IST769 Performance, Security, NoSQL

Your Name: Tajudeen Abdulazeez   
Your SUID: 69687-7373-0   
Your Email: toabdula@syr.edu   
Date Due:   
Homework #: 4

* 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;

GO

CREATE nonclustered index t\_timesheet\_index\_nonclus

ON employee\_timesheets (employee\_id)

INCLUDE (employee\_firstname, employee\_lastname, timesheet\_hourlyrate,timesheet\_hours)

GO

select employee\_id, employee\_firstname, employee\_lastname, sum(timesheet\_hourlyrate\*timesheet\_hours) as salary

from timesheets

group by employee\_id, employee\_firstname, employee\_lastname;

GO

* 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, sum(timesheet\_hourlyrate\*timesheet\_hours)

from timesheets

where employee\_id IN (1,2,3,4)

group by employee\_id, employee\_firstname, employee\_lastname;

* 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

GO

CREATE nonclustered columnstore index t\_timesheets\_index\_colstore

ON employee\_timesheets

--WITH (drop existing = on)

GO

select employee\_department, sum(timesheet\_hours)

from employee\_timesheets group by employee\_department

GO

select employee\_jobtitle, avg(timesheet\_hourlyrate)

from timesheets group by employee\_jobtitle

* 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.

GO

CREATE VIEW v\_employees

WITH schemabinding

AS

SELECT employee\_id, first\_name, last\_name, job\_title, employee\_department from timesheets

GO

CREATE UNIQUE clustered index v\_timesheet\_uniq\_clus

ON timesheets

GO

SELECT \* FROM v\_employees

GO

* 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.

GO

select

employee\_id,

first\_name,

last\_name,

count(timesheet) as counts,

sum(total\_hours) as total\_hours,

AVG(timesheet\_hour\_rate) as avgrate

from timesheets

group by employee\_id, first\_name, last\_name

FOR JSON AUTO

GO

* Q2 >> hw2 >> correction

2. Write transaction safe code as a stored procedure which when given a player id,

clock time, and whether the shot was made (bit value) will add the record to the shots table and

update the player record in the players table. For example, If Mary takes a shot and makes it,

then misses the next one, there would be two records in the shots table and her row in the players

table should have 2 attempt and 1 shot made. Execute the stored procedure to demonstrate the transaction is ACID compliant.

GO

CREATE PROCEDURE dbo.write\_shot(

@player\_id INT,

@clock\_time datetime,

@shot\_made bit

)

AS

BEGIN TRY

BEGIN TRANSACTION

INSERT dbo.shots (player\_id, clock\_time, shot\_made)

VALUES (@player\_id, @clock\_time,@shot\_made)

if @@ROWCOUNT <> 1 THROW 50005, 'Failed to update shots table, zero rows affected',0;

UPDATE dbo.players

SET shots\_attemted = COALESCE (shots\_attemted, 0) + 1,

shots\_made = CASE @shot\_made WHEN 1 THEN COALESCE (shots\_made, 0) + 1

ELSE shots\_made END

WHERE player\_id = @player\_id

if @@ROWCOUNT <> 1 THROW 50006,'Failed to insert into player table, zero rows affected',0;

COMMIT

END TRY

BEGIN CATCH

select error\_number() as error, error\_message() as message

print 'Rolling back'

rollback

END CATCH

GO