

## Instructions

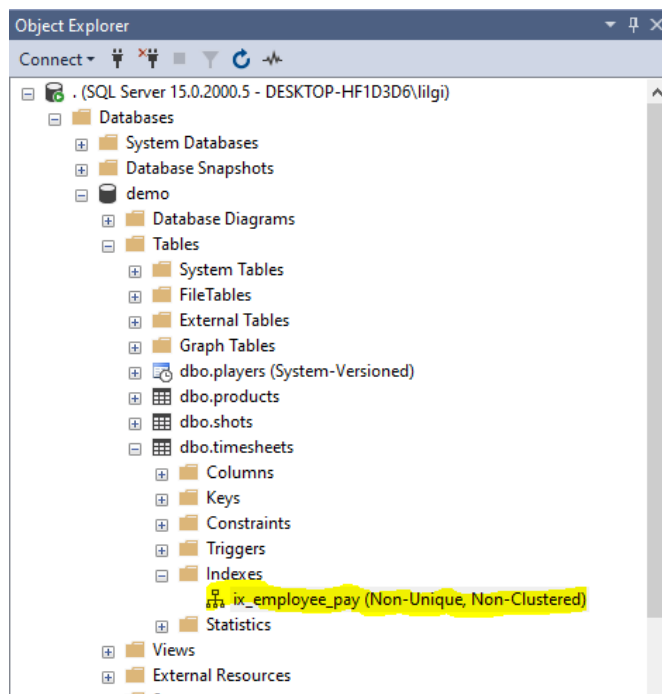
For each answer, please include your answer as text, and any screenshot(s) which demonstrate your answer was executed. Most importantly, make sure to include evidence your answer is correct. This will most likely be a screenshot. If you had issues, problems, or had to make assumptions include them in your answer.

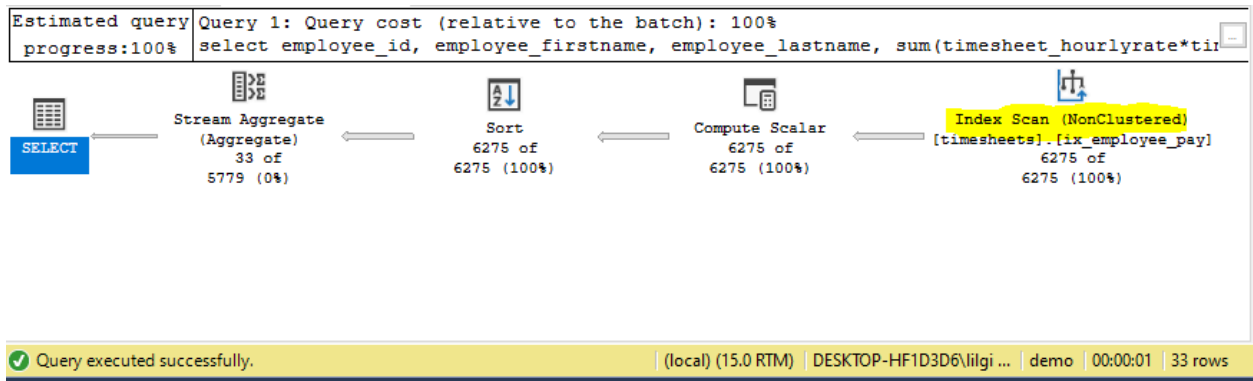
## Your Answers:

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;
```

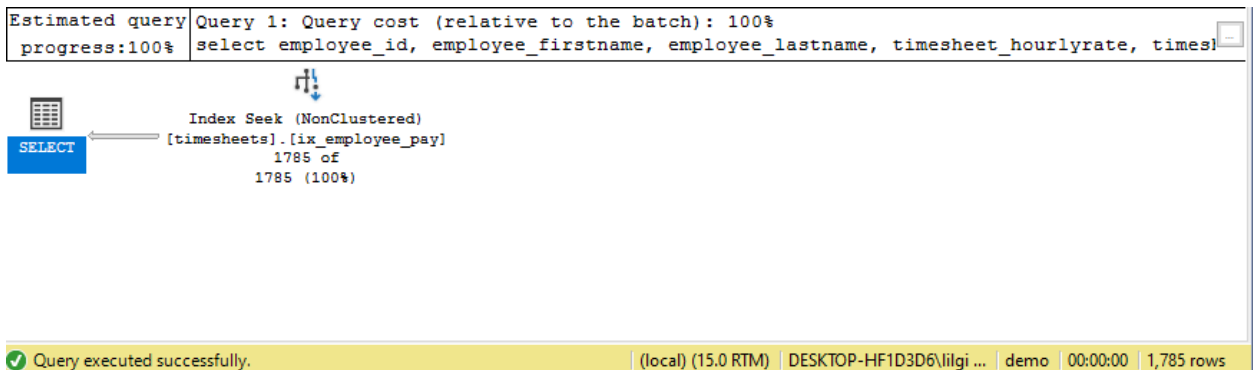
```
drop index if exists timesheets.ix_employee_pay  
create index ix_employee_pay  
on timesheets (employee_id)  
include (employee_firstname, employee_lastname, timesheet_hourlyrate,  
timesheet_hours)
```





- 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.

```
select employee_id, employee_firstname, employee_lastname, timesheet_hourlyrate,
timesheet_hours
from timesheets
where employee_id in (1, 2, 3, 4, 5, 6, 7, 8, 9, 10)
```

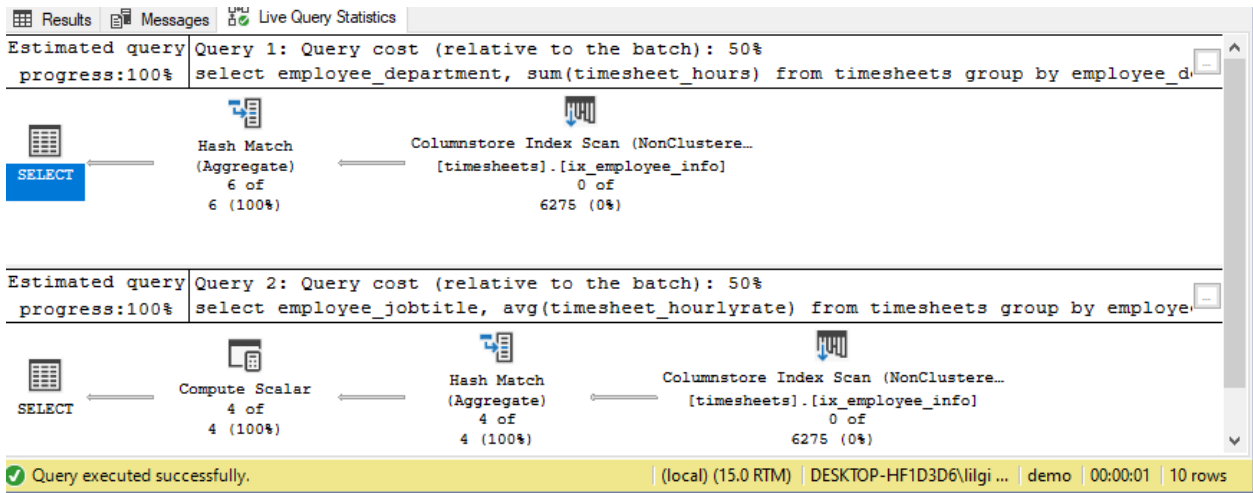


- Create a single columnstore index on the **timesheets** table in the **demo** database which will improve the following queries:

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

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

```
drop index if exists timesheets.ix_employee_info
create nonclustered columnstore index ix_employee_info
on timesheets (employee_department, employee_jobtitle, timesheet_hours,
timesheet_hourlyrate)
```



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.

```
drop view if exists v_employees
go
```

```
create view v_employees
with schemabinding
as
```


```
select employee_id, employee_firstname, employee_lastname, employee_jobtitle,
employee_department, COUNT_BIG(*) as tmp
from dbo.timesheets
group by employee_id, employee_firstname, employee_lastname, employee_jobtitle,
employee_department
```


```
go
```

```
create unique clustered index ix_employees on v_employees(employee_id)
```

```
-- use the indexed view
select employee_id, employee_firstname, employee_lastname
from v_employees
where employee_department in ('Sporting Goods', 'Hardware')
```

Estimated query progress:100% Query 1: Query cost (relative to the batch): 100%  
 select employee\_id, employee\_firstname, employee\_lastname from v\_employees where emplo;

 Clusters Index Scan (ViewClustered)  
 [v\_employees].[ix\_employees]  
 13 of  
 2208 (0%)

 SELECT

Query executed successfully. (local) (15.0 RTM) DESKTOP-HF1D3D6\lilgi ... demo 00:00:00 13 rows

- 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.

```
select
  employee_id,
  employee_firstname,
  employee_lastname,
  count(timesheet_id) as timesheet_count,
  sum(timesheet_hours) as total_hours_worked,
  avg(timesheet_hourlyrate) as avg_timesheet_hourly_rate
from
  timesheets
group by
  employee_id,
  employee_firstname,
  employee_lastname
for json auto
```

Results Messages Live Query Statistics

JSON\_F52E2B61-18A1-11d1-B105-00805F49916B

1 [{"employee\_id":1,"employee\_firstname":"Arial",...}]

Query executed successfully. (local) (15.0 RTM) DESKTOP-HF1D3D6\lilgi ... demo 00:00:02 1 rows