Unit 12 Problem Set Submission Form

Overview

Your Name	Hendi Kushta
Your SU Email	hkushta@syr.edu

Instructions

Put your name and SU email at the top. Answer these questions all from the lab. When asked to include screenshots, please follow the screen shot guidelines from the first lab.

Remember as you complete the problem sets it is not only about getting it right / correct. We will discuss the answers in class so it's important to articulate anything you would like to contribute to the discussion in your answer:

- If you feel the question is vague, include any assumptions you've made.
- If you feel the answer requires interpretation or justification provide it.
- If you do not know the answer to the question, articulate what you tried and how you are stuck.

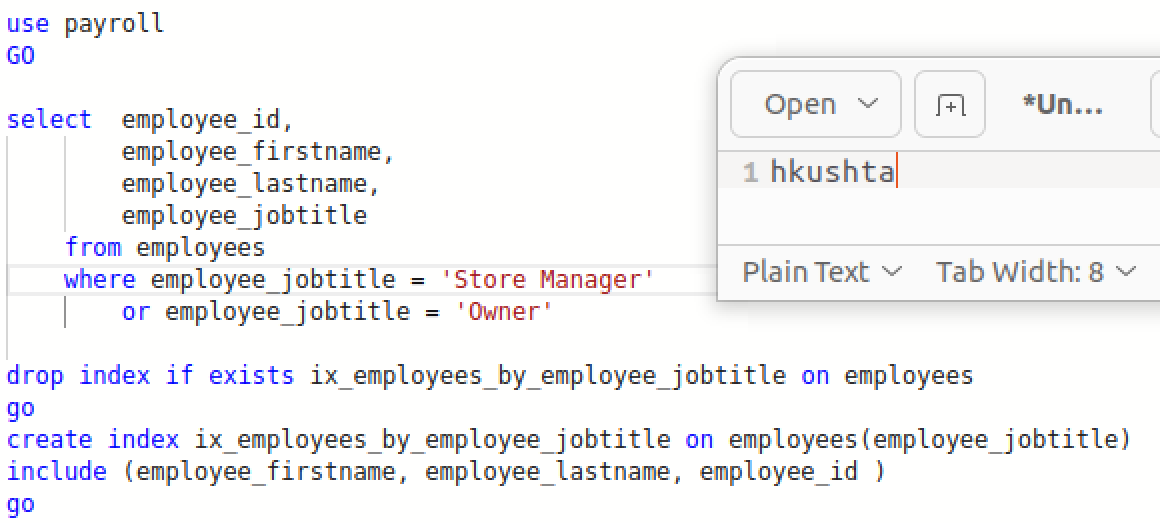
This how you receive credit for answering questions which might not be correct.

Questions

Answer these questions using the problem set submission template. You will need to consult the logical model in the overview section for details. For any screenshots provided, please follow the guidelines for submitting a screenshot.

Write the following as SQL programs. For each, include the SQL as a screenshot with the output of the SQL Code.

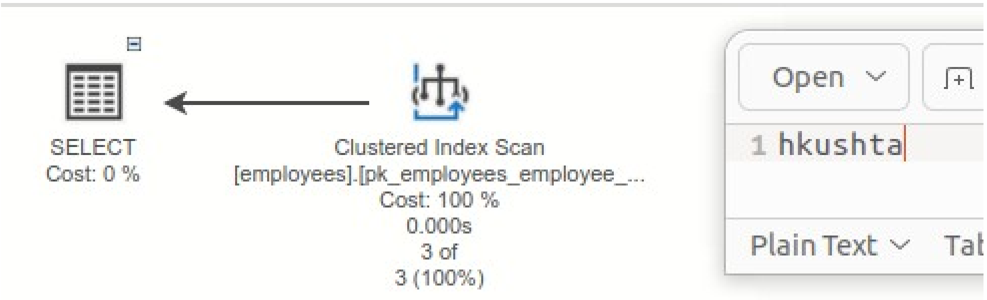
1. Using the **payroll** database write an index to improve the performance of the following query. Your screenshot should include the created index SQL code and the query plan demonstrating the index is being used.



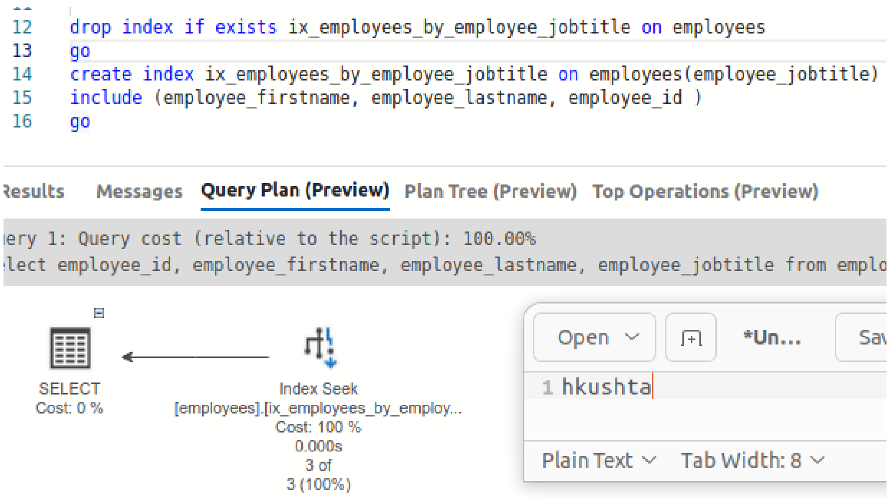
```
use payroll
GO

select  employee_id,
        employee_firstname,
        employee_lastname,
        employee_jobtitle
    from employees
    where employee_jobtitle = 'Store Manager'
        or employee_jobtitle = 'Owner'

drop index if exists ix_employees_by_employee_jobtitle on employees
go
create index ix_employees_by_employee_jobtitle on employees(employee_jobtitle)
include (employee_firstname, employee_lastname, employee_id )
go
```

2. Write another query using GROUP BY which also uses the index you created in the first question.

   Before the index was used, the plan is



   After creating the index as we see from the screen shot below, the second step has changed from scan to seek:



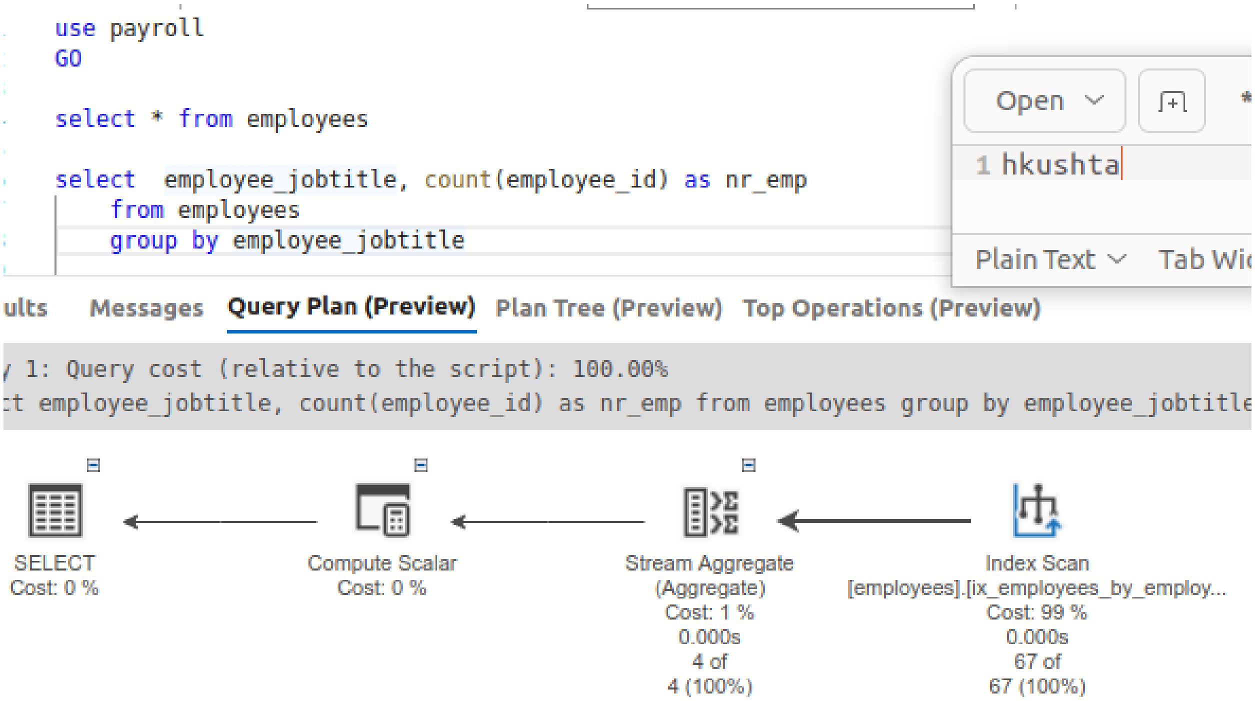
```
drop index if exists ix_employees_by_employee_jobtitle on employees
go
create index ix_employees_by_employee_jobtitle on employees(employee_jobtitle)
include (employee_firstname, employee_lastname, employee_id )
go
```

```
ery 1: Query cost (relative to the script): 100.00%
lect employee_id, employee_firstname, employee_lastname, employee_jobtitle from emplo
```

3. For the following query from a previous assignment, which provides a rank of each bid on an item:

```
select item_id, item_name,
    dense_rank() over
        ( partition by item_name order by bid_datetime) as bid_order,
    bid_amount,
    lag(user_firstname + ' ' + user_lastname) over
        (partition by item_name order by bid_datetime) as prev_bidder,
    user_firstname + ' ' + user_lastname as bidder,
    lead(user_firstname + ' ' + user_lastname) over
        (partition by item_name order by bid_datetime) as next_bidder
    from vb_items
        join vb_bids on item_id=bid_item_id
        join vb_users on bid_user_id = user_id
    where bid_status='ok'
```

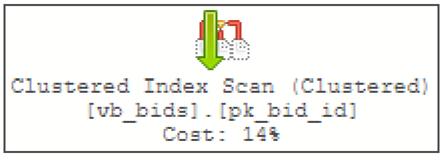
implement the query and run it. Provide a screenshot of the query plan and include the portion where the **vb_bids**, **vb_items,** and **vb_users** tables are selected and joined together.



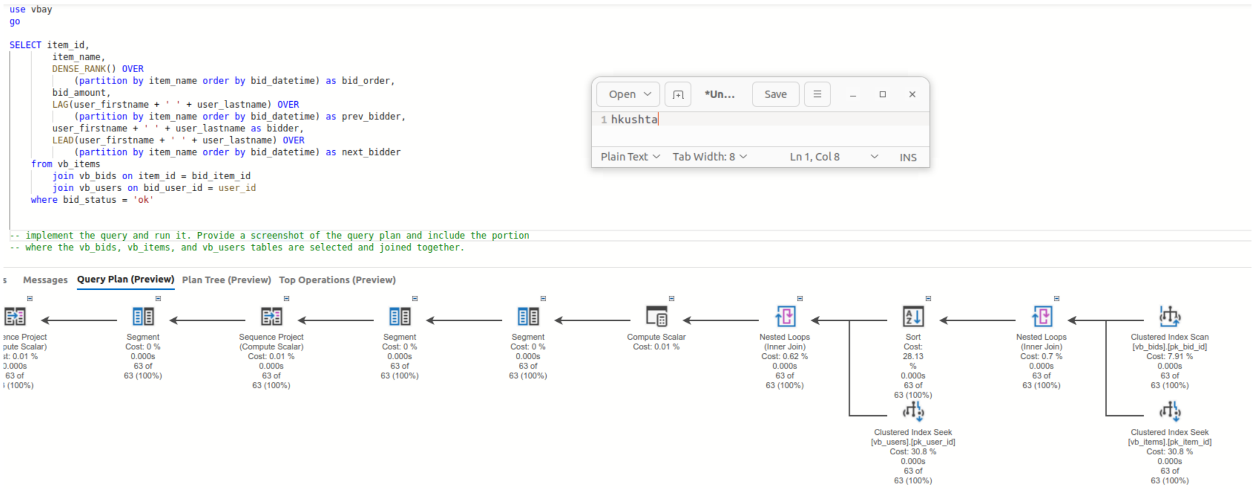
```
use payroll
GO

select * from employees

select  employee_jobtitle, count(employee_id) as nr_emp
   from employees
   group by employee_jobtitle
```

```
/ 1: Query cost (relative to the script): 100.00%
ct employee_jobtitle, count(employee_id) as nr_emp from employees group by employee_jobtitle
```

4. Write an index to improve performance of the query by replacing the clustered index scan on **vb_bids**



with an index seek on the same table. Provide a screenshot of your index code and a screenshot of the query plan demonstrating the index is being used to draw data into the query.



```
use vbay
go

SELECT item_id,
        item_name,
        DENSE_RANK() OVER
            (partition by item_name order by bid_datetime) as bid_order,
        bid_amount,
        LAG(user_firstname + ' ' + user_lastname) OVER
            (partition by item_name order by bid_datetime) as prev_bidder,
        user_firstname + ' ' + user_lastname as bidder,
        LEAD(user_firstname + ' ' + user_lastname) OVER
            (partition by item_name order by bid_datetime) as next_bidder
    from vb_items
        join vb_bids on item_id = bid_item_id
        join vb_users on bid_user_id = user_id
    where bid_status = 'ok'


-- implement the query and run it. Provide a screenshot of the query plan and include the portion
-- where the vb_bids, vb_items, and vb_users tables are selected and joined together.
```

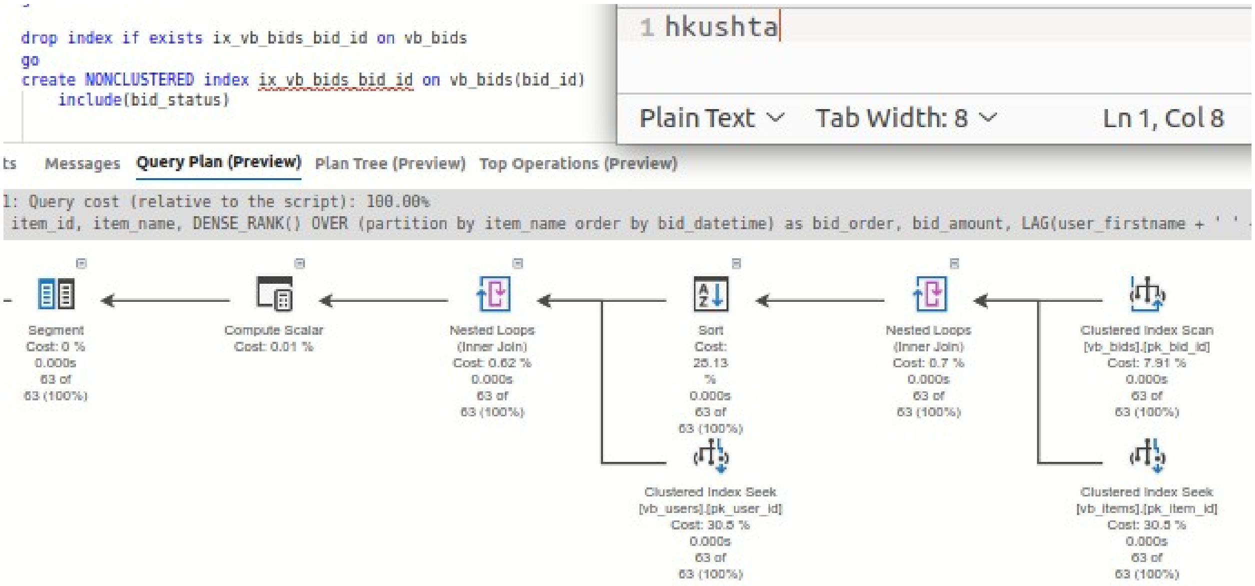
```
hkushta
```

5. Using **fudgemart_v3**, create a schemabound view from the following query:

```
select  c.customer_state, c.customer_firstname + ' ' + c.customer_lastname as customer_name,
datepart(year,order_date) as order_year, o.order_id, o.ship_via,
od.order_qty as order_detail_qty, od.order_qty * p.product_retail_price as order_detail_extd_price,
p.product_id, p.product_name, p.product_department
    from dbo.fm_orders o
    join dbo.fm_customers c on o.customer_id = c.customer_id
    join dbo.fm_order_details od on o.order_id = od.order_id
    join dbo.fm_products p on p.product_id = od.product_id
```

Name the view **v_orders** . Provide a screenshot of the code and sample output which conveys the query ran and created the view.

Tried so many different ways, couldn’t replace clustered index scan on vb_bids with an index seek on the same table.

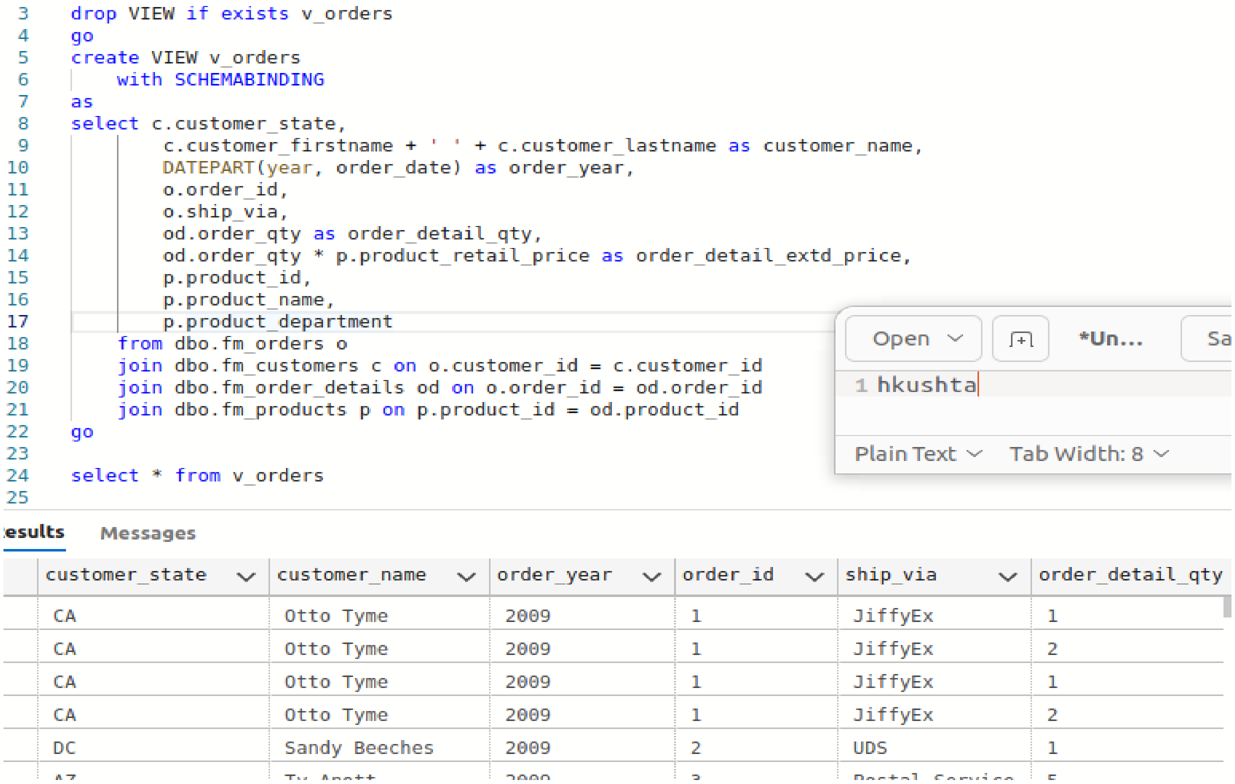


```
drop index if exists ix_vb_bids_bid_id on vb_bids
go
create NONCLUSTERED index ix_vb_bids_bid_id on vb_bids(bid_id)
    include(bid_status)
```

```
1 hkushta
```

```
1: Query cost (relative to the script): 100.00%
 item_id, item_name, DENSE_RANK() OVER (partition by item_name order by bid_datetime) as bid_order, bid_amount, LAG(user_firstname + ' ' -
```

6. Write code to add a unique clustered index to the view **v_orders**. Execute your view (**select * from v_orders)** and then observe the query plan to see if the index is being used. If the index is not being used, that's an indication there is not enough data to warrant the index. You can force the index to be used by using the **noexpand** option on the query: **select * from v_orders with (noexpand)** Provide a screenshot of code to create the index and execute the view along with the query plan showing the index is used.



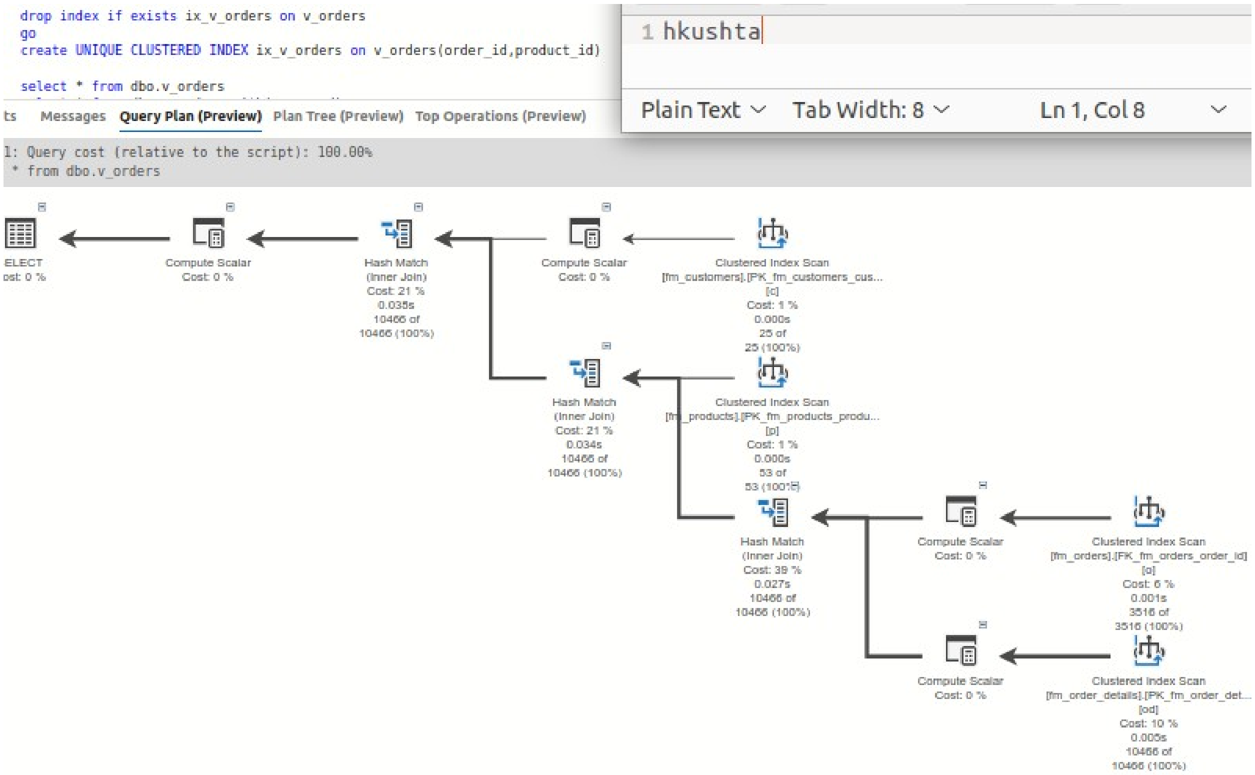
```
drop VIEW if exists v_orders
go
create VIEW v_orders
    with SCHEMABINDING
as
select c.customer_state,
        c.customer_firstname + ' ' + c.customer_lastname as customer_name,
        DATEPART(year, order_date) as order_year,
        o.order_id,
        o.ship_via,
        od.order_qty as order_detail_qty,
        od.order_qty * p.product_retail_price as order_detail_extd_price,
        p.product_id,
        p.product_name,
        p.product_department
    from dbo.fm_orders o
    join dbo.fm_customers c on o.customer_id = c.customer_id
    join dbo.fm_order_details od on o.order_id = od.order_id
    join dbo.fm_products p on p.product_id = od.product_id
go

select * from v_orders
```

```
1 hkushta
```

customer_state	customer_name	order_year	order_id	ship_via	order_detail_qty
CA	Otto Tyme	2009	1	JiffyEx	1
CA	Otto Tyme	2009	1	JiffyEx	2
CA	Otto Tyme	2009	1	JiffyEx	1
CA	Otto Tyme	2009	1	JiffyEx	2
DC	Sandy Beeches	2009	2	UDS	1

To create a unique clustered indexed, I have used order_id and product_id since none of the attributes in the view was unique.

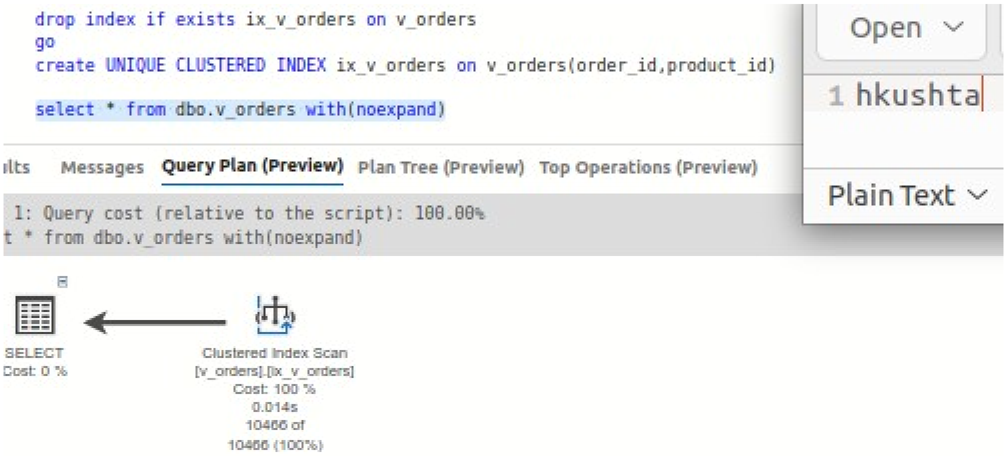
When I run **select * from v_orders,** the index is not being used as shown in the screen shot below.



```
drop index if exists ix_v_orders on v_orders
go
create UNIQUE CLUSTERED INDEX ix_v_orders on v_orders(order_id,product_id)

select * from dbo.v_orders
```

```
1: Query cost (relative to the script): 100.00%
 * from dbo.v_orders
```

When I run **select * from v_orders with (noexpand),** I see that the index has been created as shown in the screen shot below.

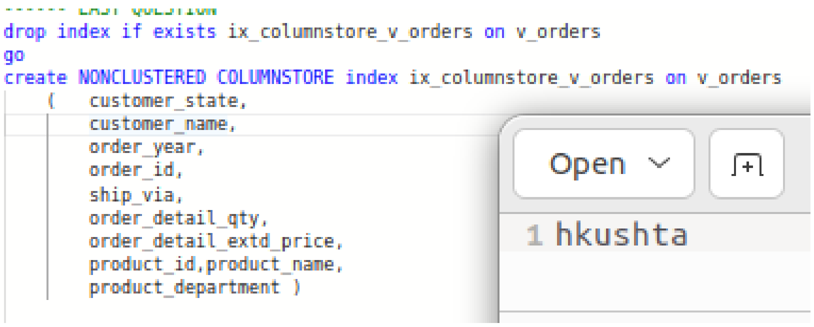


```
drop index if exists ix_v_orders on v_orders
go
create UNIQUE CLUSTERED INDEX ix_v_orders on v_orders(order_id,product_id)

select * from dbo.v_orders with(noexpand)
```

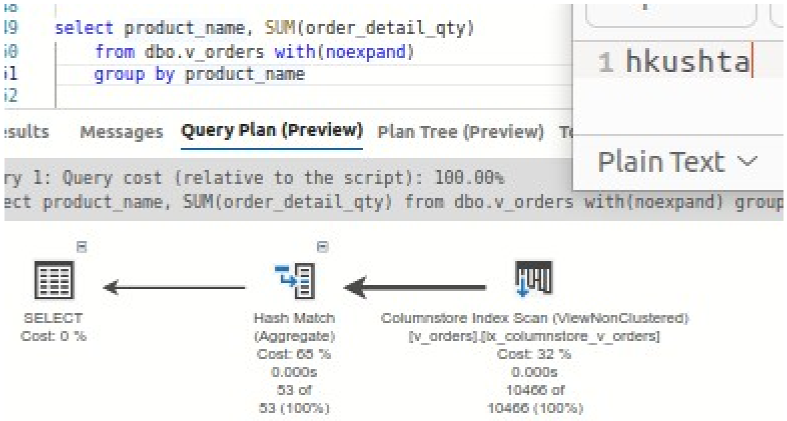
```
1: Query cost (relative to the script): 100.00%
t * from dbo.v_orders with(noexpand)
```

7. Write code to add a columnstore index to **v_orders** include all the columns from the view in the column store index. Provide screenshots with code to demonstrate you created the columnstore index and that these queries use it:

```
select product_name, sum(order_detail_qty)
    from v_orders with (noexpand)
    group by product_name

select distinct customer_name, product_department
    from v_orders with (noexpand)
```

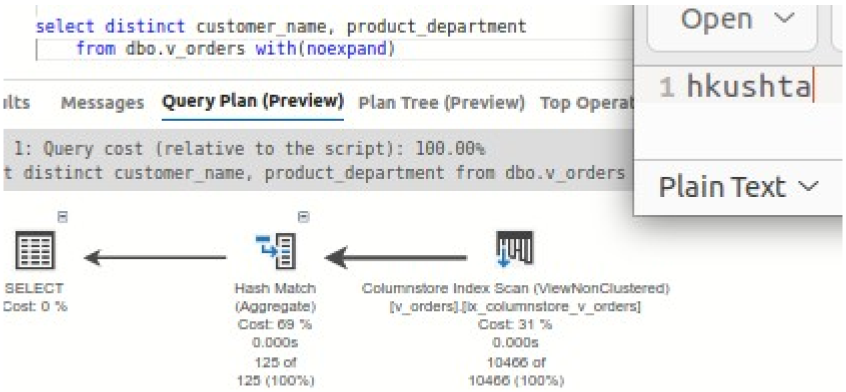


```
------ LAST QUESTION
drop index if exists ix_columnstore_v_orders on v_orders
go
create NONCLUSTERED COLUMNSTORE index ix_columnstore_v_orders on v_orders
    (   customer_state,
        customer_name,
        order_year,
        order_id,
        ship_via,
        order_detail_qty,
        order_detail_extd_price,
        product_id,product_name,
        product_department )
```



```
select product_name, SUM(order_detail_qty)
    from dbo.v_orders with(noexpand)
    group by product_name
```

```
ry 1: Query cost (relative to the script): 100.00%
ect product_name, SUM(order_detail_qty) from dbo.v_orders with(noexpand) group
```



```
select distinct customer_name, product_department
    from dbo.v_orders with(noexpand)
```

```
1: Query cost (relative to the script): 100.00%
t distinct customer_name, product_department from dbo.v_orders
```

Reflection

Use this section to reflect on your learning. To achieve the highest grade on the assignment you must be as descriptive and personal as possible with your reflection.

1. What are the key things you learned through the process of completing this assignment?

   Create different types of indexes

2. What were the challenges or roadblocks (if any) you encountered on the way to completing it?

   Question 5

3. Were you prepared for this assignment? What can you do to be better prepared?

   Yes, I was

4. Now that you have completed the assignment rate your comfort level with this week's material. This should be an honest assessment: (choose one)

   4 ==> I understand this material and can explain it to others.
   3 ==> I understand this material.
   2 ==> I somewhat understand the material but sometimes need guidance from others.
   1 ==> I understand very little of this material and need extra help.