University of Toronto
CSC343, Winter 2017

Assignment 1: Shadman Anam Siddiqui 1001534066 Lino Lastella 1001237654

Part 1: Queries

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.

   Cannot be expressed.

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.

   – Gold costumers who made at least an order in 2016.

   $Gold2016(OID, CID, when) := \Pi_{OID,CID,when}(\sigma_{membership='gold'}Customer \bowtie \sigma_{when.year=2016}Order)$

   $NotSingletons(OID) := \Pi_{L1.OID}\sigma_{L1.OID=L2.OID \wedge L1.IID \neq L2.IID}[(\rho_{L1}LineItem)\times(\rho_{L2}LineItem)]$

   $Singleton(OID) := (\Pi_{OID}LineItem) - NotSingletons$

   – Gold customers who made at least one singleton in 2016

   $GS(OID, CID, when) := \sigma_{Gold2016.OID=Singleton.OID}(Gold2016 \times Singleton)$

   $NotLast(OID, CID, when) := \Pi_{G1.OID,G1.CID,G1.when}\sigma_{G1.when<g2.when}[(\rho_{G1}GS)\times(\rho_{G2}GS)]$

   $NotFirst(OID, CID, when) := \Pi_{G1.OID,G1.CID,G1.when}\sigma_{G1.when>g2.when}[(\rho_{G1}GS)\times(\rho_{G2}GS)]$

   $Last(CID, when) := \Pi_{CID,when}(GS - NotLast)$

   $First(CID, when) := \Pi_{CID,when}(GS - NotFirst)$

   $Answer(CID, when) := \Pi_{First.CID,First.when,Second.when}\sigma_{First.CID=Last.CID}(First \times Last)$

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.

   – tuples for each instance an item ordered, the order number, and time

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

– cross LineItem with all orders in LineItem to represent pairs of orders that should exist if all orders were identical and on the same day

$Should(OID1, IID, OID2) := LineItem \times \Pi_{OID} LineItem$

– tuple contains item and two orders that both contain it in the same day

$Actual(OID1, IID, OID2) :=$

$$\Pi_{i1.OID, i1.IID, i2.OID} \sigma_{\substack{i1.when.day=i2.when.day \wedge \\ i1.OID>i2.OID \wedge \\ i1.IID=i2.IID}} \rho_{i1} ItemOrder \times \rho_{i2} ItemOrder$$

– pairs of orders in which first contains an item that the second doesn't, i.e. not identical

$NotIdentical(OID1, OID2) := \Pi_{OID1, OID2}(Should - Actual)$

– identical orders

$Identical(OID1, OID2) := (\Pi_{OID1, OID2} Actual) - NotIdentical$

– customer IDs and Order Ids of pairs of identical orders

$Answer(CID1, OID1, CID2, OID2) :=$

$$\Pi_{\substack{O1.CID, \\ OID1, \\ O2.CID, \\ OID2}} \sigma_{\substack{OID1=O1.OID \wedge \\ OID2=O2.OID \wedge \\ O1.CID!=O2.CID}} Identical \times \rho_{O1} Order \times \rho_{O2} Order$$

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.

– customers who made any orders in 2016

$No2016(CID) := (\Pi_{CID} Customer) - (\Pi_{CID} \sigma_{when.year=2016} Order)$

– customers who made at least two orders in 2014

$AtLeastTwo2014(CID) := \Pi_{O1.CID} \sigma_{\substack{O1.OID \neq O2.OID \wedge O1.CID=O2.CID \wedge \\ O1.when.year=2014 \wedge O2.when.year=2014}} [(\rho_{O1} Order) \times (\rho_{O2} Order)]$

– customers who made at least two orders in 2015

$AtLeastTwo2015(CID) := \Pi_{O1.CID} \sigma_{\substack{O1.OID \neq O2.OID \wedge O1.CID=O2.CID \wedge \\ O1.when.year=2015 \wedge O2.when.year=2015}} [(\rho_{O1} Order) \times (\rho_{O2} Order)]$

– customers who made fewer than 2 orders in 2015

$FewerThanTwo2015(CID) := (\Pi_{CID} Costumer) - (AtLeastTwo2015)$

- customers who don't break any of the conditions

$$Answer(CID) := (\Pi_{CID} \sigma_{membership='silver'} Costumer) \cap AtLeastTwo2014 \cap FewerThanTwo2015 \cap No2016$$

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.

– order, specific item in order and it's price

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

– item which is not the highest priced in order

$NotTopCost(OID, IID, price) := \Pi_{\substack{O1.OID, \\ O1.IID, \\ O1.price}} \sigma_{\substack{O1.OID=O2.OID \wedge \\ O1.IID \neq O2.IID \wedge \\ O1.price < O2.price}} [(\rho_{O1} OrderPrices) \times (\rho_{O2} OrderPrices)]$

– highest priced item in each order

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

– top cost of each order and customer who ordered

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

– order which is not the customer's skimpiest

$NotSkimpiest(CID, OID, price) :=$

$\Pi_{\substack{C1.CID, \\ C1.OID, \\ C1.price}} \sigma_{\substack{C1.CID=C2.CID \wedge \\ C1.OID \neq C2.OID \wedge \\ C1.price > C2.price}} [(\rho_{C1} CustomerTopCost) \times (\rho_{C2} CustomerTopCost)]$

– skimpiest order with customer and price

$Skimpiest(CID, OID, price) := CustomerTopCost - NotSkimpiest$

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.

– item that was ever rated non-zero

$NotZeroRating(IID) := \Pi_{IID} \sigma_{rating \neq 0} Review$

– item that has only ever been rated 0

$ZeroRating(IID) := (\Pi_{IID} Review) - NotZeroRating$

– item that ever recieved non-5 rating

$NotFiveRating(IID) := \Pi_{IID} \sigma_{rating \neq 5} Review$

– item that only recieved ratings of 5

$FiveRating(IID) := (\Pi_{IID} Review) - NotFiveRating$

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

– orders that contained a all-zero-rated item

$ZeroOrders(OID) := \Pi_{OID}(LineItem \bowtie ZeroRating)$

– orders that contained an all-five-rated item

$FiveOrders(OID) := \Pi_{OID}(LineItem \bowtie FiveRating)$

–orders that contained an all-zero- and an all-five-rated item $Answer(CID, lastName, firstName, OID,$

$$\Pi_{\substack{CID,lastName,\\firstName,OID,when}}[(FiveOrders \cap ZeroOrders) \bowtie Order \bowtie Customer]$$

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.

   – tuples for each review(defined by reviewer and item) crossed with each reader; representing relation that should exist if every reader found every helpful

   $Should(reviewer, item, reader) := (\Pi_{CID,IID} Review) \times (\Pi_{reader} Helpfulness)$

   – tuples of reviewer and reader in which reader found any review of reviewer unhelpful or simply didn't read it

   $NotAnswer(reviewer, reader) :=$
   $\Pi_{reviewer,reader}[Should - (\Pi_{reviewer,item,reader} \sigma_{helpful=\text{``yes''}} Helpfulness)]$

   – tuples of each reviewer with each reader who found every review helpful

   $Answer(c_1, c_2) := \Pi_{reader,reviewer}(\Pi_{reviewer,reader} Helpful - NotAnswer)$

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.

   – get each ID of each item ordered the order ID, time, and ordering customer ID

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

   – tuples for all orders of an item which weren't the last time it was ordered

   $NotLastOrder(CID, OID, IID, when); =$
   $\Pi_{\substack{I1.CID,I1.OID,\\I1.IID,I1.when}} \sigma_{\substack{I1.IID=I2.IID\wedge\\I1.when>I2.when}}[(\rho_{I1} ItemsOrdered) \times (\rho_{I2} ItemsOrdered)]$

   – each item ordered with the last customer(s) to order it

   $LastOrder(CID, IID) := \Pi_{CID,IID}(ItemsOrdered - NotLastOrdered)$

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.)

Cannot be expressed.

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.

$$ThreeReviews(CID, item1, rating1, when1, item2, rating2, when2) :=$$
$$\Pi_{\substack{R1.CID, R1.IID, \\ R1.rating, R1.when, \\ R2.IID, R2.rating, \\ R2.when}} \sigma_{\substack{R1.CID=r2.CID=R3.CID\wedge \\ R1.IID\neq R2.IID\wedge \\ R2.IID\neq R3.IID\wedge \\ R1.IID\neq R3.IID}} [(\rho_{R1}Review)\times(\rho_{R2}Review)\times(\rho_{R3}Review)]$$

– out of customers with three reviews, find the ones whose review ever increased over time or even stayed the same

$$NotDecreasingReviews(CID) := \Pi_{CID}\sigma_{when1>when2\wedge rating1\geq rating2}ThreeReviews$$

– Filter ThreeReviews relation to only those CIDs whose reviews always decreased over time

$$DecreasingReviews(CID, item1, rating1, when1, item2, rating2, when2) :=$$
$$ThreeReviews \bowtie (\Pi_{CID}ThreeReviews - NotDecreasingReviews)$$

– find items reviews by each decreasing review customer which are not his/her last item reviewed

$$NotLastItem(CID, IID) := \Pi_{CID,item1}\sigma_{when1<when2}DecreasingReviews$$

– tuples for all items reviewed by each decreasing review customer who reviewed at least three items; candidates for last reviewed items

$$ItemsReviewed(CID, IID) := \Pi_{CID,item1}DecreasingReviews$$

– decreasing review customers who reviewed at least thrice and the last item reviewed

$$LastReviewed(CID, IID) := ItemsReviewed - NotLastItem$$

– decreasing review customers who reviewed at least thrice, with last name, email and last item reviewed

$$Answer(CID, lastName, email, lastItem) := \Pi_{\substack{CID,lastName, \\ email,IID}}LastReviewed \bowtie Customer$$

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.

– top-level categories

$$TopLevel(type) := \Pi_{b}Subcategory - \Pi_{a}Subcategory$$

– every customer who has reviewed anything crossed multiplied with every top-level category to show what categories they should have reviewed to be in the answer

$Should(CID, type) := \sigma_{SID} Review \times TopLevel$

– tuples of customers and the categories they actually bought

$Actual(CID, type) := \Pi_{CID,type} Review \bowtie Item \bowtie TopLevel$

– find difference of should and actual to get customers who didn't review every type they should have to be in answer

$NotAnswer(CID) := \Pi_{CID}(Should - Actual)$

– get answer from difference of candidates and those who don't pass

$Answer(CID) := (\Pi_{CID} Review) - NotAnswer$

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.

– tuples of items ordered along with order ID, type of item and parent type; candidates for answer

$OrderItems(OID, IID, type, parent) := \Pi_{\substack{OID,IID,\\ type,b}} \sigma_{type=\text{“}a\text{”}} [(LineItem \bowtie Item) \times Subcategory]$

– orders with at least one item of a type that is neither "book" nor a subcategory of "book"; don't belong in answer

$NotAnswer(OID) := \Pi_{OID} \sigma_{type != \text{“}book\text{”} \wedge parent != \text{“}book\text{”}} OrderItems$

$Answer(OID) := \Pi_{OID} OrderItems - NotAnswer$

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.

Cannot be expressed.

Part 2: Additional Integrity Constraints

1. A customer who reviews an item must have ordered that item.

   $(\Pi_{CID} Orders - \Pi_{CID} Reviews) = \emptyset$

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.

   – silver members constraint

   – orders made by silver members

   $SilverOrders(OID) := \Pi_{OID} \sigma_{member=\text{``}silver\text{''}} (Order \bowtie Customer)$

   – orders made by silver members which had at least one item that was over 50

   $SilverOver50(OID) := \Pi_{OID} \sigma_{member=\text{``}silver\text{''} \wedge price>50} Customer \bowtie Order \bowtie Item$

   – orders made by silver members which had no item over 50; this should be empty

   $Under50SilverOrders(OID) := SilverOrders - SilverOver50$

   $Under50SilverOrders = \emptyset$

   – non-members constraint

   – all orders made by non-members

   $NonMemberOrders(OID) := \Pi_{OID} \sigma_{member=\text{``}none\text{''}} Order \bowtie Customer$

   – orders by non-members which have any item less than 50; must be empty

   $(\sigma_{price<50} NonMemberOrders \bowtie LineItem \bowtie Item) = \emptyset$