IST 659- Week 12

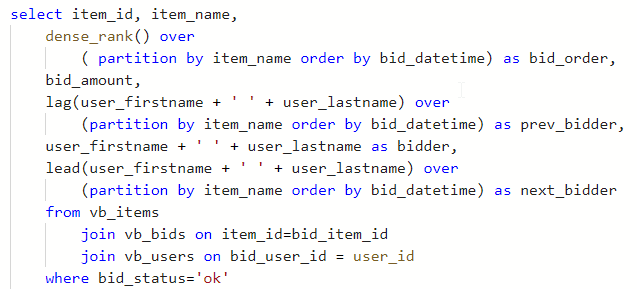
By Shivani Sanjay Mahaddalkar

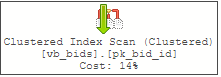
# Questions

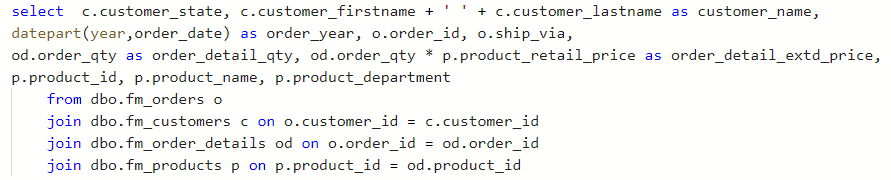
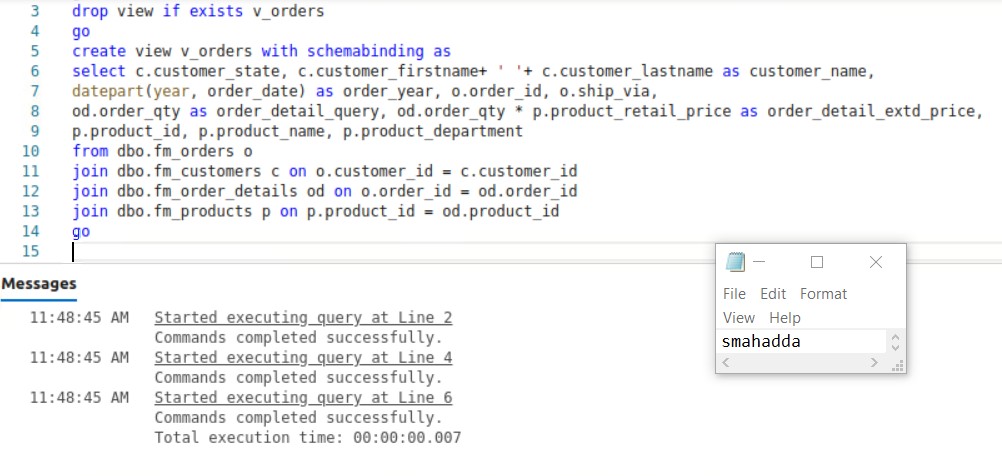
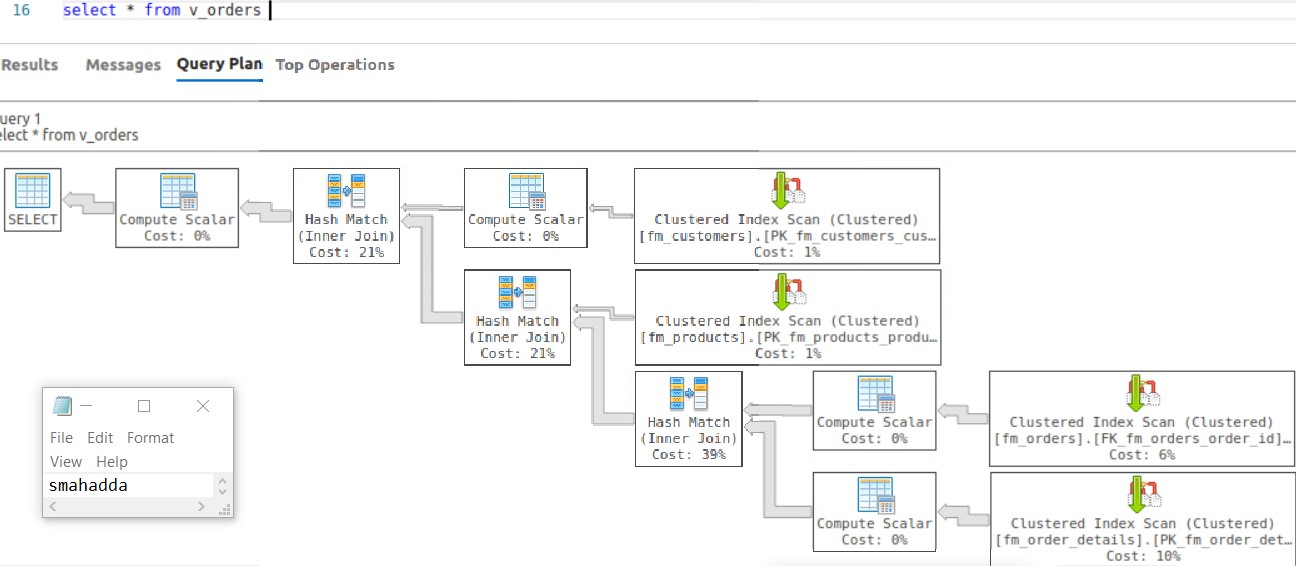
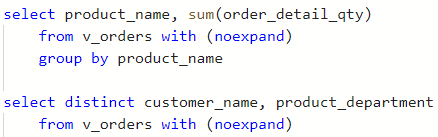
Write the following as SQL programs. For each, include the SQL as a screenshot with the output of the SQL Code.

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.  
   Graphical user interface, text, application, email

   Description automatically generated
2. Write another query using GROUP BY which also uses the index you created in the first question.  
   Graphical user interface, text, application

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

   Description automatically generated
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.  
   I was able to improve the performance, but I could not replace the scan with seek.  
   A picture containing application

   Description automatically generated
5. Using **fudgemart\_v3**, create a schemabound view from the following query:   
     
   Name the view **v\_orders** . Provide a screenshot of the code and sample output which conveys the query ran and created the view.  
   
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.   
   I am unable to understand how queries with join be indexed.  
   
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:  
   

I do not understand how to implement the columnstore index.