Index Report Project 2

Configuration Settings :

- `shared_buffers` = 256MB
- `work_mem` = 96MB
- `seq_page_cost / random_page_cost` ratio is 1, making index usage more likely to be chosen by PostgreSQL
- VM memory : 1GB
- 1 processor
- Storage Device : 512GB SSD

Initial Choice of Indices :

- small database : sales.uid, sales.pid, product.cid, product.name
- large database : sales.uid, sales.pid, product.cid, product.name, user.state

Indices that were retained :

- small database : sales.uid, sales.pid, product.cid, product.name
- large database : sales.uid, sales.pid, product.cid, product.name

Indices that were not :

- user.state
- user.state

Running Times :

All running times are available in files located in the `index_report/` folder. Those files contains the query plan of the various queries used for each configuration. They are generated through the `run_experiments.sh` program and are directly executing their namesake SQL files located under `sql_files/` directory.

* : all query plans are available in the `index_report` folder in the solution.

Index Justification

All chosen examples are valid on both small and large databases and explain outputs are provided for both database sizes each time.

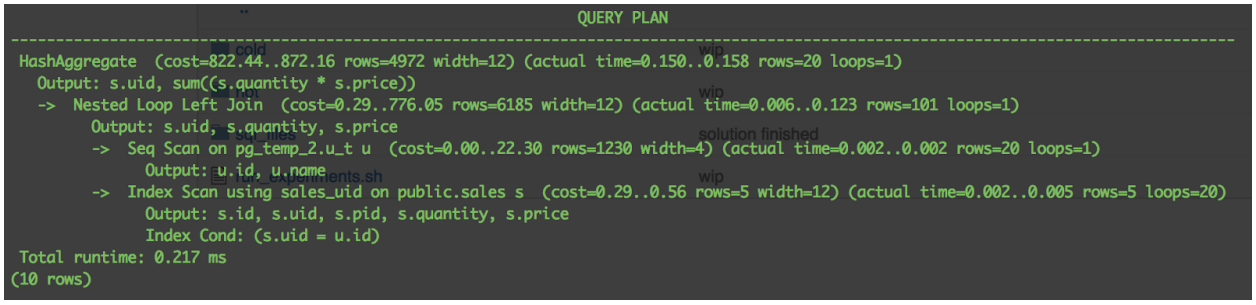
1 Sales UID :

1.1 SQL Query in which the index is useful:

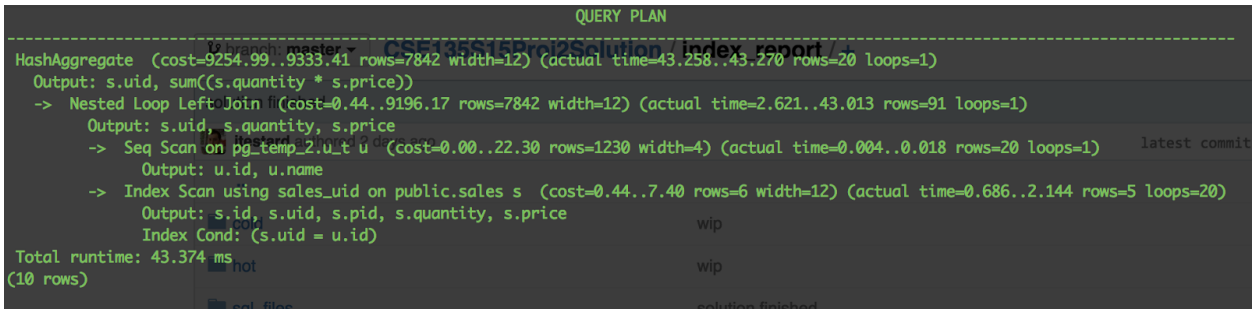
```
CREATE TEMP TABLE u_t (id int, name text) ON COMMIT DELETE ROWS;
insert into u_t (id, name) select id,name
from users order by name asc offset 0 limit 20;
select s.uid, sum(s.quantity*s.price)
from  u_t u LEFT OUTER JOIN sales s ON s.uid=u.id group by s.uid;
```

2.1 Verbal explanation :
The proportion of the sales which belong to the top 20 user represents a very small fraction of the total number of sales. For example, on the large database the top 20 users have completed on average 100 purchases, yet the sales table has 25 million entries. This motivates the use of an index on `sales.uid`.

Explain Output for small database :



Explain Output for large database :



2 Products Name :

* : all query plans are available in the `index_report` folder in the solution.

2.1 SQL Query in which the index is useful:

```
CREATE TEMP TABLE p_t (id int, name text)ON COMMIT DELETE ROWS;
insert into p_t (id, name) select id,name from products order by name
asc offset 0 limit 10;
```

2.2 Experiment or Justification :

The top 10 products name must be fetched when ordering by name, and running the above query requires scanning and sorting the entire products table (100 000 entries on the large database). This work can easily be avoided with an index.

3 Product Cid and Sales Pid :

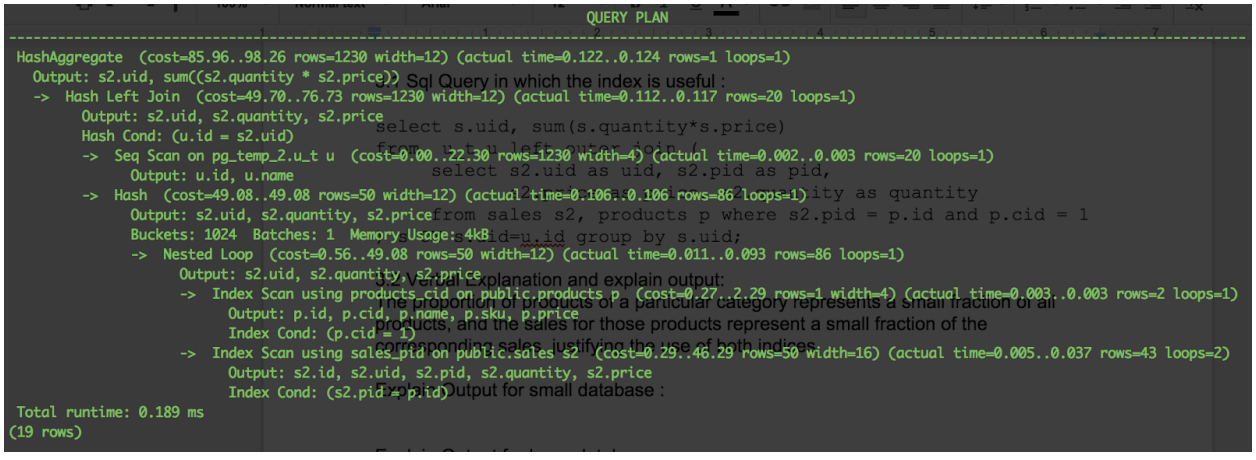
3.1 Sql Query in which the index is useful :

```
select s.uid, sum(s.quantity*s.price)
from  u_t u left outer join (
     select s2.uid as uid, s2.pid as pid,
            s2.price as price, s2.quantity as quantity
     from sales s2, products p where s2.pid = p.id and p.cid = 1
) s ON s.uid=u.id group by s.uid;
```

3.2 Verbal Explanation and explain output:

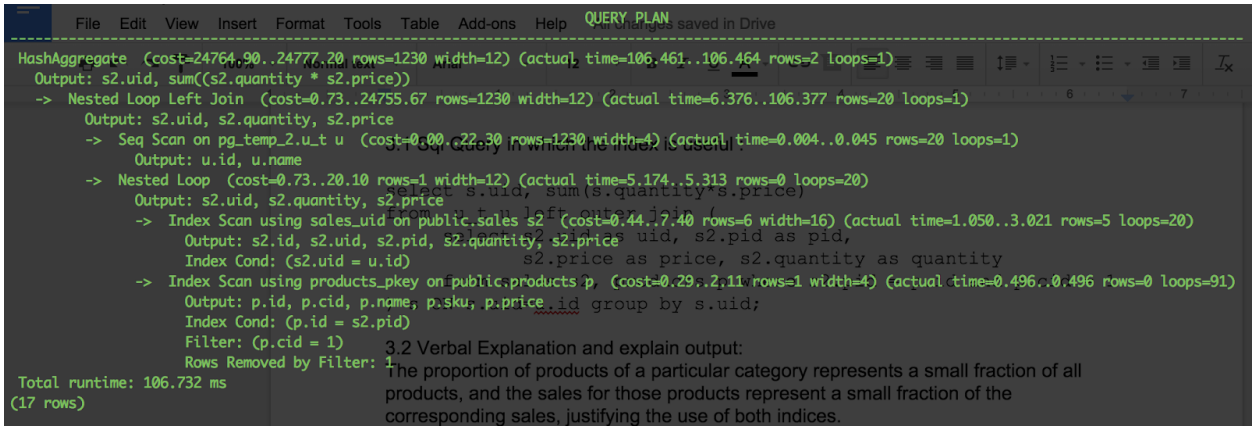
The proportion of products of a particular category represents a small fraction of all products, and the sales for those products represent a small fraction of the corresponding sales, justifying the use of both indices.

Explain Output for small database :



Explain Output for large database :

* : all query plans are available in the `index_report` folder in the solution.



4 (Rejected) User.state :

3.1 Sql Query in which the index was thought to be useful :

```
CREATE TEMP TABLE u_t (id int, name text) ON COMMIT DELETE ROWS;
insert into u_t (id, name) select id,name
from users order by name asc offset 0 limit 20;
s_t.id, coalesce(sum(s.quantity*s.price),0)
from s_t left outer join (
     select u.state as state,
            s.quantity as quantity,
               s.price as price
     from users u JOIN sales s ON s.uid=u.id
) s on s.state=s_t.id group by s_t.id;
```

3.2 Index not being used

User.state could be used in a query that fetches users corresponding to the specified states. There problem is that 20 states represent 40% of users, which is too large a share of users to make the user.state ever beneficial, and it was not used by the query plan.

* : all query plans are available in the `index_report` folder in the solution.