**Answers\_Assignment 9**

Wanyi Su

**Question 1.**

*Query:*

SELECT p2.mtype AS payment\_method, SUM(p1.amount)/COUNT(c.custid) AS average\_amount

FROM purchases p1 JOIN paymentmethods p2

ON p1.pmid = p2.pmid

JOIN customers c

ON p2.custid = c.custid

WHERE c.province = 'ON'

GROUP BY p2.mtype;

*Results:*

Background pattern

Description automatically generated with medium confidence

(A custid can have multiple pmid)

From the result, although average credit purchase amount by Ontario people is slight more than average debit purchase amount, the two purchase methods don't differ too much.

**Question 2.**

*a. original answer:*

Graphical user interface, application

Description automatically generated

Since there are three groups of people, we can define three set of conditions for "from\_bc\_non\_van" and "from\_van" to define these three groups of people. From the result of average and median, we can conclude that visitors from outside BC spend the most per transaction.

*b. view query:*

CREATE VIEW vancouver\_custs AS

WITH

vprefixes (vp) AS

(SELECT DISTINCT pcprefix FROM greater\_vancouver\_prefixes)

SELECT c1.custid AS custid,

(CASE WHEN c1.pcode3 = v.vp THEN 1 ELSE 0 END) AS in\_vancouver

FROM ( (SELECT custid, city, SUBSTRING(postalcode, 1, 3) AS pcode3

FROM customers) AS c1

LEFT OUTER JOIN vprefixes v

ON c1.pcode3 = v.vp );

*c. original query:*

SELECT (CASE WHEN (v1.in\_vancouver = 0 AND c2.province = 'BC') THEN True ELSE False END) AS From\_BC\_non\_Van,

(CASE WHEN v1.in\_vancouver = 1 THEN True ELSE False END) AS From\_Van,

COUNT(p.purchid) AS Count,

SUM(p.amount)/COUNT(p.purchid) AS Average,

MEDIAN(p.amount) AS Median

FROM purchases p JOIN vancouver\_custs v1

ON p.custid = v1.custid

JOIN customers c2

ON v1.custid = c2.custid

GROUP BY (CASE WHEN (v1.in\_vancouver = 0 AND c2.province = 'BC') THEN True ELSE False END), (CASE WHEN v1.in\_vancouver = 1 THEN True ELSE False END)

ORDER BY MEDIAN(p.amount) ASC;

**Question 3.**

*a. answer:*

Graphical user interface

Description automatically generated with low confidence

From the result, tourists spend more at restaurants that serve sushi.

*b. query:*

WITH sushi AS

(SELECT amenid

FROM amenities

WHERE tags.cuisine ILIKE '%sushi%' AND amenity = 'restaurant')

SELECT SUM(p.amount)/COUNT(v4.custid) AS avg,

(CASE WHEN v4.in\_vancouver = 1 THEN 1 ELSE 0 END) AS in\_vancouver

FROM (sushi s JOIN amenities a

ON s.amenid = a.amenid

JOIN purchases p

ON a.amenid = p.amenid

JOIN vancouver\_custs v4

ON p.custid = v4.custid)

GROUP BY (CASE WHEN v4.in\_vancouver = 1 THEN 1 ELSE 0 END)

ORDER BY (CASE WHEN v4.in\_vancouver = 1 THEN 1 ELSE 0 END);

**Question 4.**

1. *Answer:*

Graphical user interface, application

Description automatically generated

1. *Query:*

SELECT pdate, AVG(amount) AS avg

FROM s3ext.purchases

WHERE DATE\_PART(mon, pdate) = 8 AND DATE\_PART(d, pdate) IN (1,2,3,4,5)

GROUP BY pdate

ORDER BY DATE\_PART(d, pdate) ASC;

Using “Purchases”:

Graphical user interface, text

Description automatically generated with medium confidence

Graphical user interface, application

Description automatically generated Graphical user interface

Description automatically generated

Using “s3ext.purchases”:

Graphical user interface, text

Description automatically generated

Graphical user interface, application

Description automatically generated

Graphical user interface, application

Description automatically generated

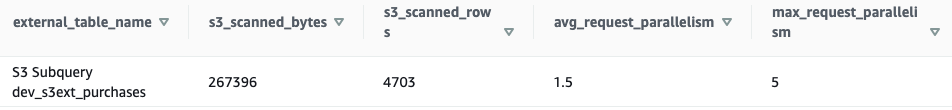
1. *What was the bytes / record ratio for Redshift on the 5-day query?*

94.06/4703 \* 1000 = 20

1. What was the bytes / record ratio for Spectrum on the 5-day query?

267396/4703 = 56.856

pg\_catalog.svl\_s3query\_summary:



Graphical user interface, text

Description automatically generated with medium confidence Graphical user interface

Description automatically generated

1. *For this purchase dataset, the averages are 57 bytes/line and 968 lines/day. (It may be useful to explore the*[*public-cmpt-732 bucket*](https://s3.console.aws.amazon.com/s3/buckets/public-cmpt-732?region=us-west-2&tab=objects)*to derive these for yourself.) From these values, what might you infer about how Redshift scans the table? How Spectrum scans the table?*

Redshift scans only the parts of rows required by the query in the table, which are the “pdate” and “amount” columns in this question, since the bytes/record ratio is only 0.02 bytes/row.

Spectrum scans the whole table (which means it scans each line fully), since the bytes / record ratio is about 56 bytes/row.

1. *Based on these computations and the parallelism results, what properties of a dataset might make it well-suited to loading from S3 into Redshift before querying it?*

Since Redshift scans only what related to the query and charges based on cluster computation time, we prefer to use Redshift under these conditions:

If the dataset is well-structured, and our query only relates to a small number of columns.

If we need to use the dataset very often (for example, we need to query on the dataset every day), we prefer to use Redshift.

1. *Conversely, what properties of a dataset might make it well-suited to retaining in S3 and querying it using Spectrum?*

Since we are paying for the number of bytes read when using Spectrum:

If the dataset is not structured, or if the dataset is very large.

If the computation is heavy and we need to execute sophisticated queries on the exabyte level dataset in a short period of time (because Spectrum allows for parallelism, and it manages the dataset on nodes concurrently).

If we need to do quite a lot of queries on the large dataset in a short period of time (not spread in a large time range).