IST769 Homework Submission

Name: **Sathish Kumar Rajendiran**
SUID: **666555028**
Email: srajendi@syr.edu
Date Due: **08/03/2021**
Date Due: **Performance, Security, NoSQL**
Homework #:**4**

Exercise(s):

1. **Create a non-clustered index on the timesheets table in the demo database. The index you create should be designed to improve the following query:**

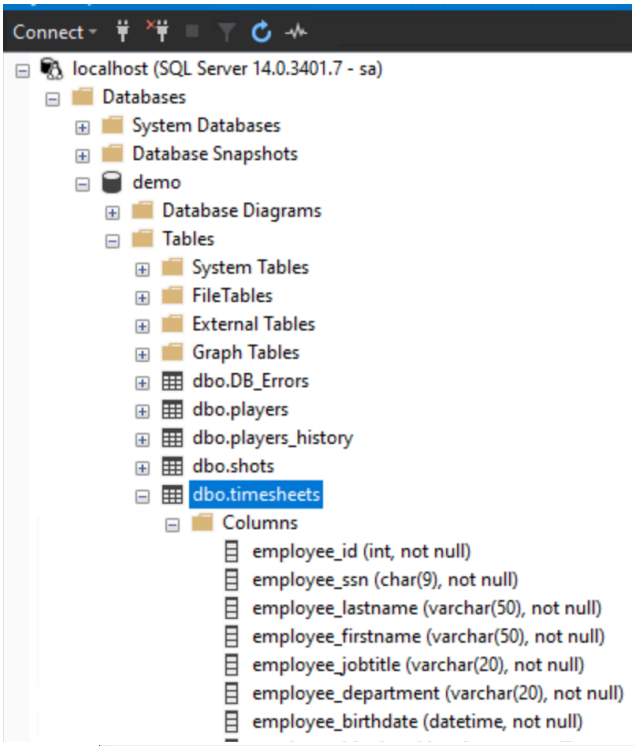
```
select employee_id, employee_firstname, employee_lastname,
sum(timesheet_hourlyrate*timesheet_hours)
from timesheets
group by employee_id, employee_firstname, employee_lastname
```

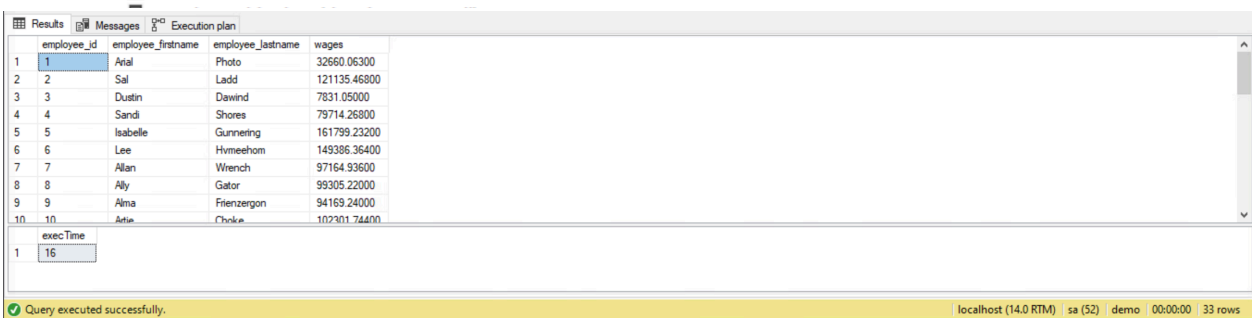
Solution:

```
----Pre-setup command
----Use [fudgemart_v3]
----Go
----select * into demo.dbo.timesheets
----from fudgemart_employees join fudgemart_employee_timesheets
----on employee_id = timesheet_employee_id
----Go

USE [demo]
GO

SET STATISTICS TIME ON
SET STATISTICS IO ON
GO

--before Index creation
declare @startTime datetime2
declare @endTime datetime2
set @startTime = getdate()
select employee_id, employee_firstname, employee_lastname,
sum(timesheet_hourlyrate*timesheet_hours) as wages
from timesheets
group by employee_id, employee_firstname, employee_lastname
set @endTime = getdate()
---total execution time in milliseconds
```

```
select datediff(millisecond,@startTime,@endTime) as execTime
GO

SET STATISTICS TIME OFF
SET STATISTICS IO OFF
GO

---- Find an existing index named IX_Timesheets_EmployeeID_Name_Hours_Rate and delete
it if found.
IF EXISTS (SELECT name FROM sys.indexes
        WHERE name = N'IX_Timesheets_EmployeeID_Name_Hours_Rate')
DROP INDEX IX_Timesheets_EmployeeID_Name_Hours_Rate ON dbo.timesheets;
GO


-- Create a nonclustered index called IX_Timesheets_EmployeeID_Name_Hours_Rate
-- on the dbo.timesheets table using the employee_id,
employee_firstname,employee_lastname,timesheet_hourlyrate,timesheet_hours columns.
CREATE NONCLUSTERED INDEX IX_Timesheets_EmployeeID_Name_Hours_Rate
ON dbo.timesheets (employee_id)
INCLUDE (employee_firstname, employee_lastname,timesheet_hourlyrate,timesheet_hours)
GO


SET STATISTICS TIME ON
SET STATISTICS IO ON
GO

--after Index creation
declare @startTime datetime2
declare @endTime datetime2
set @startTime = getdate()
select employee_id, employee_firstname, employee_lastname,
sum(timesheet_hourlyrate*timesheet_hours) as wages
from timesheets
group by employee_id, employee_firstname, employee_lastname
set @endTime = getdate()
---total execution time in milliseconds
select datediff(millisecond,@startTime,@endTime) as execTime
GO

SET STATISTICS TIME OFF
SET STATISTICS IO OFF
GO
```

Evidence:



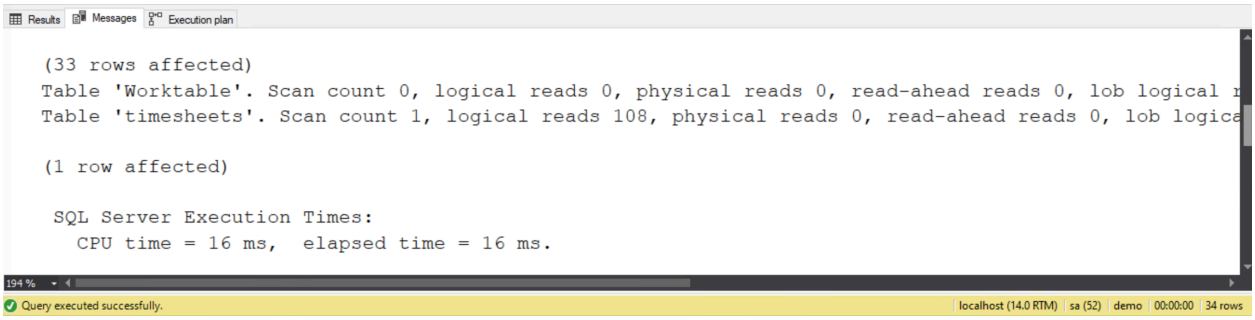


Results | Messages | Execution plan

	employee_id	employee_firstname	employee_lastname	wages
1	1	Arial	Photo	32660.06300
2	2	Sal	Ladd	121135.46800
3	3	Dustin	Dawind	7831.05000
4	4	Sandi	Shores	79714.26800
5	5	Isabelle	Gunnering	161799.23200
6	6	Lee	Hvmeehom	149386.36400
7	7	Allan	Wrench	97164.93600
8	8	Ally	Gator	99305.22000
9	9	Alma	Frienzergon	94169.24000
10	10	Artie	Choke	102301.74400

	execTime
1	16

Query executed successfully. | localhost (14.0 RTM) | sa (52) | demo | 00:00:00 | 33 rows

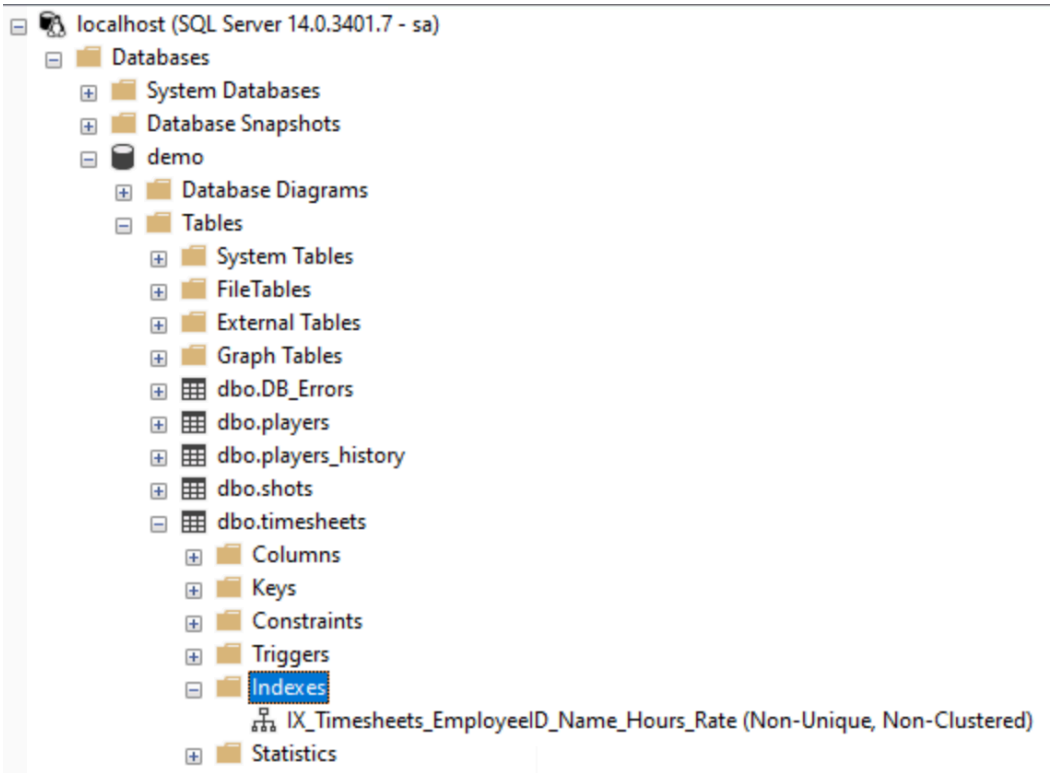


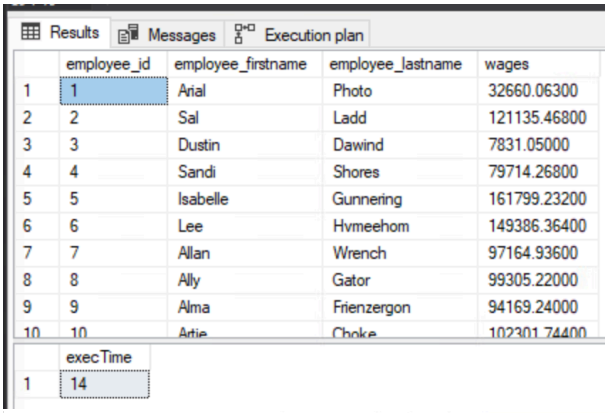
Results | Messages | Execution plan

```
(33 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical r
Table 'timesheets'. Scan count 1, logical reads 108, physical reads 0, read-ahead reads 0, lob logica

(1 row affected)

 SQL Server Execution Times:
   CPU time = 16 ms,  elapsed time = 16 ms.
```

194 %

Query executed successfully. | localhost (14.0 RTM) | sa (52) | demo | 00:00:00 | 34 rows

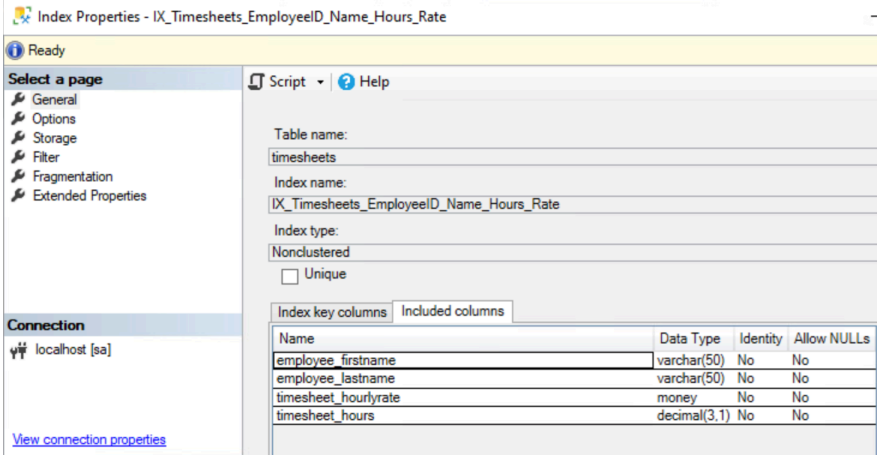


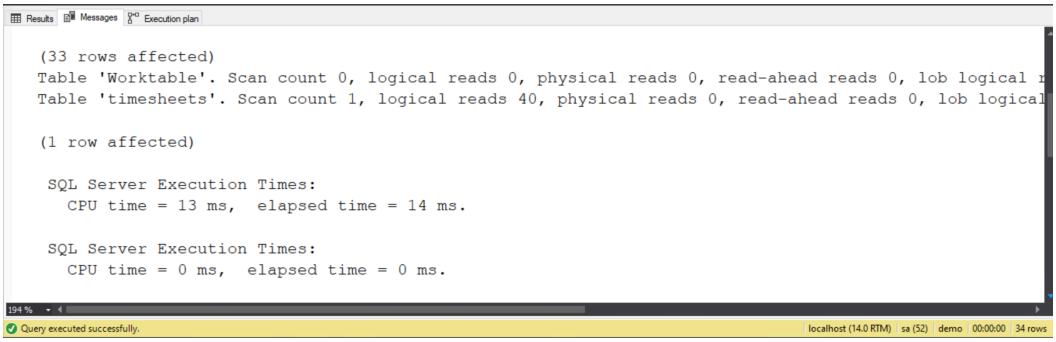


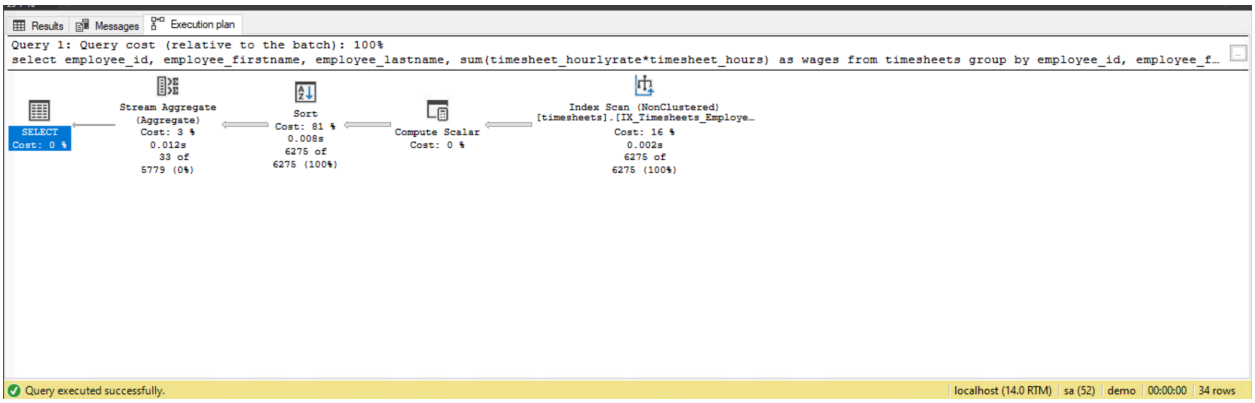
Results | Messages | Execution plan

	employee_id	employee_firstname	employee_lastname	wages
1	1	Arial	Photo	32660.06300
2	2	Sal	Ladd	121135.46800
3	3	Dustin	Dawind	7831.05000
4	4	Sandi	Shores	79714.26800
5	5	Isabelle	Gunnering	161799.23200
6	6	Lee	Hvmeehom	149386.36400
7	7	Allan	Wrench	97164.93600
8	8	Ally	Gator	99305.22000
9	9	Alma	Frienzergon	94169.24000
10	10	Artie	Choke	102301.74400

	execTime
1	14







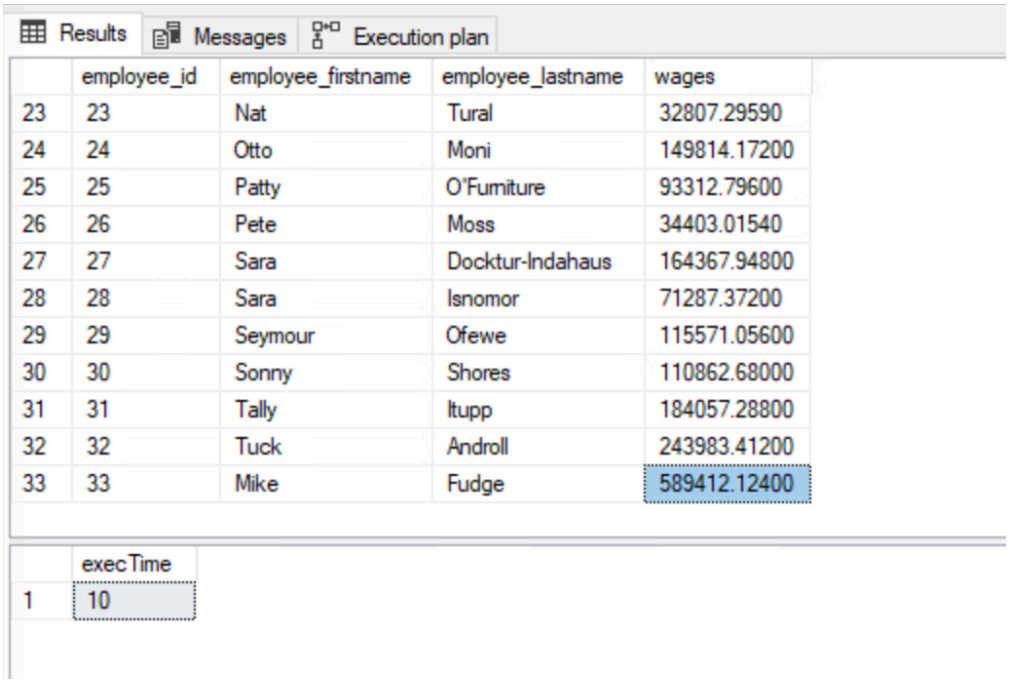
2. **Write an SQL Select query which uses the index you created in the first question but does an index seek instead of an index scan.**

Solution:

```
SET STATISTICS TIME ON
SET STATISTICS IO ON
GO

--after Index creation
declare @startTime datetime2
declare @endTime datetime2
--declare @maxemployeeID int
set @startTime = getdate()
--select @maxemployeeID = max(employee_id) from dbo.timesheets
select employee_id, employee_firstname, employee_lastname,
sum(timesheet_hourlyrate*timesheet_hours) as wages
from timesheets
where employee_id < 50
--where employee_id < ( select max(employee_id) from dbo.timesheets)
--where employee_id < @maxemployeeID
group by employee_id, employee_firstname, employee_lastname
```

```
set @endTime = getdate()
---total execution time in milliseconds
select datediff(millisecond,@startTime,@endTime) as execTime
GO

SET STATISTICS TIME OFF
SET STATISTICS IO OFF
GO
```

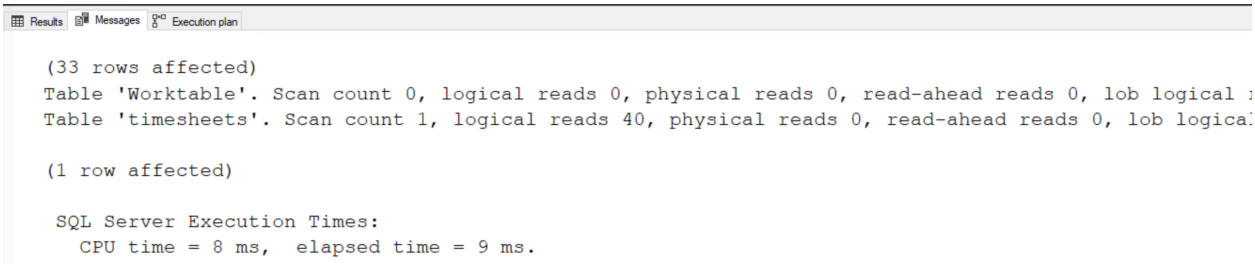
Evidence:



Results | Messages | Execution plan

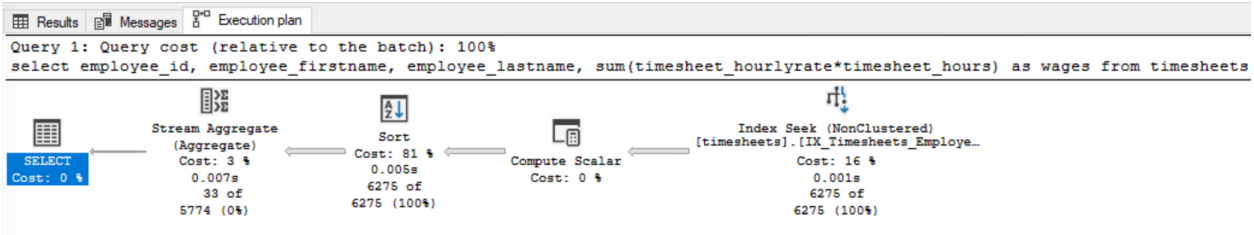
	employee_id	employee_firstname	employee_lastname	wages
23	23	Nat	Tural	32807.29590
24	24	Otto	Moni	149814.17200
25	25	Patty	O'Furniture	93312.79600
26	26	Pete	Moss	34403.01540
27	27	Sara	Docktur-Indahaus	164367.94800
28	28	Sara	Isnomor	71287.37200
29	29	Seymour	Ofewe	115571.05600
30	30	Sonny	Shores	110862.68000
31	31	Tally	Itupp	184057.28800
32	32	Tuck	Androll	243983.41200
33	33	Mike	Fudge	589412.12400

	execTime
1	10



Results | Messages | Execution plan

```
(33 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical
Table 'timesheets'. Scan count 1, logical reads 40, physical reads 0, read-ahead reads 0, lob logica

(1 row affected)

 SQL Server Execution Times:
   CPU time = 8 ms,  elapsed time = 9 ms.
```



3. **Create a single columnstore index on the timesheets table in the demo database which will improve the following queries:**

```
select employee_department, sum(timesheet_hours)
from timesheets group by employee_department

select employee_jobtitle, avg(timesheet_hourlyrate)
from timesheets group by employee_jobtitle
```

Solution:

```
*/
SET STATISTICS TIME ON
SET STATISTICS IO ON
GO

--before Index creation
declare @startTime datetime2
declare @endTime datetime2
set @startTime = getdate()

select employee_department
, sum(timesheet_hours) agg_timesheet_hours
from timesheets
group by employee_department
set @endTime = getdate()

---total execution time in milliseconds
select datediff(millisecond,@startTime,@endTime) as execTime
GO

SET STATISTICS TIME OFF
SET STATISTICS IO OFF
GO

---- Find an existing index named IX_CLM_Timesheets_Employee_Department_Hours
---- and delete it if found.
IF EXISTS (SELECT name FROM sys.indexes
        WHERE name = N'IX_CLM_Timesheets')
DROP INDEX IX_CLM_Timesheets ON dbo.timesheets;
GO
-- Create a nonclustered COLUMNSTORE index called IX_CLM_Timesheets
-- on the dbo.timesheets table using the employee_id,
employee_department,timesheet_hours,timesheet_hourlyrate,timesheet_hours columns.

CREATE NONCLUSTERED COLUMNSTORE INDEX IX_CLM_Timesheets
ON dbo.timesheets(employee_department,timesheet_hours)
GO
```

```
SET STATISTICS TIME ON
SET STATISTICS IO ON
GO

--after Index creation
declare @startTime datetime2
declare @endTime datetime2
set @startTime = getdate()

select employee_department
, sum(timesheet_hours) agg_timesheet_hours
from timesheets
group by employee_department
set @endTime = getdate()

---total execution time in milliseconds
select datediff(millisecond,@startTime,@endTime) as execTime
GO

SET STATISTICS TIME OFF
SET STATISTICS IO OFF
GO

/************************************************************/

SET STATISTICS TIME ON
SET STATISTICS IO ON
GO

--before Index creation
declare @startTime datetime2
declare @endTime datetime2
set @startTime = getdate()

select employee_jobtitle
, avg(timesheet_hourlyrate) avg_hourlyrate
from timesheets
group by employee_jobtitle
set @endTime = getdate()

---total execution time in milliseconds
select datediff(millisecond,@startTime,@endTime) as execTime
GO

SET STATISTICS TIME OFF
SET STATISTICS IO OFF
```

```
GO

IF EXISTS (SELECT name FROM sys.indexes
        WHERE name = N'IX_CLM_Timesheets')
DROP INDEX IX_CLM_Timesheets ON dbo.timesheets;
GO
-- Create a nonclustered COLUMNSTORE index called IX_CLM_Timesheets
-- on the dbo.timesheets table using the employee_id,
employee_department,timesheet_hours,timesheet_hourlyrate,timesheet_hours columns.

CREATE NONCLUSTERED COLUMNSTORE INDEX IX_CLM_Timesheets
ON
dbo.timesheets(employee_department,timesheet_hours,employee_jobtitle,timesheet_hourlyr
ate)
GO

SET STATISTICS TIME ON
SET STATISTICS IO ON
GO

--after Index creation
declare @startTime datetime2
declare @endTime datetime2
set @startTime = getdate()

select employee_jobtitle
, avg(timesheet_hourlyrate) avg_hourlyrate
from timesheets
group by employee_jobtitle
set @endTime = getdate()

---total execution time in milliseconds
select datediff(millisecond,@startTime,@endTime) as execTime
GO

SET STATISTICS TIME OFF
SET STATISTICS IO OFF
GO
```

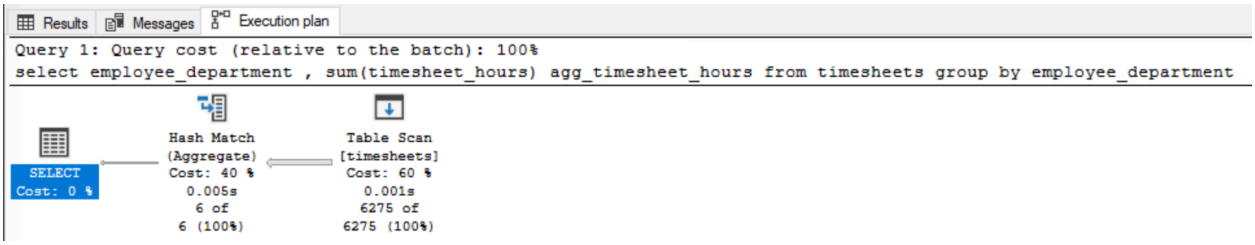
Evidence:

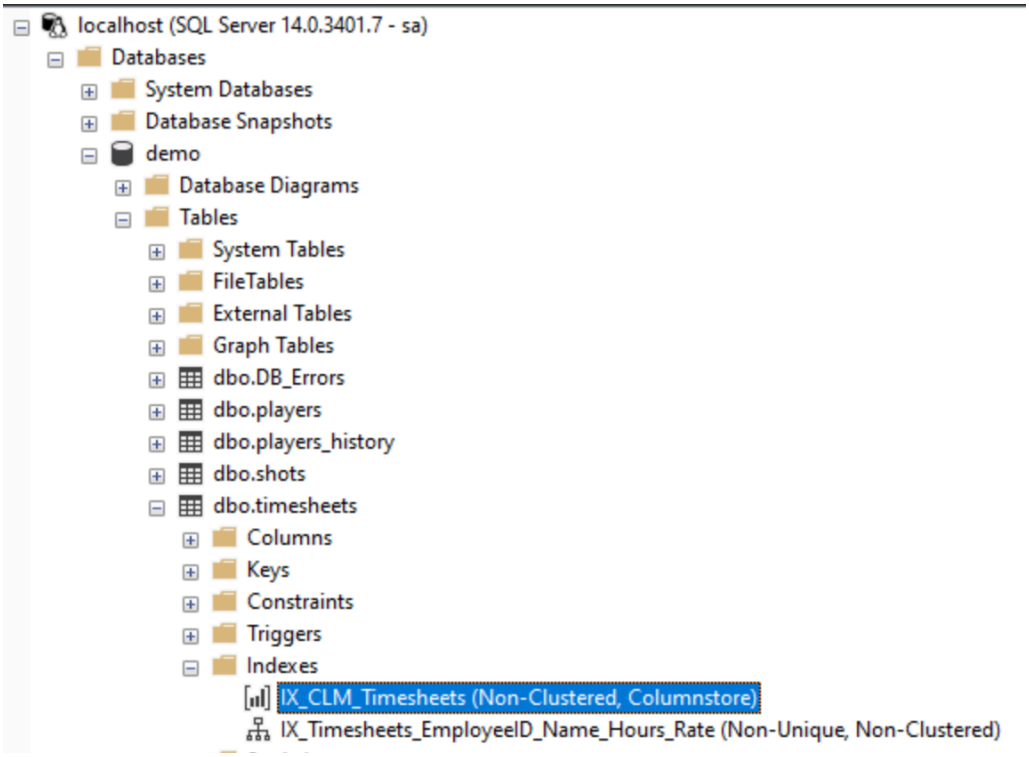
Results | Messages | Execution plan

	employee_department	agg_timesheet_hours
1	Hardware	31829.0
2	Sporting Goods	36455.0
3	Housewares	36618.0
4	Customer Service	41600.0
5	Electronics	35458.0
6	Clothing	30996.0

	execTime
1	4

```
(6 rows affected)
Table 'Worktable'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical r
Table 'Workfile'. Scan count 0, logical reads 0, physical reads 0, read-ahead reads 0, lob logical re
Table 'timesheets'. Scan count 1, logical reads 108, physical reads 0, read-ahead reads 0, lob logica

(1 row affected)

 SQL Server Execution Times:
   CPU time = 5 ms,  elapsed time = 5 ms.
```



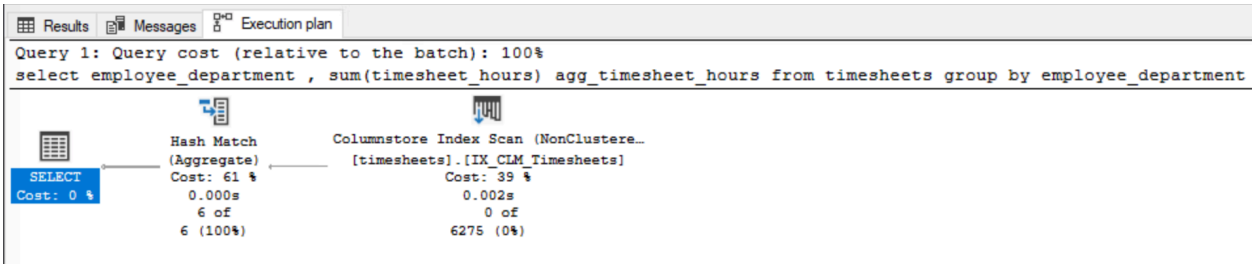


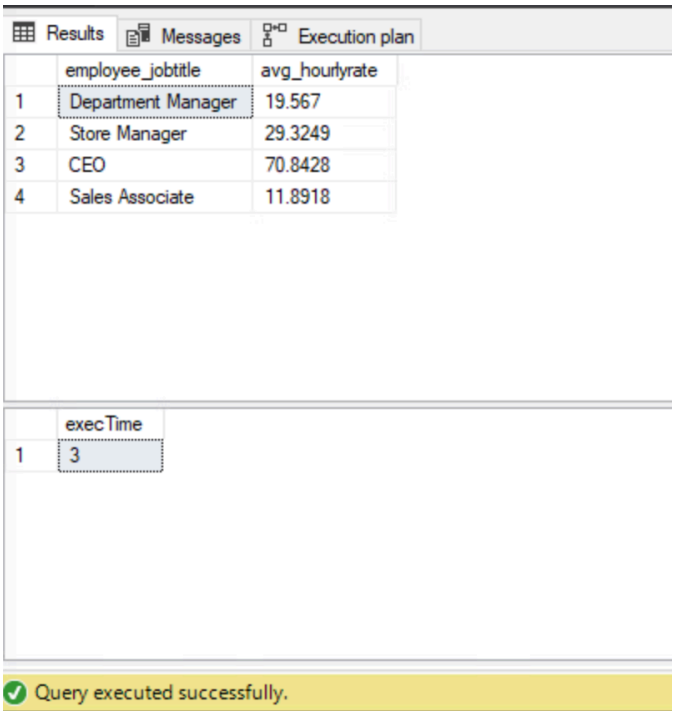
Results | Messages | Execution plan

	employee_department	agg_timesheet_hours
1	Housewares	36618.0
2	Sporting Goods	36455.0
3	Clothing	30996.0
4	Customer Service	41600.0
5	Hardware	31829.0
6	Electronics	35458.0

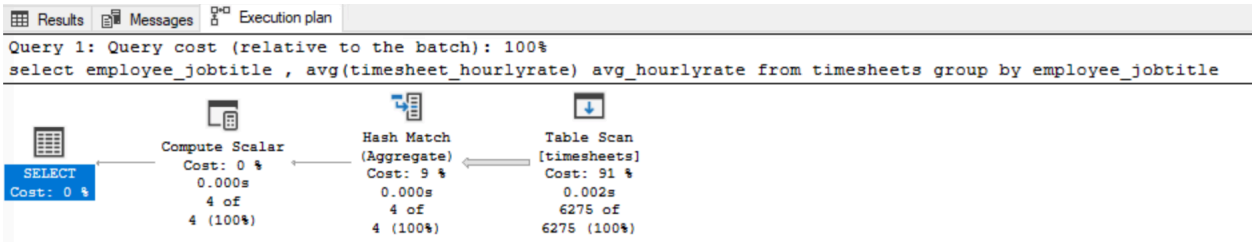
	execTime
1	0

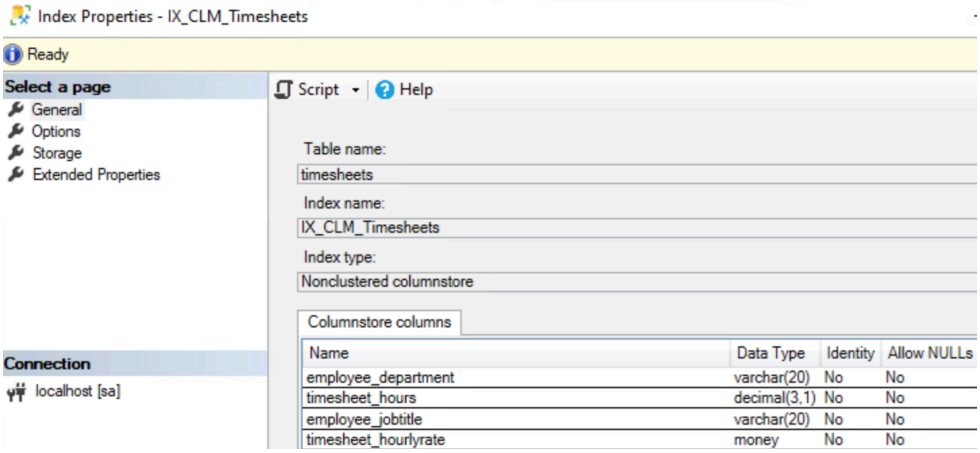
```
(6 rows affected)
Table 'timesheets'. Scan count 2, logical reads 0, physical reads 0, read-ahead reads 0, lob logical
Table 'timesheets'. Segment reads 1, segment skipped 0.

(1 row affected)

 SQL Server Execution Times:
   CPU time = 2 ms,  elapsed time = 1 ms.
```

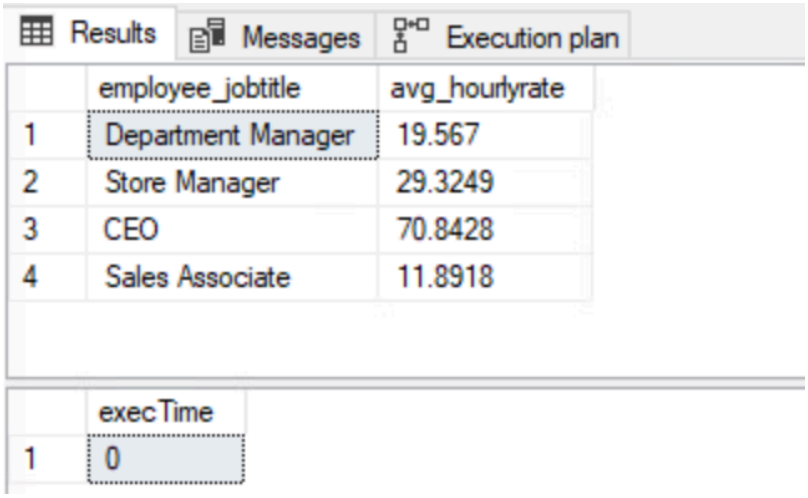




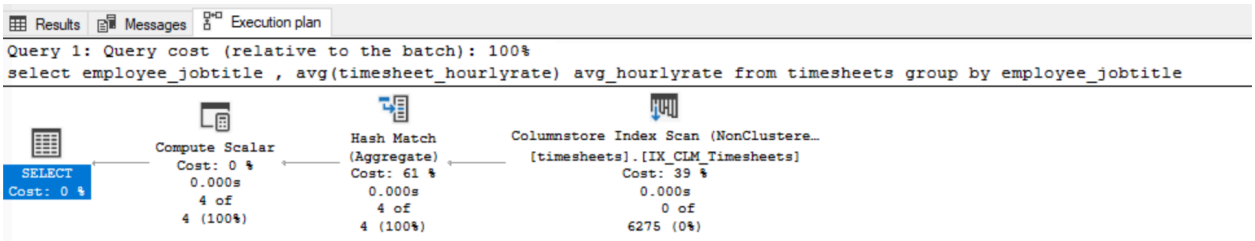

```
 SQL Server Execution Times:
   CPU time = 0 ms,  elapsed time = 0 ms.
SQL Server parse and compile time:
   CPU time = 3 ms, elapsed time = 3 ms.

 SQL Server Execution Times:
   CPU time = 0 ms,  elapsed time = 0 ms.

(4 rows affected)
Table 'timesheets'. Scan count 1, logical reads 108, physical reads 0, read-ahead reads 0, lob logic
```








```
 SQL Server Execution Times:
   CPU time = 0 ms,  elapsed time = 0 ms.
SQL Server parse and compile time:
   CPU time = 0 ms, elapsed time = 0 ms.

 SQL Server Execution Times:
   CPU time = 0 ms,  elapsed time = 0 ms.

(4 rows affected)
Table 'timesheets'. Scan count 2, logical reads 0, physical reads 0, read-ahead reads 0, lob logical
Table 'timesheets'. Segment reads 1, segment skipped 0.
```



4. **Create an indexed view named v_employees on the timesheets table in the demo database which lists the employee id, first name, last name, job title, and department columns values and one row per employee (essentially re-building the employee table). Then set a unique clustered index on the view and finish by writing an SQL Select query which uses the indexed view.**

Solution:

```
IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.VIEWS WHERE TABLE_NAME='v_employees')
BEGIN
DROP VIEW v_employees
END
GO

CREATE VIEW  v_employees WITH schemabinding
AS
SELECT    employee_id
    , employee_firstname
    , employee_lastname
    , employee_jobtitle
    , employee_department
    , COUNT_BIG(*) as employee_count
from dbo.timesheets
Group by
employee_id
, employee_firstname
, employee_lastname
, employee_jobtitle
, employee_department
GO

--before index creation
SELECT * FROM dbo.v_employees
GO


IF EXISTS (SELECT name FROM sys.indexes
        WHERE name ='IX_CLS_v_employees_EmployeeID')
DROP INDEX IX_CLS_v_employees_EmployeeID ON dbo.v_employees;
GO
```

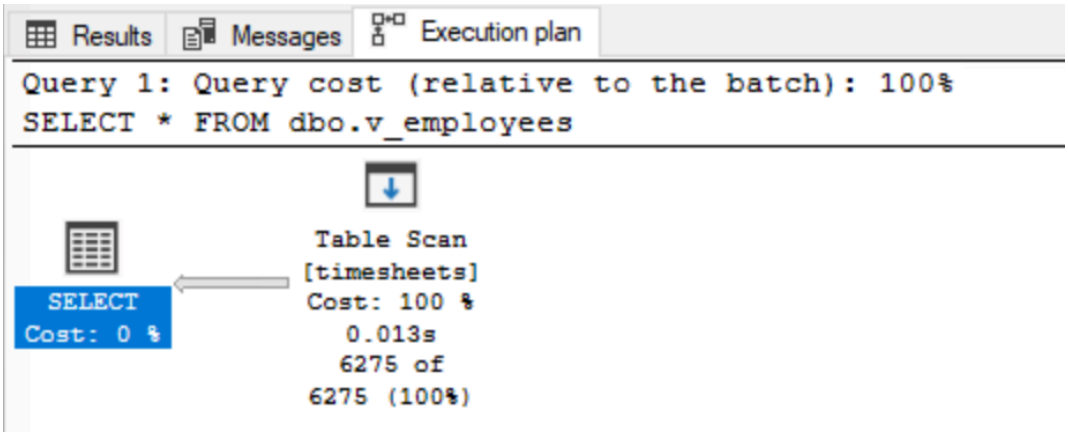
```sql
-- Create an unique clustered index called IX_CLS_v_employees_EmployeeID
-- on the dbo.v_employees table using the employee_id

CREATE UNIQUE CLUSTERED  INDEX IX_CLS_v_employees_EmployeeID
ON dbo.v_employees(employee_id)
GO
--query from the view after index creation
SELECT * FROM dbo.v_employees
GO
```

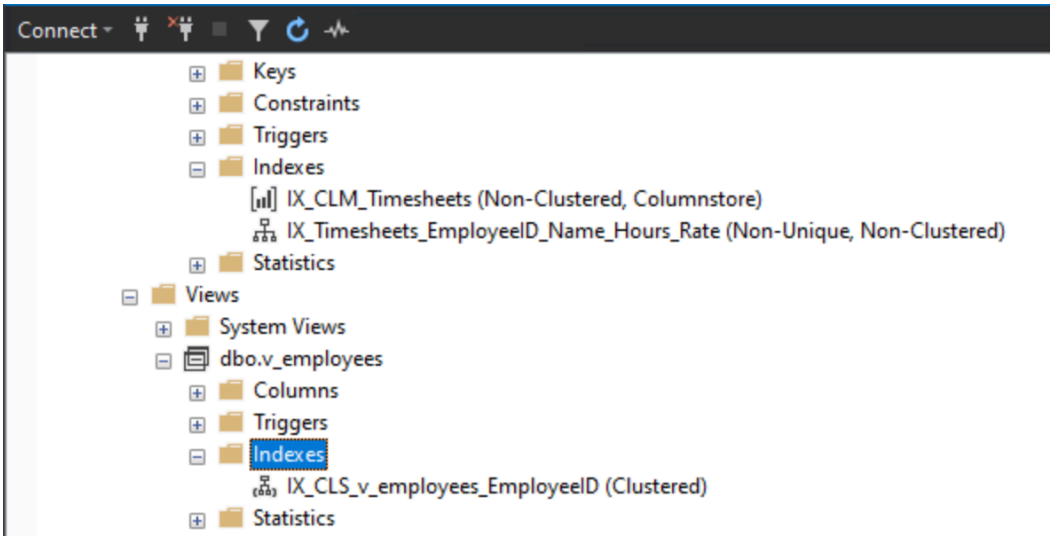
Evidence:

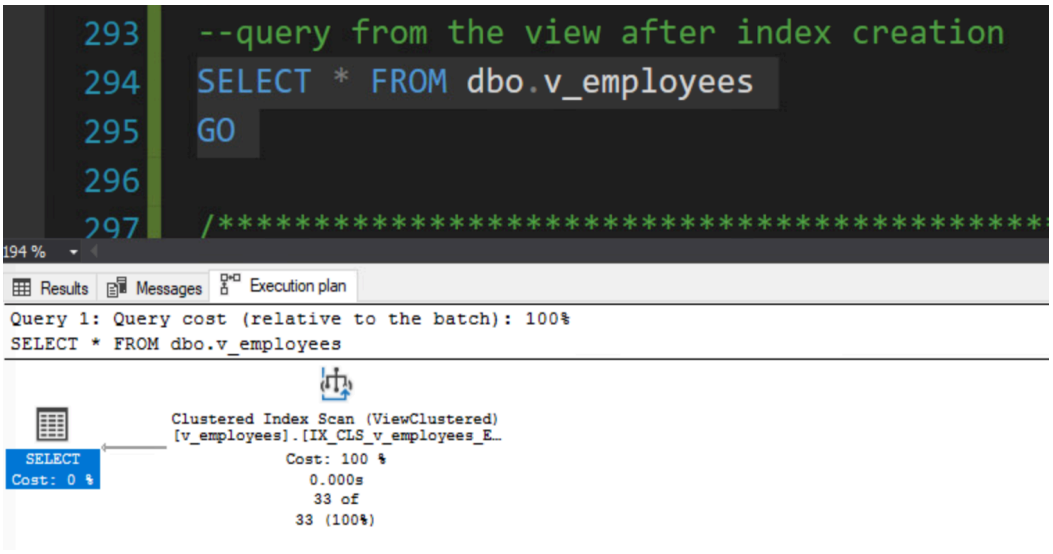
Results | Messages | Execution plan

	employee_id	employee_firstname	employee_lastname	employee_jobtitle	employee_department	employee_count
1	5	Isabelle	Gunnering	Department Manager	Electronics	208
2	12	Bill	Melator	Sales Associate	Sporting Goods	54
3	22	Mike	Rophone	Sales Associate	Housewares	208
4	1	Arial	Photo	Sales Associate	Electronics	129
5	18	Justin	Case	Sales Associate	Clothing	208
6	25	Patty	O'Furniture	Sales Associate	Housewares	208
7	8	Ally	Gator	Sales Associate	Sporting Goods	208
8	17	Kurt	Tan	Sales Associate	Clothing	208
9	29	Seymour	Ofewe	Sales Associate	Customer Service	208
10	30	Sonny	Shores	Sales Associate	Customer Service	208
11	28	Sara	Isnomor	Sales Associate	Clothing	192
12	11	Bette	Alott	Sales Associate	Sporting Goods	208
13	16	Detyers	Erin	Sales Associate	Hardware	208
14	23	Nat	Tural	Sales Associate	Sporting Goods	171
15	2	Sal	Ladd	Sales Associate	Electronics	208
16	20	Mary	Mi	Department Manager	Clothing	208
17	9	Alma	Frienzergon	Sales Associate	Housewares	208
18	14	Chris P.	Nugget	Sales Associate	Electronics	208

Query executed successfully.







	employee_id	employee_firstname	employee_lastname	employee_jobtitle	employee_department	employee_count
1	1	Arial	Photo	Sales Associate	Electronics	129
2	2	Sal	Ladd	Sales Associate	Electronics	208
3	3	Dustin	Dawind	Sales Associate	Hardware	44
4	4	Sandi	Shores	Sales Associate	Hardware	156
5	5	Isabelle	Gunnering	Department Manager	Electronics	208
6	6	Lee	Hvmeehom	Department Manager	Hardware	208
7	7	Allan	Wrench	Sales Associate	Housewares	208
8	8	Ally	Gator	Sales Associate	Sporting Goods	208
9	9	Alma	Frienzergon	Sales Associate	Housewares	208
10	10	Artie	Choke	Sales Associate	Hardware	208
11	11	Bette	Alott	Sales Associate	Sporting Goods	208
12	12	Bill	Melator	Sales Associate	Sporting Goods	54
13	13	Bob	Enweave	Sales Associate	Sporting Goods	121
14	14	Chris P.	Nugget	Sales Associate	Electronics	208
15	15	Chuck	Itupp	Sales Associate	Sporting Goods	208
16	16	Detyers	Erin	Sales Associate	Hardware	208
17	17	Kurt	Tan	Sales Associate	Clothing	208
18	18	Justin	Case	Sales Associate	Clothing	208

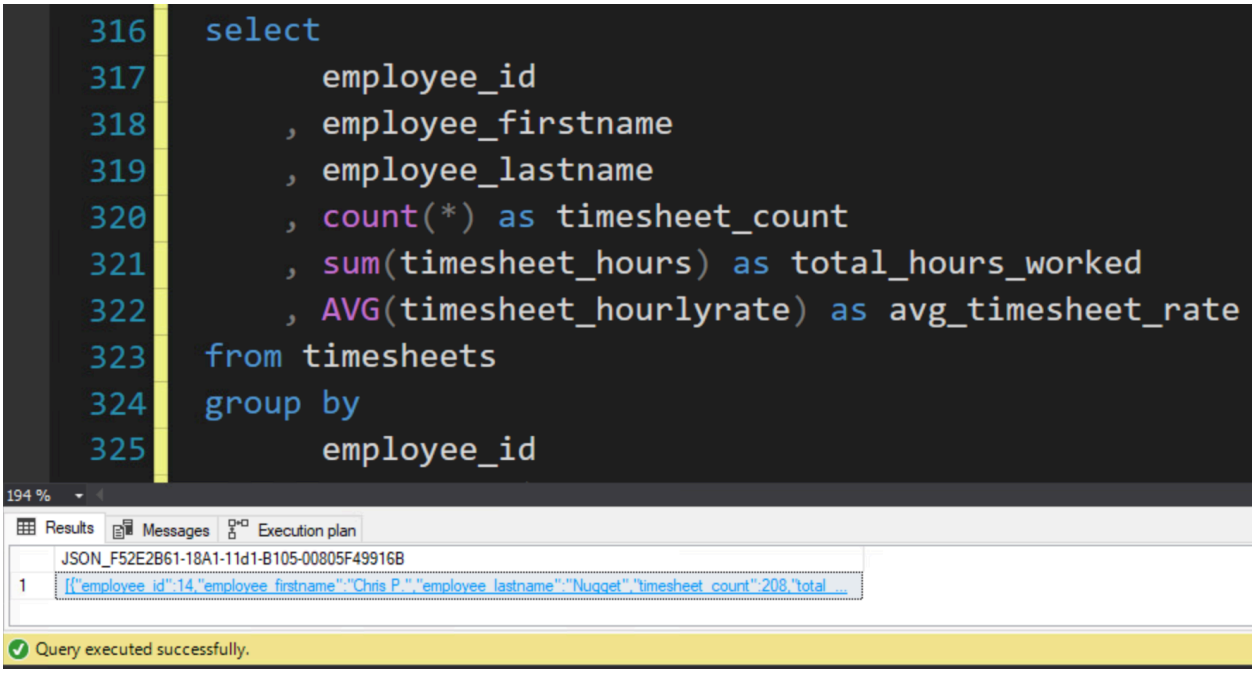
Query executed successfully. localhost (14.0 RTM) | sa (52) | demo | 00:00:00 | 33 rows

5. **Output the following query in JSON format: Display the employee id, first name, last name, count of timesheets, total hours worked, and average timesheet hourly rate.**

Solution:

```
--SQL statement with NOSQL Output
select
      employee_id
    , employee_firstname , employee_lastname
    , count(*) as timesheet_count
    , sum(timesheet_hours) as total_hours_worked
    , AVG(timesheet_hourlyrate) as avg_timesheet_rate
from timesheets
group by
      employee_id
    , employee_firstname
    , employee_lastname
FOR JSON AUTO
GO
```

Evidence:



Results | Messages | Execution plan

	employee_id	employee_firstname	employee_lastname	timesheet_count	total_hours_worked	avg_timesheet_rate
1	14	Chris P.	Nugget	208	8320.0	15.1769
2	22	Mike	Rophone	208	3338.0	11.3184
3	24	Otto	Moni	208	8320.0	18.0065
4	30	Sonny	Shores	208	8320.0	13.3248
5	1	Arial	Photo	129	2178.0	14.9883
6	3	Dustin	Dawind	44	629.0	12.45
7	12	Bill	Melator	54	2160.0	10.2539
8	2	Sal	Ladd	208	8320.0	14.5595

Query executed successfully.

```
325      employee_id
326    , employee_firstname
327    , employee_lastname
328  FOR JSON AUTO
329  GO
330
331
```

194 %

Results | Messages | Execution plan

	JSON_F52E2B61-18A1-11d1-B105-00805F49916B
1	[{"employee_id":14,"employee_firstname":"Chris P...

{JSON formatter}

JSON PARSER | XML FORMATTER | JSBEAUTIFIER | SAVE | RECENT LINKS | LOGIN

Sample

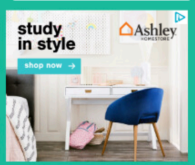

```
:"Tan","timesheet_count":208,"total_hours_worked"
:3338.0,"avg_timesheet_rate":9.5177},{"employee_id"
:27,"employee_firstname":"Sara","employee_lastname"
:"Docktur-Indahaus","timesheet_count":208
,"total_hours_worked":8320.0,"avg_timesheet_rate":19
.7557},{"employee_id":28,"employee_firstname":"Sara"
,"employee_lastname":"Isnomor","timesheet_count":192
,"total_hours_worked":7680.0,"avg_timesheet_rate":9
.2822},{"employee_id":25,"employee_firstname"
:"Patty","employee_lastname":"O'Furniture"
,"timesheet_count":208,"total_hours_worked":8320.0
,"avg_timesheet_rate":11.2154},{"employee_id":9
,"employee_firstname":"Alma","employee_lastname"
:"Frienzergon","timesheet_count":208
,"total_hours_worked":8320.0,"avg_timesheet_rate":11
.3184},{"employee_id":18,"employee_firstname"
:"Justin","employee_lastname":"Case"
,"timesheet_count":208,"total_hours_worked":3338.0
,"avg_timesheet_rate":9.1575},{"employee_id":31
,"employee_firstname":"Tally","employee_lastname"
:"Itupp","timesheet_count":208,"total_hours_worked"
:8320.0,"avg_timesheet_rate":22.1222},{"employee_id"
```


Upload Data

Validate

2 Tab Space

Format / Beautify



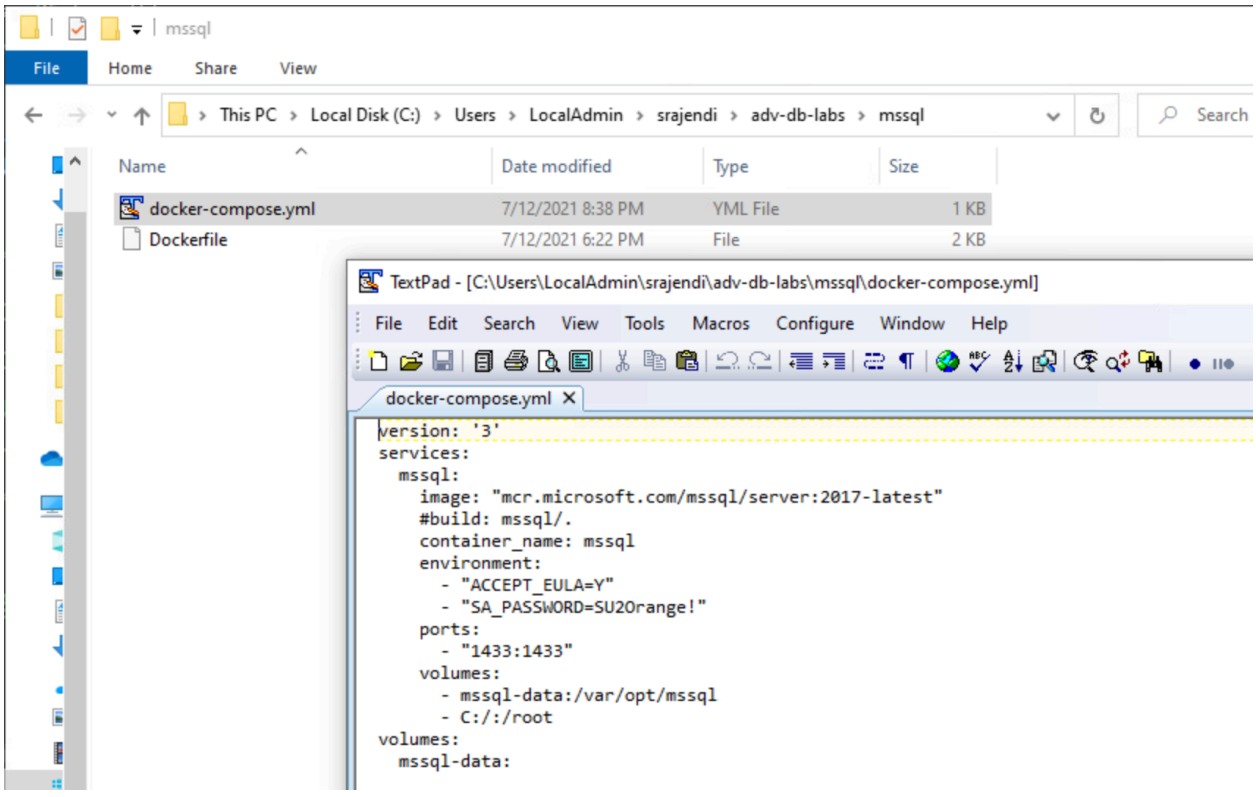
Minify / Compact

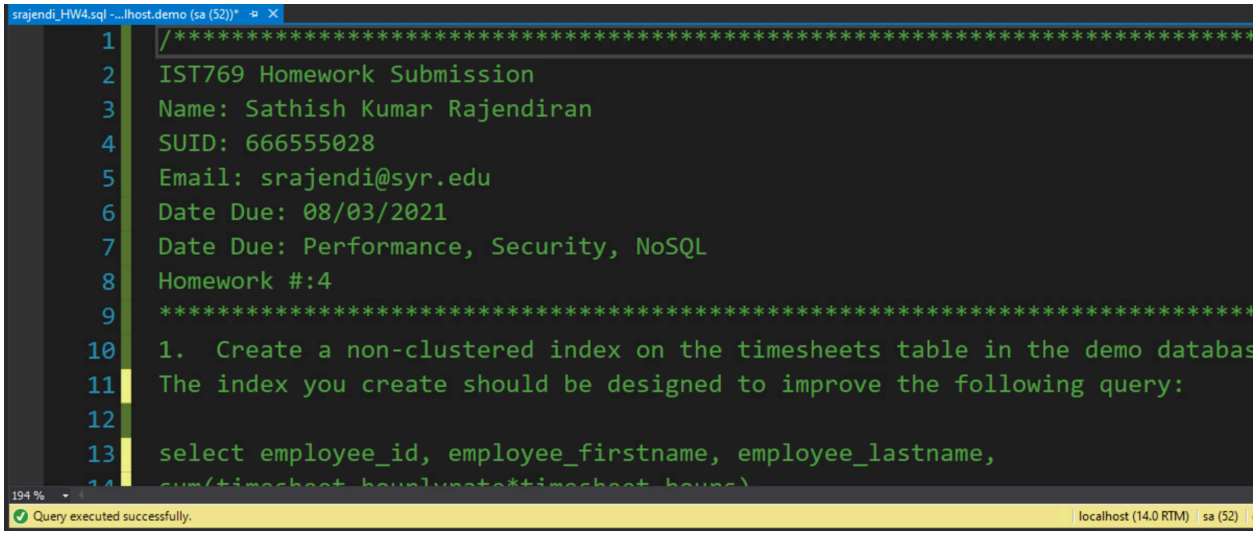
Code

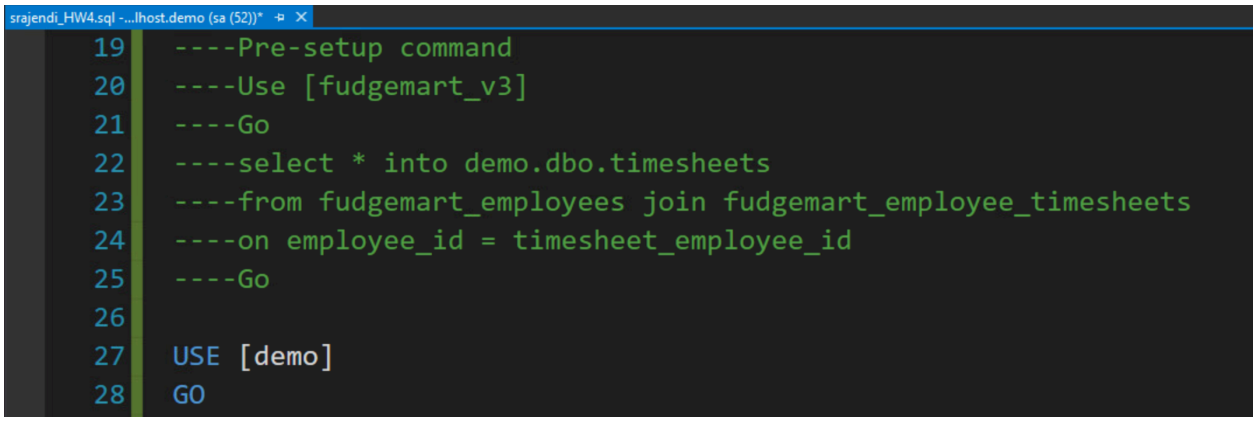

```
[
  {
    "employee_id": 14,
    "employee_firstname": "Chris P.",
    "employee_lastname": "Nugget",
    "timesheet_count": 208,
    "total_hours_worked": 8320,
    "avg_timesheet_rate": 15.1769
  },
  {
    "employee_id": 22,
    "employee_firstname": "Mike",
    "employee_lastname": "Rophone",
    "timesheet_count": 208,
    "total_hours_worked": 3338,
    "avg_timesheet_rate": 11.3184
  },
  {
    "employee_id": 24,
    "employee_firstname": "Otto",
    "employee_lastname": "Moni",
    "timesheet_count": 208,
```

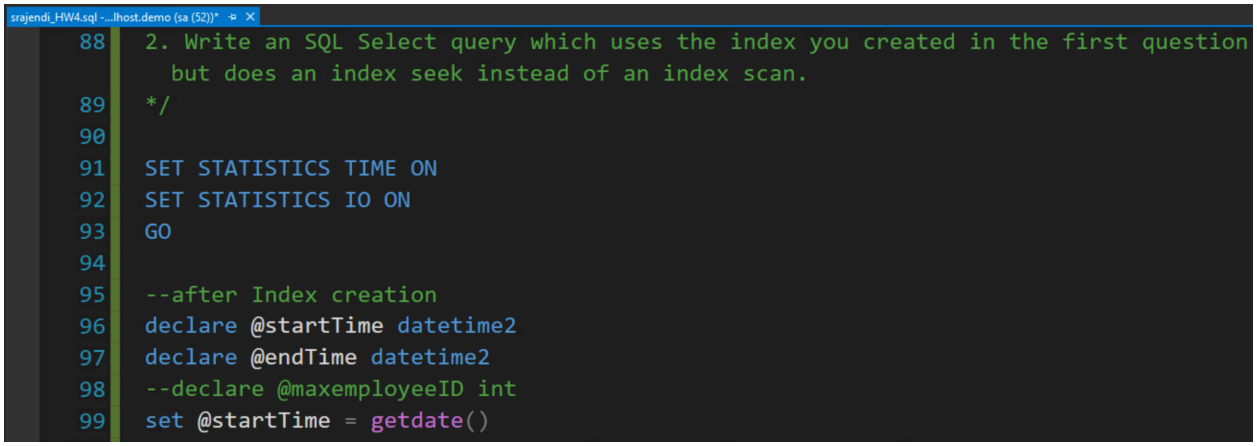
Appendix

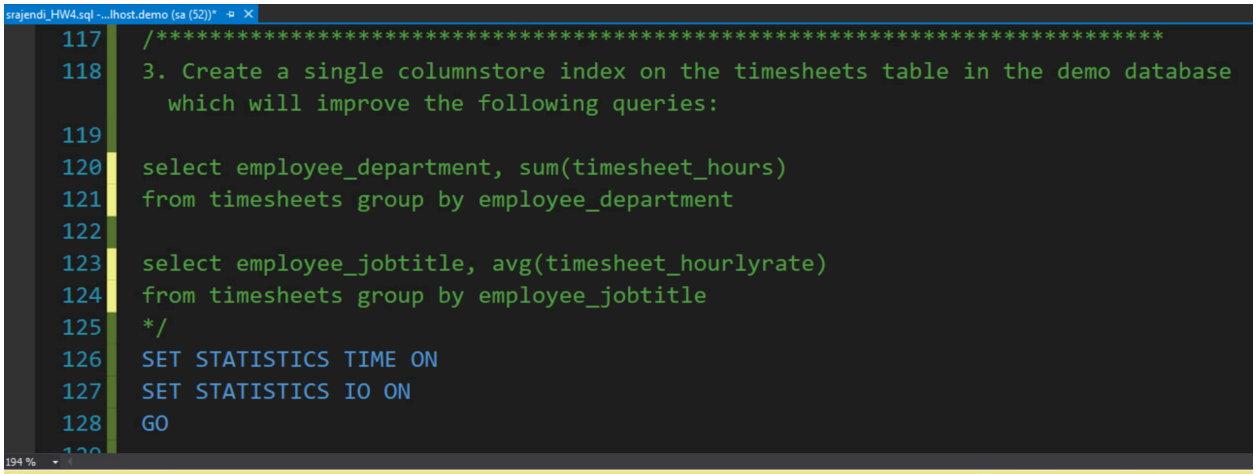

Administrator: Command Prompt
Microsoft Windows [Version 10.0.19041.1110]
(c) Microsoft Corporation. All rights reserved.
C:\WINDOWS\system32>cd C:\Users\LocalAdmin\srajendi\adv-db-labs\mssql
C:\Users\LocalAdmin\srajendi\adv-db-labs\mssql>docker-compose ps
Name Command State Ports
C:\Users\LocalAdmin\srajendi\adv-db-labs\mssql>docker-compose up -d
Creating network "mssql_default" with the default driver
Creating mssql ... done
C:\Users\LocalAdmin\srajendi\adv-db-labs\mssql>docker-compose ps
Name Command State Ports
mssql /opt/mssql/bin/nonroot_msg ... Up 0.0.0.0:1433->1433/tcp,:::1433->1433/tcp
C:\Users\LocalAdmin\srajendi\adv-db-labs\mssql>

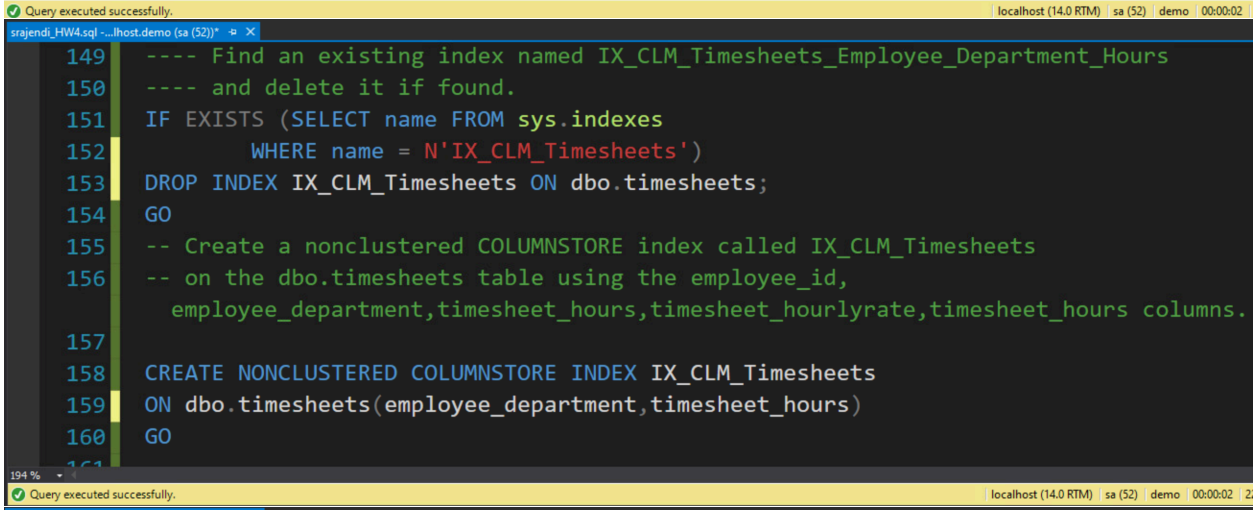

mssql
File Home Share View
This PC › Local Disk (C:) › Users › LocalAdmin › srajendi › adv-db-labs › mssql
Search
Name Date modified Type Size
docker-compose.yml 7/12/2021 8:38 PM YML File 1 KB
Dockerfile 7/12/2021 6:22 PM File 2 KB
TextPad - [C:\Users\LocalAdmin\srajendi\adv-db-labs\mssql\docker-compose.yml]
File Edit Search View Tools Macros Configure Window Help
docker-compose.yml
version: '3'
services:
mssql:
image: "mcr.microsoft.com/mssql/server:2017-latest"
#build: mssql/.
container_name: mssql
environment:
- "ACCEPT_EULA=Y"
- "SA_PASSWORD=SU2Orange!"
ports:
- "1433:1433"
volumes:
- mssql-data:/var/opt/mssql
- C:/:/root
volumes:
mssql-data:

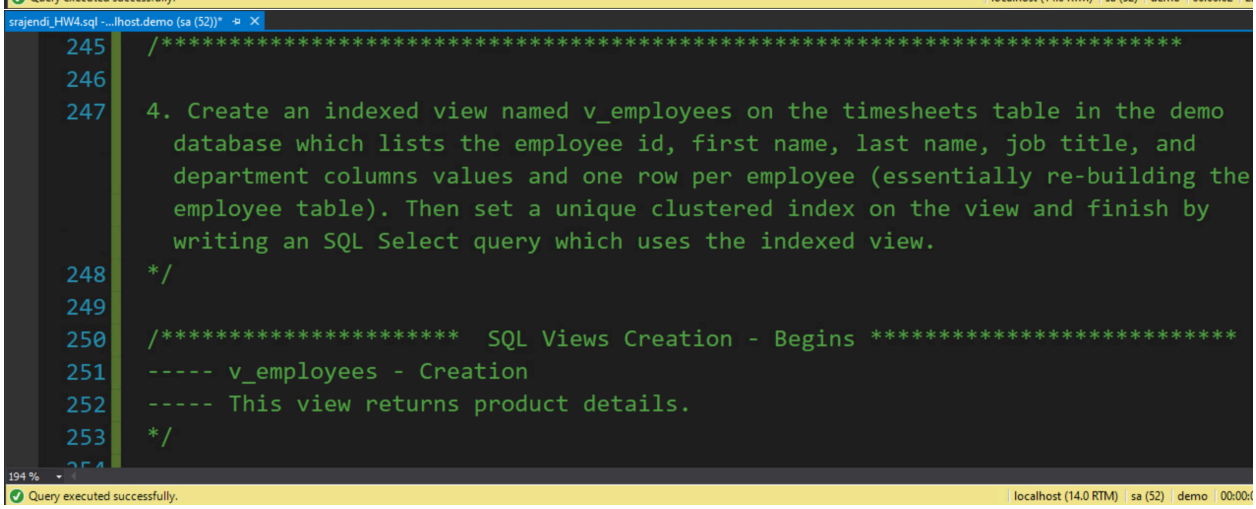

srajendi_HW4.sql -...lhost.demo (sa (52))*
/***
IST769 Homework Submission
Name: Sathish Kumar Rajendiran
SUID: 666555028
Email: srajendi@syr.edu
Date Due: 08/03/2021
Date Due: Performance, Security, NoSQL
Homework #:4
****
1. Create a non-clustered index on the timesheets table in the demo databas
The index you create should be designed to improve the following query:
select employee_id, employee_firstname, employee_lastname,
194 %
Query executed successfully.
localhost (14.0 RTM)
sa (52)


srajendi_HW4.sql -...lhost.demo (sa (52))*
----Pre-setup command
----Use [fudgemart_v3]
----Go
----select * into demo.dbo.timesheets
----from fudgemart_employees join fudgemart_employee_timesheets
----on employee_id = timesheet_employee_id
----Go
USE [demo]
GO


srajendi_HW4.sql -...lhost.demo (sa (52))*
2. Write an SQL Select query which uses the index you created in the first question
but does an index seek instead of an index scan.
*/
SET STATISTICS TIME ON
SET STATISTICS IO ON
GO
--after Index creation
declare @startTime datetime2
declare @endTime datetime2
--declare @maxemployeeID int
set @startTime = getdate()


srajendi_HW4.sql -...lhost.demo (sa (52))*
117 /***
118 3. Create a single columnstore index on the timesheets table in the demo database
which will improve the following queries:
119
120 select employee_department, sum(timesheet_hours)
121 from timesheets group by employee_department
122
123 select employee_jobtitle, avg(timesheet_hourlyrate)
124 from timesheets group by employee_jobtitle
125 */
126 SET STATISTICS TIME ON
127 SET STATISTICS IO ON
128 GO
194 %
Query executed successfully.
localhost (14.0 RTM) | sa (52) | demo | 00:00:02


srajendi_HW4.sql -...lhost.demo (sa (52))*
149 ---- Find an existing index named IX_CLM_Timesheets_Employee_Department_Hours
150 ---- and delete it if found.
151 IF EXISTS (SELECT name FROM sys.indexes
152 WHERE name = N'IX_CLM_Timesheets')
153 DROP INDEX IX_CLM_Timesheets ON dbo.timesheets;
154 GO
155 -- Create a nonclustered COLUMNSTORE index called IX_CLM_Timesheets
156 -- on the dbo.timesheets table using the employee_id,
employee_department,timesheet_hours,timesheet_hourlyrate,timesheet_hours columns.
157
158 CREATE NONCLUSTERED COLUMNSTORE INDEX IX_CLM_Timesheets
159 ON dbo.timesheets(employee_department,timesheet_hours)
160 GO
194 %
Query executed successfully.
localhost (14.0 RTM) | sa (52) | demo | 00:00:02


srajendi_HW4.sql -...lhost.demo (sa (52))*
245 /***
246
247 4. Create an indexed view named v_employees on the timesheets table in the demo
database which lists the employee id, first name, last name, job title, and
department columns values and one row per employee (essentially re-building the
employee table). Then set a unique clustered index on the view and finish by
writing an SQL Select query which uses the indexed view.
248 */
249
250 /********************* SQL Views Creation - Begins *************************
251 ----- v_employees - Creation
252 ----- This view returns product details.
253 */
194 %
Query executed successfully.
localhost (14.0 RTM) | sa (52) | demo

srajendi_HW4.sql -...lhost.demo (sa (52))*

```
261  CREATE VIEW  v_employees WITH schemabinding
262  AS
263  SELECT    employee_id
264       , employee_firstname
265       , employee_lastname
266       , employee_jobtitle
267       , employee_department
268       ,COUNT_BIG(*) as employee_count
269  from dbo.timesheets
270  Group by
271  employee_id
272  , employee_firstname
273  , employee_lastname
```

194 %

Query executed successfully.

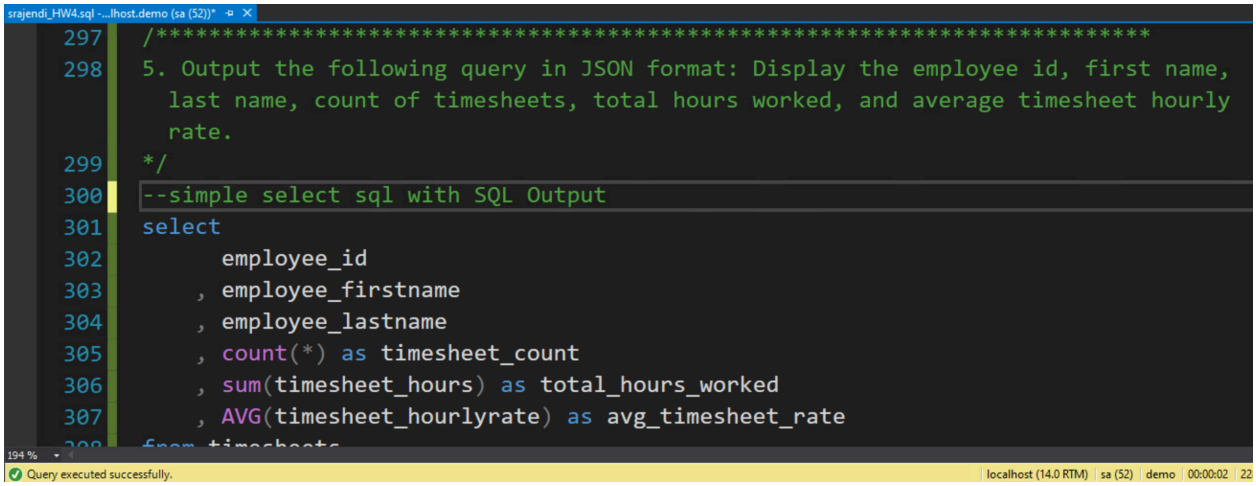
srajendi_HW4.sql -...lhost.demo (sa (52))*

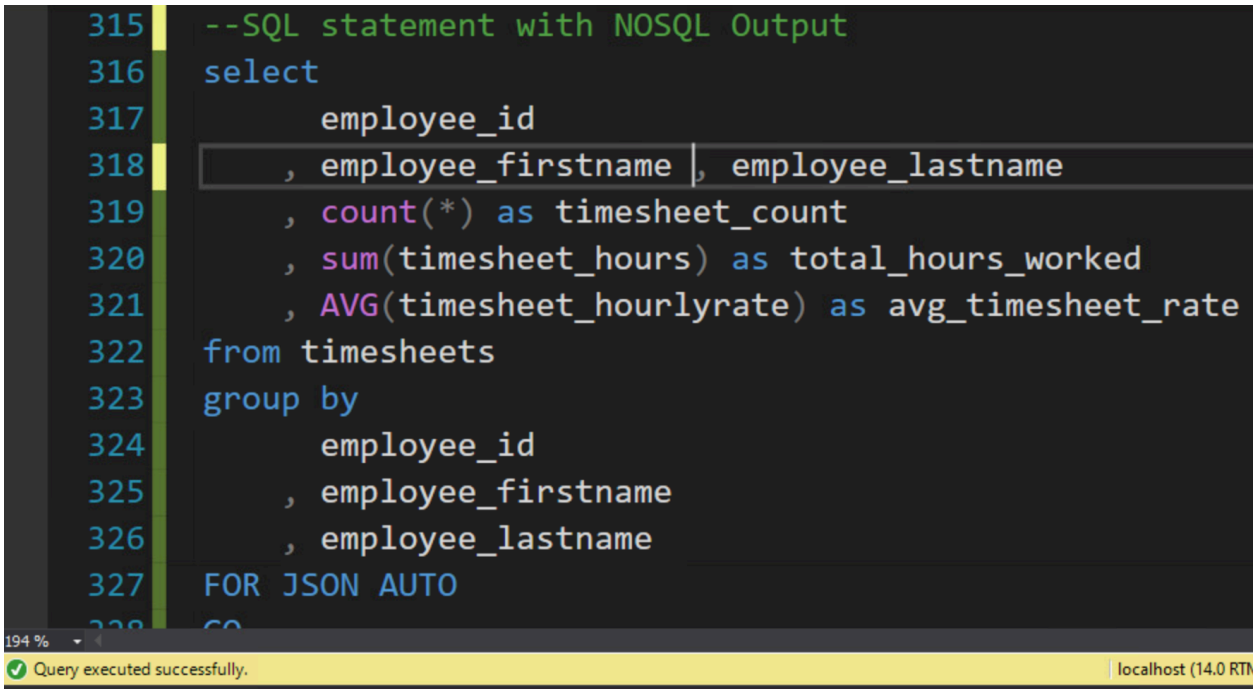
```
285  DROP INDEX IX_CLS_v_employees_EmployeeID ON dbo.v_employees;
286  GO
287  -- Create an unique clustered index called IX_CLS_v_employees_EmployeeID
288  -- on the dbo.v_employees table using the employee_id
289
290  CREATE UNIQUE CLUSTERED  INDEX IX_CLS_v_employees_EmployeeID
291  ON dbo.v_employees(employee_id)
292  GO
293  --query from the view after index creation
294  SELECT * FROM dbo.v_employees
295  GO
296
297  /*****************************************************************
```

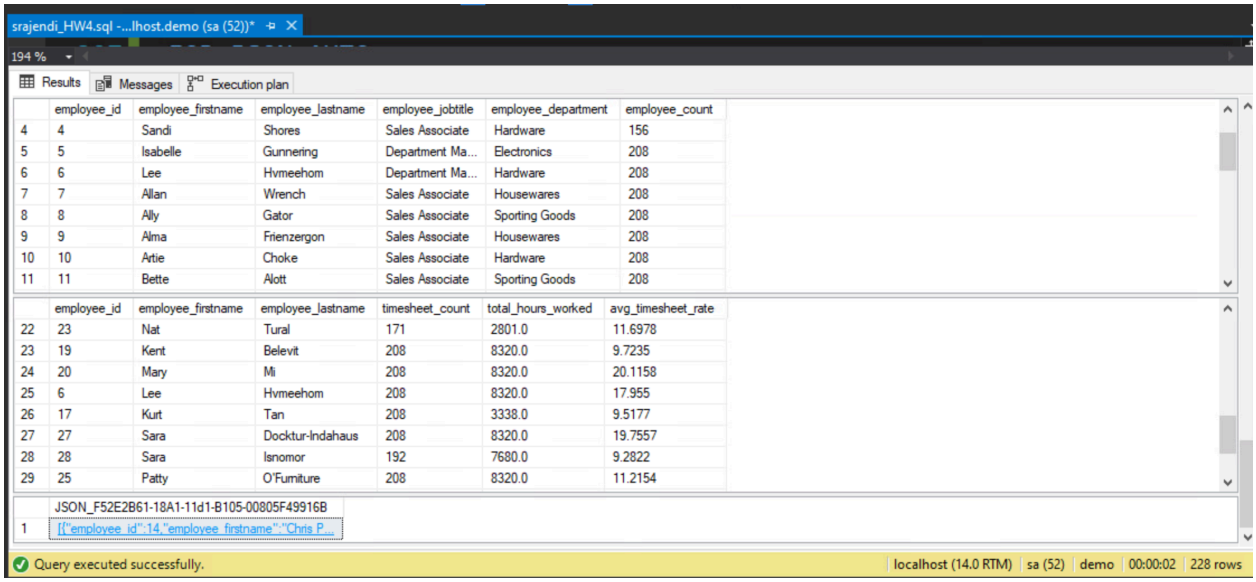
194 %

Query executed successfully.

localhost (14.0 RTM) | sa (52)


srajendi_HW4.sql -...lhost.demo (sa (52))*
297 /***
298 5. Output the following query in JSON format: Display the employee id, first name,
last name, count of timesheets, total hours worked, and average timesheet hourly
rate.
299 */
300 --simple select sql with SQL Output
301 select
302 employee_id
303 , employee_firstname
304 , employee_lastname
305 , count(*) as timesheet_count
306 , sum(timesheet_hours) as total_hours_worked
307 , AVG(timesheet_hourlyrate) as avg_timesheet_rate
194 %
Query executed successfully.
localhost (14.0 RTM)
sa (52)
demo
00:00:02

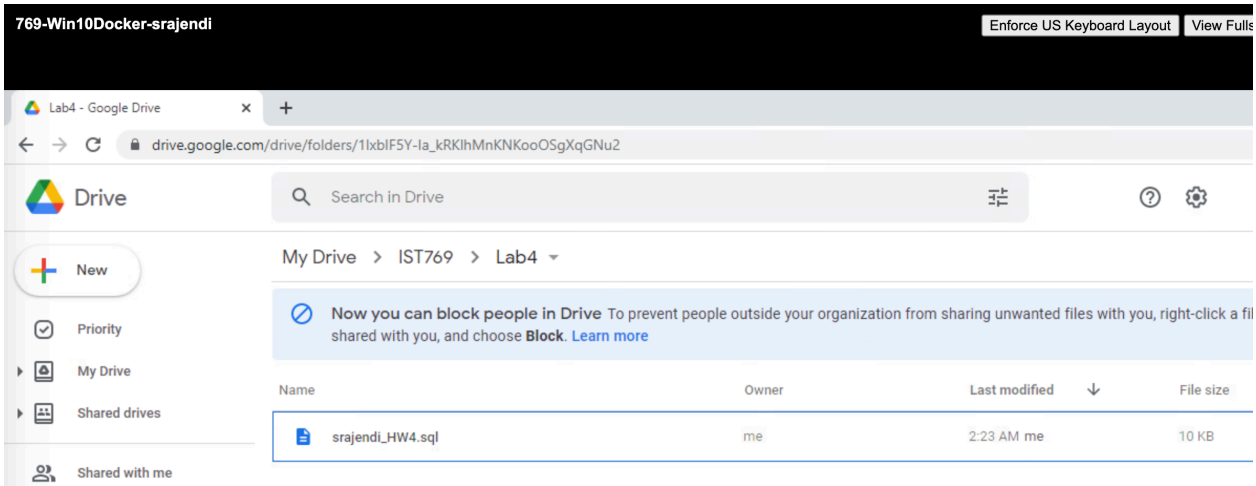

315 --SQL statement with NOSQL Output
316 select
317 employee_id
318 , employee_firstname , employee_lastname
319 , count(*) as timesheet_count
320 , sum(timesheet_hours) as total_hours_worked
321 , AVG(timesheet_hourlyrate) as avg_timesheet_rate
322 from timesheets
323 group by
324 employee_id
325 , employee_firstname
326 , employee_lastname
327 FOR JSON AUTO
194 %
Query executed successfully.
localhost (14.0 RTM

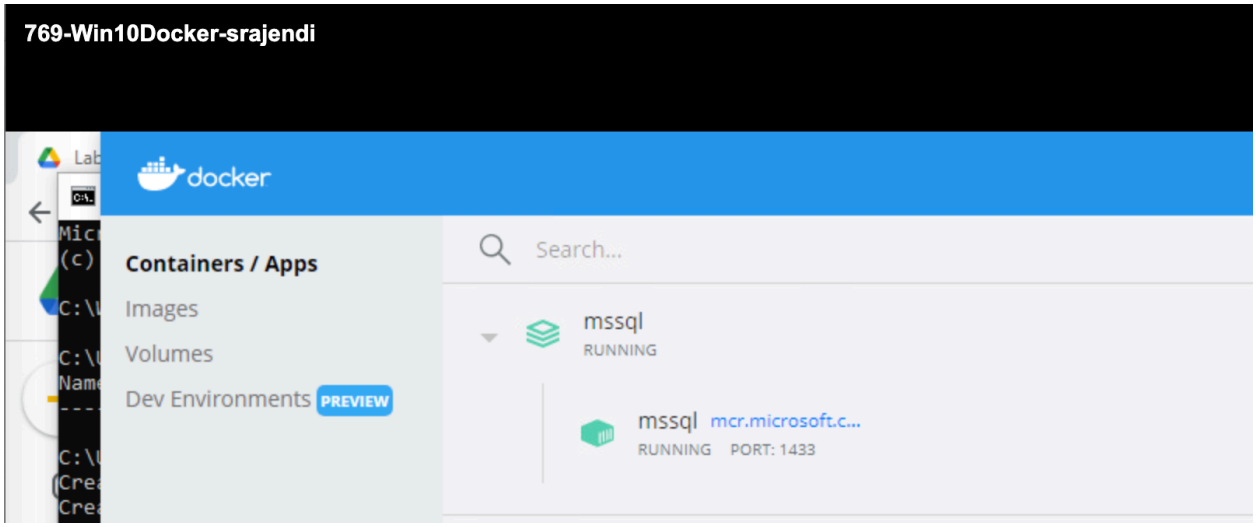


	employee_id	employee_firstname	employee_lastname	employee_jobtitle	employee_department	employee_count
4	4	Sandi	Shores	Sales Associate	Hardware	156
5	5	Isabelle	Gunnering	Department Ma...	Electronics	208
6	6	Lee	Hvmeehom	Department Ma...	Hardware	208
7	7	Allan	Wrench	Sales Associate	Housewares	208
8	8	Ally	Gator	Sales Associate	Sporting Goods	208
9	9	Alma	Frienzergon	Sales Associate	Housewares	208
10	10	Artie	Choke	Sales Associate	Hardware	208
11	11	Bette	Alott	Sales Associate	Sporting Goods	208

	employee_id	employee_firstname	employee_lastname	timesheet_count	total_hours_worked	avg_timesheet_rate
22	23	Nat	Tural	171	2801.0	11.6978
23	19	Kent	Belevit	208	8320.0	9.7235
24	20	Mary	Mi	208	8320.0	20.1158
25	6	Lee	Hvmeehom	208	8320.0	17.955
26	17	Kurt	Tan	208	3338.0	9.5177
27	27	Sara	Docktur-Indahaus	208	8320.0	19.7557
28	28	Sara	Isnomor	192	7680.0	9.2822
29	25	Patty	O'Furniture	208	8320.0	11.2154

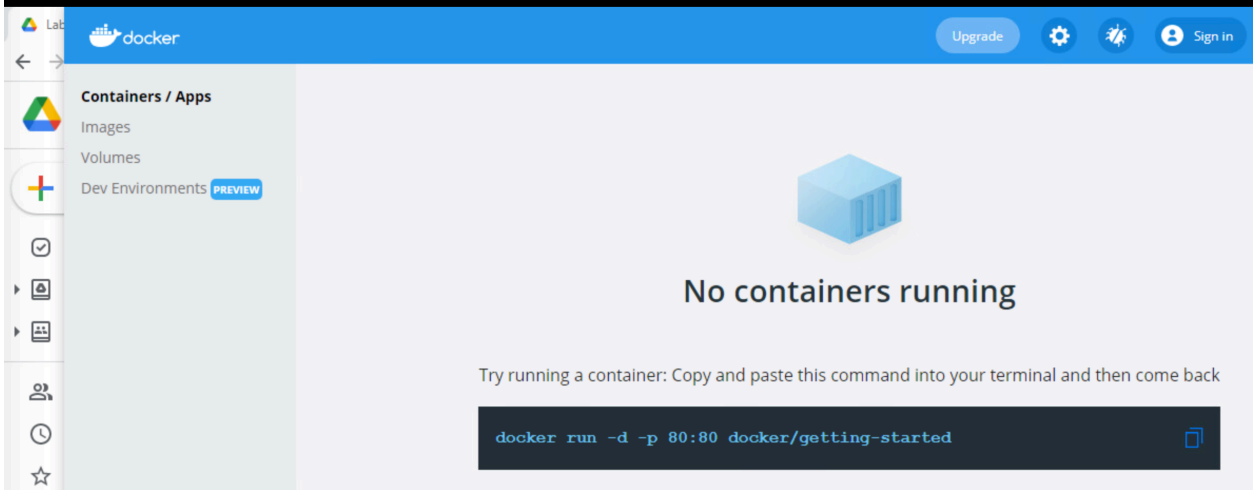
	JSON_F52E2B61-18A1-11d1-B105-00805F49916B
1	[{"employee_id":14,"employee_firstname":"Chris P...



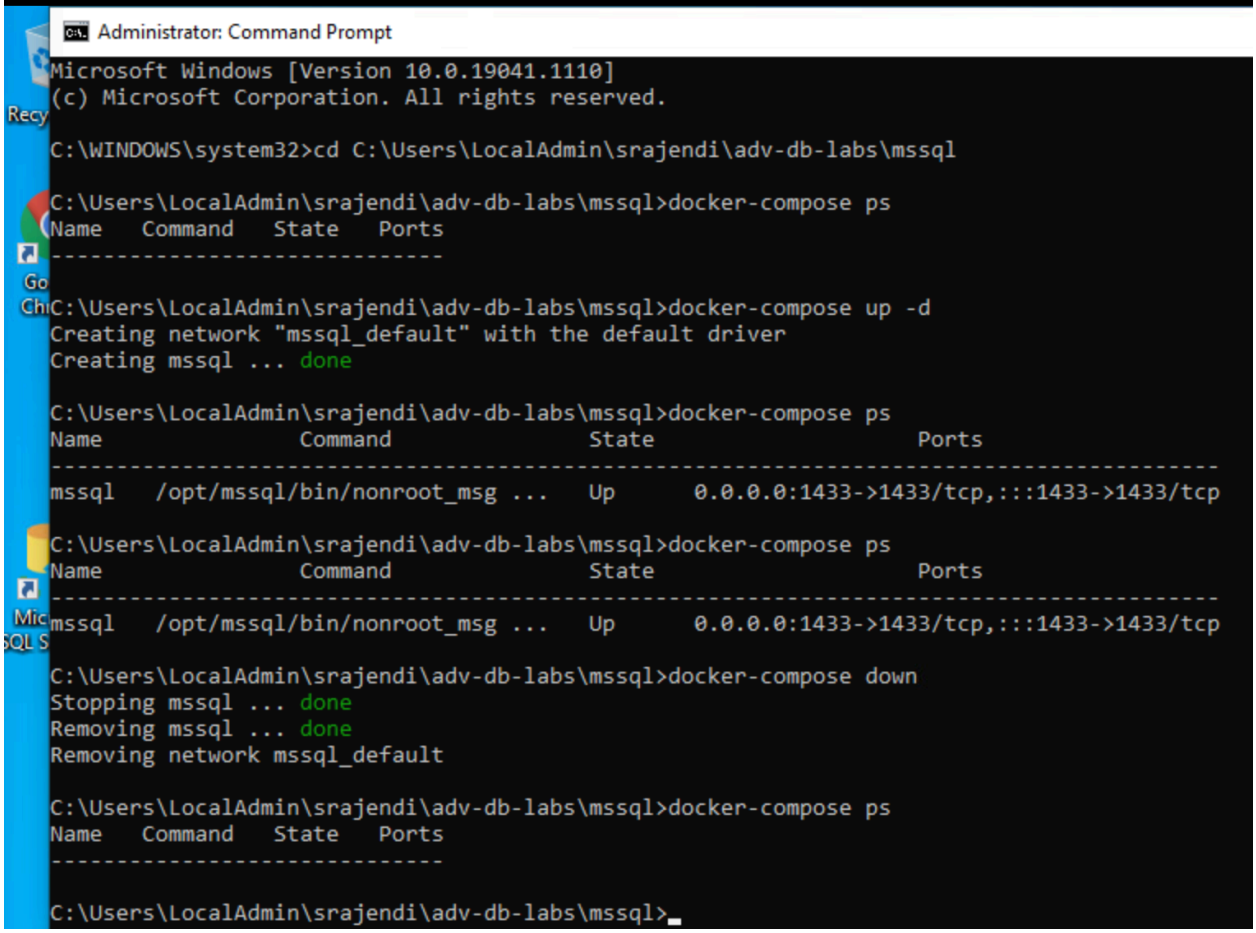


769-Win10Docker-srajendi

Enforce US Keyboard Layout | View Full



769-Win10Docker-srajendi



769-Win10Docker-srajendi

Enforce US Keyboard Layout

The console has been disconnected. Close this window and re-launch the console to reconnect.