviewed and remove items that have had a review with a non-5 rating. Same for unanimously 0. The rest is easy.

– Items that have had a review that was not a 5.
$Non5Review(IID) :=$
$\Pi_{IID}\sigma_{rating \neq 5}Review$

– Items that have unanimously been rated a 5.
$Unanimously5(IID) :=$
$\Pi_{IID}Review - Non5Review$

– Items that have had a review that was not a 0.
$Non0Review(IID) :=$
$\Pi_{IID}\sigma_{rating \neq 0}Review$

[1] An item must have been reviewed at least once in order to pass this condition.
[2] Ditto!

– Items that have unanimously been rated a 0.
$Unanimously0(IID) :=$
$\Pi_{IID} Review - Non0Review$

– Orders that include an item unanimously rated a 5.
$IncludesHigh(OID) :=$
$\Pi_{Order.OID}(Order \bowtie LineItem \bowtie Unanimously5)$

– Orders that include an item unanimously rated a 0.
$IncludesLow(OID) :=$
$\Pi_{Order.OID}(Order \bowtie LineItem \bowtie Unanimously0)$

– Orders that include both an item unanimously rated a 5 and item unanimously rated a 0.
$IncludesHighLow(OID) :=$
$IncludesHigh \cap IncludesLow$

$Answer(CID, lastName, firstName, OID, when) :=$
$\Pi_{Order.CID,Customer.lastName,Customer.firstName,Order.OID,Order.when}$
$(IncludesHighLow \bowtie Order \bowtie Customer)$

7. Find all pairs of customers c_1 and c_2 such that: c_2 has reviewed at least one item, and c_1 assessed every review of c_2 as helpful.

– If C1, C2 are to be in the answer, C1 should have assessed C2's review of IID as helpful.
$ShouldHave(C1, C2, IID) :=$
$\Pi_{reviewer,CID,IID}(\Pi_{reviewer} Helpfulness \times \Pi_{CID,IID} Review)$

– C1 did assess C2's review of IID as helpful.
$DidAssessHelpful(C1, C2, IID) :=$
$\Pi_{reader,reviewer,item}(\sigma_{helpful="yes"} Helpfulness)$

– For at least one item that C2 reviewed, CI did not assess C2's review as helpful
– (or did not assess it at all).
$NotAllHelpful(C1, C2) :=$
$\Pi_{C1,C2}(ShouldHave - DidAssessHelpful)$

$Answer(C1, C2) :=$
$(\Pi_{C1,C2} DidAssessHelpful) - NotAllHelpful$

8. For every item that has been ordered, find the last customer to order it. Report the item ID and the customer ID of the customer who ordered it last. If several customers are tied to be last to order a particular item, report a tuple for each of these customers.

$ItemOrder(IID, OID, CID, when) :=$
$\Pi_{LineItem.IID,Order.OID,Order.CID,Order.when}$
$(Order \bowtie LineItem)$

$NotLast(IID, OID) :=$
$\Pi_{I1.IID,I1.OID}(\sigma_{I1.IID=I2.IID \wedge I1.when<I2.when}$
$(\rho_{I1} ItemOrder \times \rho_{I2} ItemOrder))$

$Last(IID, OID) :=$
$\Pi_{IID,OID} ItemOrder - NotLast$

$Answer(IID, CID) :=$
$\Pi_{IID,CID}(Last \bowtie ItemOrder)$

9. Find all the customers who have given a review that at most one reader assessed as helpful. For each of these customers, find every review that had more "yes" (helpful) assessments than "no" assessments.

Report the customer ID, item ID, and item price. (A customer will appear multiple times if they have more than one qualifying review.)

Impossible.

10. Find all customers who have given at least three reviews, and for whom the rating they give has always gone down over time from review to review. (This customer has grown increasingly dissatisfied, so maybe we should reach out to him or her.) Report the customer ID, last name, and email address, and the item ID for the last item they reviewed.

    NB: As per the FAQ, all of the customer's ratings had to go down over time, not just three of them.

    – Customers who have given at least 3 reviews.
    $Three(CID) :=$
    $\Pi_{R1.CID}(\sigma_{R1.CID=R2.CID=R3.CID \wedge R1.IID<R2.IID<R3.IID}(\rho_{R1}Review \times \rho_{R2}Review \times \rho_{R3}Review))$
    – Customers who have given two reviews where the later one gave a higher (or at least equal) rating.
    – That is, the rating did NOT go down.
    $WentUp(CID) :=$
    $\Pi_{CID}\sigma_{R1.when<R2.when \wedge R1.rating<=R2.rating}(\rho_{R1}Review \times \rho_{R2}Review)$
    – Customers whose reviews the went up (or at least stayed equal).
    $AlwaysDown(CID) :=$
    $Three - WentUp$
    $Answer(CID, lastName, email) := \Pi_{CID,lastName,firstName}(AlwaysDown \bowtie Customer)$

11. A "top-level category" is one that is not a subcategory of anything else. Find all customers who have reviewed an item in each top-level category. Report just the customer ID.

    Note: An item type that has no subcategories and no parent category — it is not connected to any of the hierarchies — is considered a top-level category. We have to look in the Item relation to find these.

    – "type" is a top level category. Note that all categories that are parents of another category
    – could be top-level, and so could categories defined in Item (which may not occur in
    – Subcategory at all). Also note that, in this query, we must rename the attributes in order to have
    – syntactically valid set operations.
    $TopLevel(type) :=$
    $(\rho_{Parent(category)}\Pi_b Subcategory \cup \rho_{Used(category)}\Pi_{type} Item) - (\rho_{Child(category}\Pi_a Subcategory)$
    $TypeReview(CID, type) :=$
    $\Pi_{CID,type}(Item \bowtie Review)$
    $ShouldHave(CID, type) :=$
    $(\Pi_{CID}Customer \times TopLevel)$
    $Answer(CID) :=$
    $\Pi_{CID}Customer - (\Pi_{CID}(ShouldHave - TypeReview))$

12. Find the orders with at least one item, and for which every item on the order had a type that was either "book" or a direct a subcategory of "book". Report the order ID.

    – Orders with at least one item.
    $AtLeastOneItem(OID) :=$

$\Pi_{OID} LineItem$

– Items that are a book or a subcategory of a book.
$Bookish(IID) :=$
$(\Pi_{IID}\sigma_{type="book"} Item) \cup$
$\Pi_{Item.IID}(\sigma_{Subcategory.b="book" \wedge Item.type=Subcategory.a}$
$(Item \times Subcategory))$

– Items that are not bookish.
$NotBookish(IID) :=$
$\Pi_{IID} Item - Bookish$

– Orders that include something that is not bookish.
$NotAllBookishOrder(OID) :=$
$\Pi_{OID}(Order \bowtie LineItem \bowtie NotBookish)$

– Orders for which everything is bookish.
$BookishOrder(OID) :=$
$\Pi_{OID} Order - NotAllBookishOrder$

$Answer(OID) :=$
$AtLeastOneItem \cap BookishOrder$

13. Find the orders with more than three items, and for which at least half of the items have a category that is not "book". Report the order ID, customer ID, and the credit that they used.

    Impossible.

Part 2: Additional 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 customer who reviews an item must have ordered that item.

   $Review(CID, IID) - \Pi_{CID,IID}(LineItem \bowtie Order) = \emptyset$

2. Orders made by gold members have no limit on the items that can be included. However, orders made by silver members must include at least one item costing over $50, and orders made by non-members cannot include any items costing under $50.

   Notice that we express this one as two separate constraints. We could have unioned the CIDs together and made it into one constraint.

   $\Pi_{CID}(\sigma_{membership="silver"} Customer \bowtie Order) -$
   $\Pi_{CID}\sigma_{membership="silver" \wedge price>50}\ (LineItem \bowtie Order \bowtie Item \bowtie Customer) = \emptyset$

   $\Pi_{CID}(\sigma_{membership="none"} Customer \bowtie Order) \bowtie$
   $\Pi_{CID}\sigma_{membership="none" \wedge price<50}(LineItem \bowtie Order \bowtie Item \bowtie Customer) = \emptyset$