Ryan Riopelle

Due: March 19, 2016

**MAS 201 – Homework 4 – Sales**

**Step 1) Original Query**

*SELECT cate.category\_id,cust.customer\_id,sum(quantity),sum(price) FROM*

*(SELECT category\_id,sum(price) AS dollar\_value FROM*

*sales.category NATURAL JOIN sales.product NATURAL JOIN sales.sale*

*GROUP BY category\_id ORDER BY dollar\_value DESC limit 10) AS cate,*

*(SELECT customer\_id,sum(price) AS dollar\_value FROM sales.sale*

*GROUP BY customer\_id ORDER BY dollar\_value DESC limit 10) AS cust, sales.sale s,sales.product p*

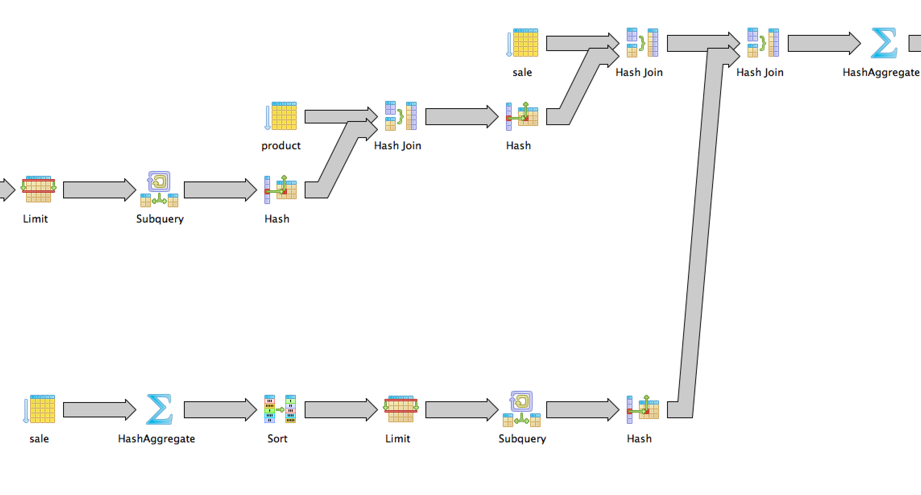
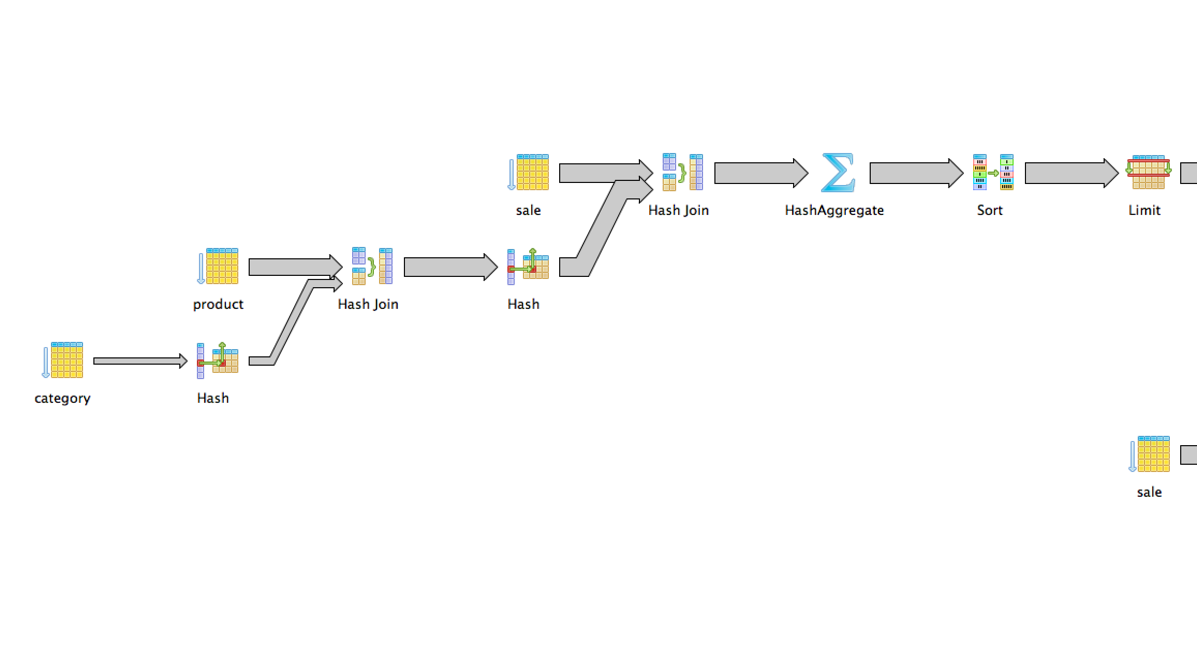
*WHERE p.category\_id = cate.category\_id and s.customer\_id = cust.customer\_id and s.product\_id = p.product\_id*

*GROUP BY cate.category\_id,cust.customer\_id ORDER BY cate.category\_id*

**Size**

"Sort (cost=206072.90..206073.10 rows=80 width=18)"

**Visual Representation**



**Step 2) Create Pre-computed Tables**

*DROP TABLE IF EXISTS sales.PRE\_Category\_Sale ;*

*create table sales.PRE\_Category\_Sale as*

*SELECT category\_id,sum(price) AS dollar\_value*

*FROM sales.category NATURAL*

*INNER JOIN sales.product NATURAL*

*INNER JOIN sales.sale*

*GROUP BY category\_id;*

*DROP TABLE IF EXISTS sales.PRE\_Customer\_Sale;*

*create table sales.PRE\_Customer\_Sale as*

*SELECT customer\_id,sum(price) AS dollar\_value*

*FROM sales.sale*

*GROUP BY customer\_id;*

**Step 3) Create ID’s**

*CREATE INDEX pre\_category\_sale\_id\_key*

*ON sales.pre\_category\_sale*

*USING btree*

*(dollar\_value DESC, category\_id);*

*CREATE INDEX category\_id\_product\_id\_key*

*ON sales.product*

*USING btree*

*(category\_id, product\_id);*

*CREATE INDEX product\_id\_category\_id\_key*

*ON sales.product*

*USING btree*

*(product\_id, category\_id);*

*CREATE INDEX product\_id\_quantity\_price\_id*

*ON sales.sale*

*USING btree*

*(product\_id, quantity, price);*

**Step 4) Create Triggers Using Trigger for product\_id=** **87943**

**--Trigger Table For Category Sales**

*CREATE or replace FUNCTION sales.FN\_PRE\_Category\_Sale() RETURNS trigger AS $FN\_PRE\_Category\_Sale$*

*BEGIN*

*UPDATE sales.PRE\_Category\_Sale as pre*

*SET dollar\_value = pre.dollar\_value + NEW.price*

*FROM sales.product as p*

*WHERE pre.category\_id = p.category\_Id*

*AND p.product\_id = NEW.product\_id;*

*IF NOT FOUND THEN*

*--no category\_id updated*

*INSERT INTO sales.PRE\_Category\_Sale (category\_id, dollar\_value)*

*SELECT p.category\_id, NEW.price*

*FROM sales.product p*

*WHERE p.product\_id = NEW.product\_id;*

*END IF;*

*RETURN NEW;*

*END;*

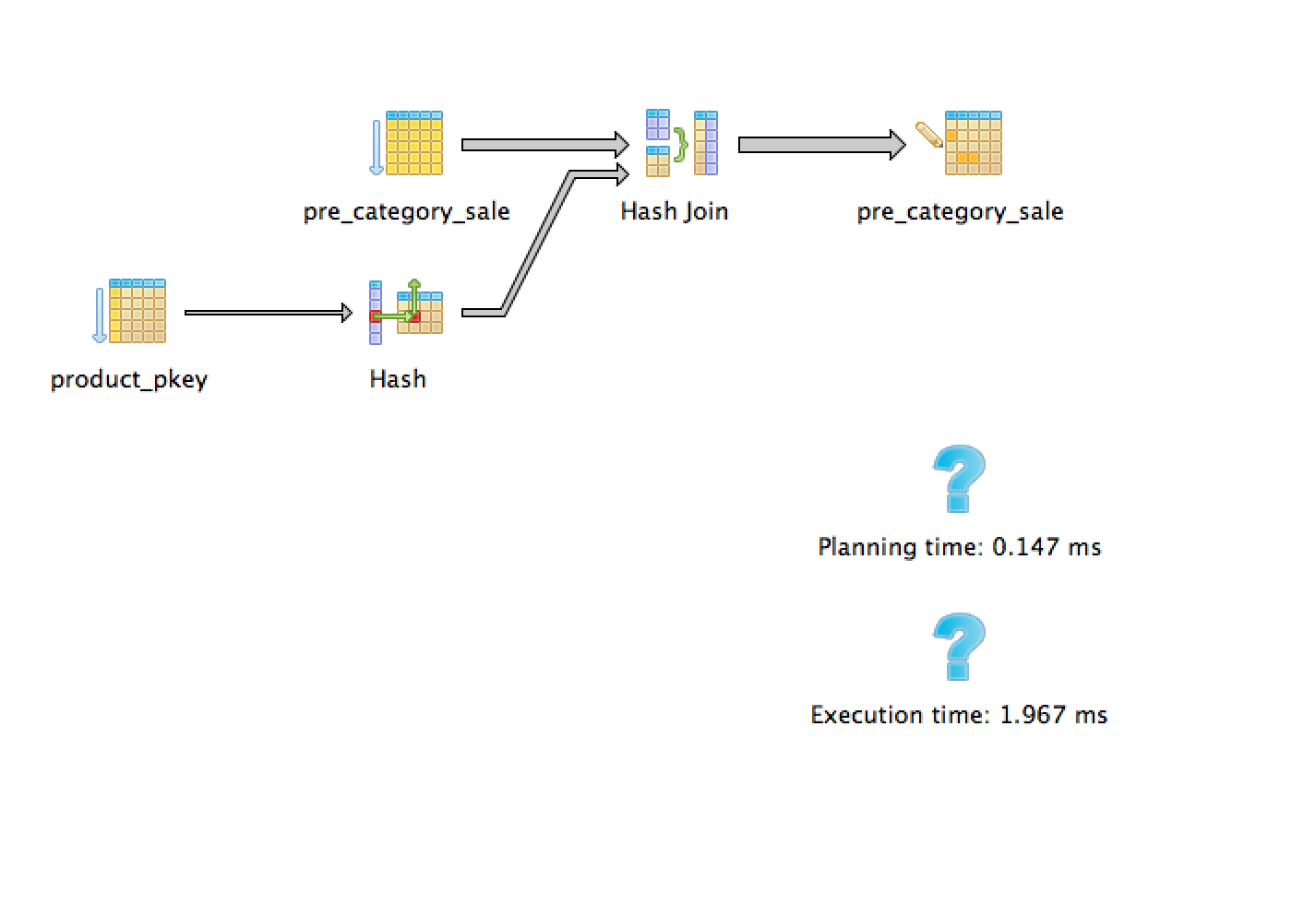
*$FN\_PRE\_Category\_Sale$ LANGUAGE plpgsql;*

*DROP TRIGGER IF EXISTS trigger\_Sale\_Category ON sales.sale ;*

*CREATE TRIGGER trigger\_Sale\_Category*

*BEFORE INSERT ON sales.sale*

*FOR EACH ROW EXECUTE PROCEDURE sales.FN\_PRE\_Category\_Sale();*



"Update on pre\_category\_sale pre (cost=8.45..105.21 rows=1 width=24) (actual time=1.918..1.918 rows=0 loops=1)"

**-- Trigger Table For Customer Sale**

*CREATE or replace FUNCTION sales.FN\_PRE\_Customer\_Sale() RETURNS trigger AS $FN\_PRE\_Customer\_Sale$*

*BEGIN*

*UPDATE sales.PRE\_Customer\_Sale*

*SET dollar\_value = dollar\_value + NEW.price*

*WHERE customer\_id = NEW.customer\_id;*

*IF NOT FOUND THEN*

*--no category\_id updated*

*INSERT INTO sales.PRE\_Customer\_Sale (customer\_id, dollar\_value)*

*SELECT NEW.customer\_id, NEW.price;*

*END IF;*

*RETURN NEW;*

*END;*

*$FN\_PRE\_Customer\_Sale$ LANGUAGE plpgsql;*

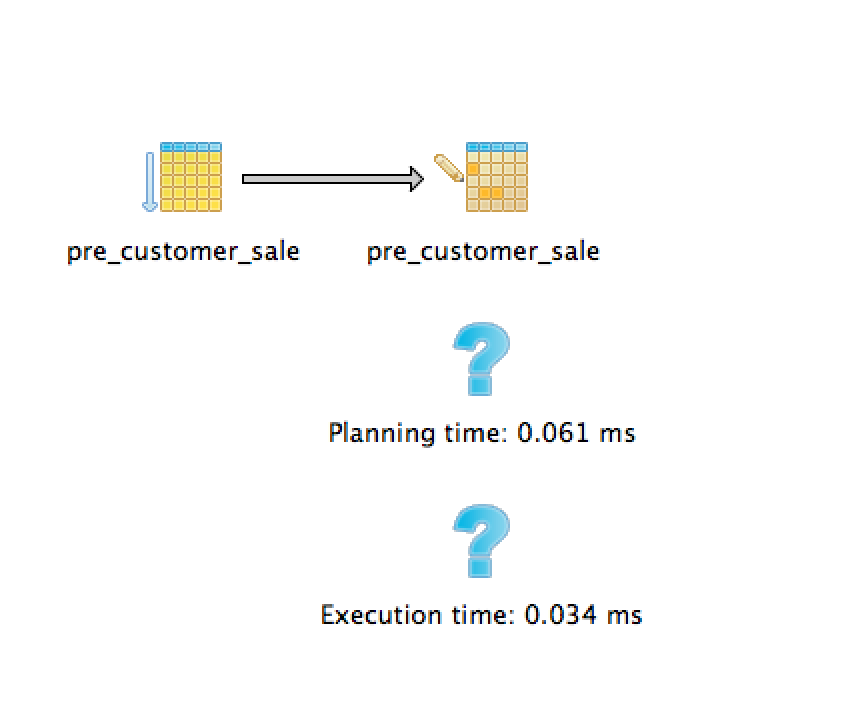
*DROP TRIGGER IF EXISTS trigger\_Sale\_Customer ON sales.sale ;*

*CREATE TRIGGER trigger\_Sale\_Customer*

*BEFORE INSERT ON sales.sale*

*FOR EACH ROW EXECUTE PROCEDURE sales.FN\_PRE\_Customer\_Sale();*

"Update on pre\_customer\_sale (cost=0.00..25.39 rows=6 width=42) (actual time=0.010..0.010 rows=0 loops=1)"



**Step 4) Run Query that uses pre-computed tables**

"Sort (cost=15658.99..15659.24 rows=100 width=18) (actual time=43.675..43.683 rows=80 loops=1)"

*WITH cate AS (SELECT category\_id FROM sales.PRE\_Category\_Sale ORDER BY dollar\_value DESC limit 10),cust as (SELECT customer\_id*

*FROM sales.PRE\_Customer\_Sale ORDER BY dollar\_value DESC limit 10)*

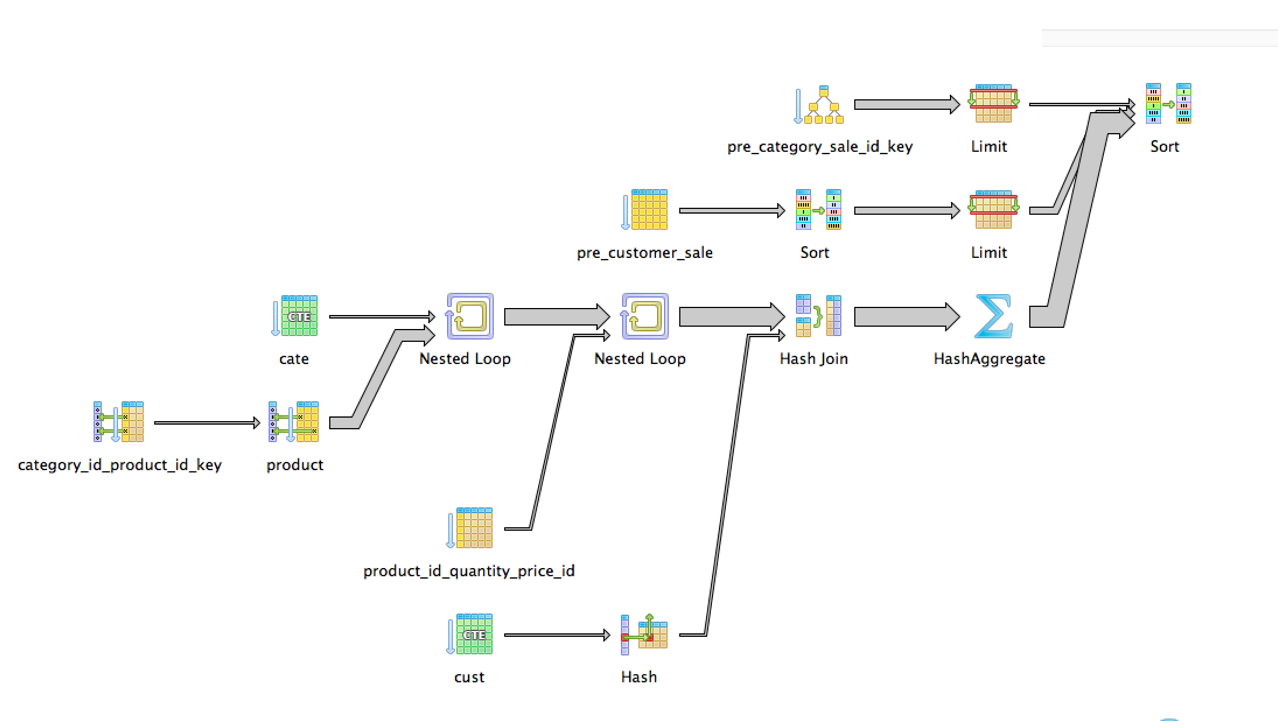
*SELECT cate.category\_id,cust.customer\_id,sum(quantity),sum(price)*

*FROM cate,cust,sales.sale s,sales.product p*

*WHERE p.category\_id = cate.category\_id AND s.customer\_id = cust.customer\_id AND s.product\_id = p.product\_id*

*GROUP BY cate.category\_id,cust.customer\_id*

*ORDER BY cate.category\_id*

****

**Conclusion:**

|  |  |
| --- | --- |
| **Trigger Maintenance Query Cost** | |
| **Category Sale** | **Customer Sale** |
| 105 | 25.39 |

The total cost is 105+25=130. Compared to the original query of 206072 there is a large difference which means the trigger can perform frequently and still have a lower overall cost than running the original query. This value is (206072-130)/130=1584.