Partitioning Technicals

ISM 6218

Due on November 6th

The Avengers Team

"We will avenge every problem on our way"

Aitemir Yeskenov (Team Lead)

Nagarjuna Kanneganti

Sai Suraj Argula

Vinay Kumar Reddy Baradi

Table of Contents

Partitioning Assignment

Requirements

Required Elements:

- MDF versus NDF file allocation
- Create File Groups
- Create Partition Scheme and Partition Function
- Partition must distribute database table across server group
- Use of switch, merge, split, rebuild

Demonstrate:

- Query entire table across partition plan
- Query using a linked server
- Query using openquery operator
- Query using registered server group

Deliverables:

Upload Report from Team Lab Notebook

Refer to the Slide in the PPT for specific tasks I am testing.

Partitioning Assignment

• Create a table.

• Add rows.

• Run query, analyze execution plan.

• Partition Table.

• You need to deploy partition on different server.

• Run query, analyze execution plan.

• Generate report of experiment with a supported user story.

• You must also demonstrate: Switching, Merging, Splitting, Rebuild

Business Process Supported

We have created the Employee database which consists of the employee details such as name, Salary, EmployeeSalaryDay over a year(1-365 days) and EmployeeSalaryLink which consists of salary calculation details.

USER STORY:

HR wants to analyze the salary of the employee on daily basis of the Employee Vinay for annual rating.

Create a table and add rows.

We have created a EMployeeSalary Table which consists of EmployeeDetails such as name, Salary, EmployeeSalaryDay over a year(1-365 days) and EmployeeSalaryLink which consists of salary calculation details.

```
CREATE TABLE EmployeeSalary(
EmployeeName varchar (100),
EmployeeSalary varchar (20),
EmployeeSalaryDay varchar (max),
SalaryLink varchar(100)
CONSTRAINT Employee_PK PRIMARY KEY CLUSTERED (EmployeeId)
)
```

esults | Messages | Execution plan

EmployeeId	EmployeeName	EmployeeSalary	EmployeeSalaryDay	SalaryLink
1	Vinay	100	1	http//Company

Run query, analyze execution plan

HR wants to know the salary details of the Employee on the 100th working day of the Employee Vinay.

Here in the below snapshots we can see the salary details of the employee on 100th day along with execution time and plan.

```
truncate table EmployeeSalary
SET STATISTICS IO ON
SET STATISTICS TIME ON
SELECT *
FROM dbo.EmployeeSalary
WHERE EmployeeSalaryDay = 100
SET STATISTICS IO OFF
```

100 %

Results | Messages

	EmployeeName	EmployeeSalary	EmployeeSalaryDay	SalaryLink
1	Vinay	100	100	http//Company

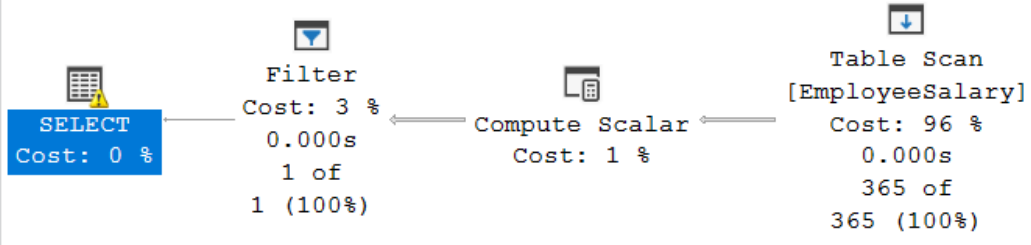
Execution Plan:

```
SET STATISTICS IO ON
SET STATISTICS TIME ON
SELECT *
FROM dbo.EmployeeSalary
WHERE EmployeeSalaryDay = 100
SET STATISTICS IO OFF
```

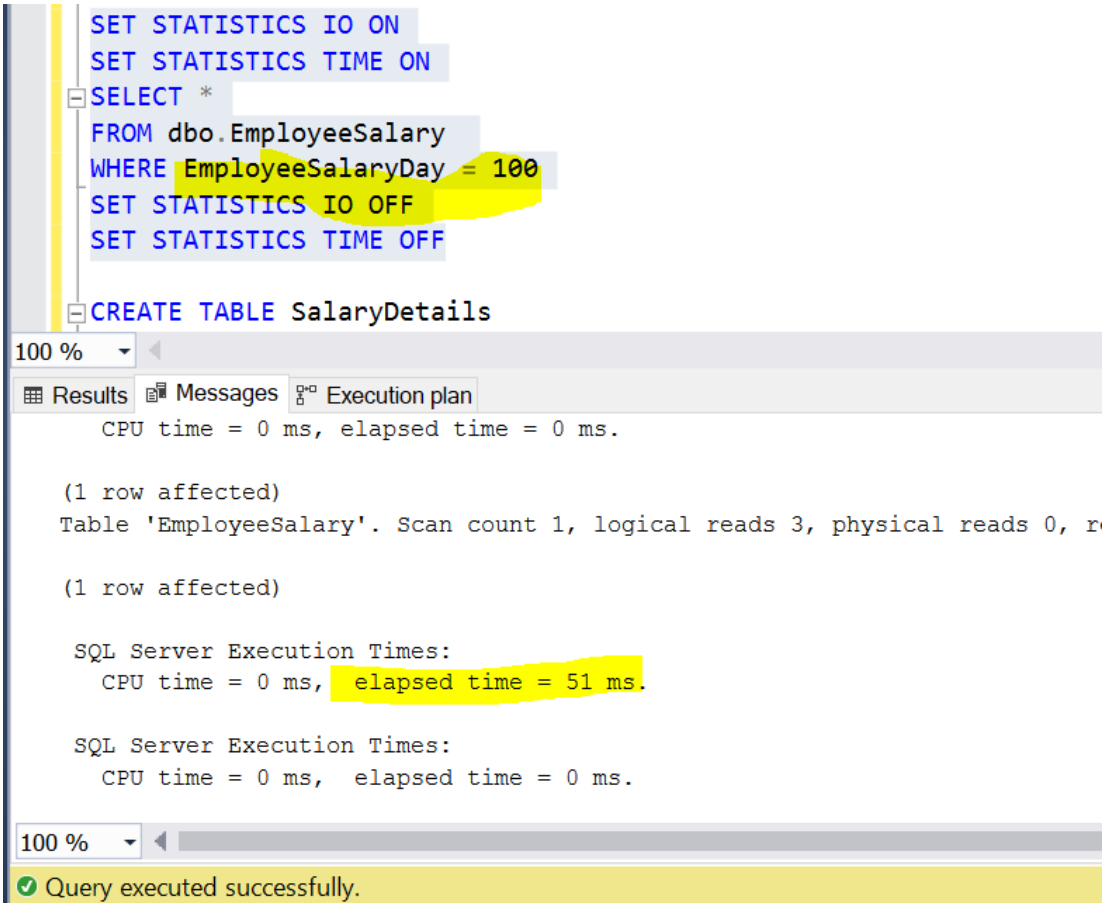
100 %

Results | Messages | Execution plan

```
Query 1: Query cost (relative to the batch): 100%
SELECT * FROM [dbo].[EmployeeSalary] WHERE [EmployeeSalaryDay]=@1
```



Query executed successfully. DESK



Partitioning the database based on Quarterly Payments of Employee to analyze performance:

Creating Filegroups:

```
ALTER DATABASE Employee2
ADD FILEGROUP Q1
GO
ALTER DATABASE Employee2
ADD FILEGROUP Q2
GO
ALTER DATABASE Employee2
ADD FILEGROUP Q3
GO
ALTER DATABASE Employee2
ADD FILEGROUP Q4
GO
```

100 %

Messages

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

Completion time: 2019-11-06T21:48:47.4355406-05:00
```

100 %

Query executed successfully.

Exisiting Filegroup Names:

```
SELECT name AS AvailableFilegroups
FROM sys.filegroups
WHERE type = 'FG'
```

100 %

Results | Messages | Execution plan

	AvailableFilegroups
1	PRIMARY
2	Q1
3	Q2
4	Q3
5	Q4

Query executed successfully.

Checking existing filegroups and their locations:

MDF versus NDF file allocation

```
SELECT
name as [FileName],
physical_name as [FilePath]
FROM sys.database_files
where type_desc = 'ROWS'
GO
```

100 %

Results | Messages | Execution plan

	FileName	FilePath
1	Employee2	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\Employee2.mdf
2	PartQ1	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\Employee2DB1.ndf
3	PartQ2	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\Employee2DB2.ndf
4	PartQ3	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\Employee2DB3.ndf
5	PartQ4	C:\Program Files\Microsoft SQL Server\MSSQL14.MSSQLSERVER\MSSQL\DATA\Employee2DB4.ndf

Partition Function and Scheme:

We have partitioned the Employeesalaryday over a year of 365 days into 4 quarters Q1,Q2,Q3,Q4.

```
drop  PARTITION FUNCTION [PartitioningByQuarters]
Create PARTITION FUNCTION [PartitioningByQuarters] (int)
AS RANGE RIGHT FOR VALUES (90, 180,270);

CREATE PARTITION SCHEME PartitionByQuarter
AS PARTITION PartitioningByQuarters
TO (Q1, Q2, Q3, Q4);
```

100 %

Messages

```
 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 parse and compile time:
   CPU time = 0 ms, elapsed time = 0 ms.

Completion time: 2019-11-06T21:54:19.3105561-05:00
```

100 %

Query executed successfully. DESKTOP-QJLVPE3 (1

```
SELECT
p.partition_number AS PartitionNumber,
f.name AS PartitionFilegroup,
p.rows AS NumberOfRows
FROM sys.partitions p
JOIN sys.destination_data_spaces dds ON p.partition_number = dds.destination_id
JOIN sys.filegroups f ON dds.data_space_id = f.data_space_id
WHERE OBJECT_NAME(OBJECT_ID) = 'Reports'
```

100 %

Results | Messages | Execution plan

	PartitionNumber	PartitionFilegroup	NumberOfRows
1	1	Q1	89
2	2	Q2	90
3	3	Q3	90
4	4	Q4	96

Split:

Below query is the demonstration of Split in Partitioning where we have split the datapartition 1 into a new partition.

```
ALTER PARTITION FUNCTION [PartitioningByQuarters]()
SPLIT RANGE (60)

SELECT
p.partition_number AS PartitionNumber,
f.name AS PartitionFilegroup,
p.rows AS NumberOfRows
FROM sys.partitions p
JOIN sys.destination_data_spaces dds ON p.partition_number = dds.destination_id
JOIN sys.filegroups f ON dds.data_space_id = f.data_space_id
WHERE OBJECT_NAME(OBJECT_ID) = 'Reports'

ALTER PARTITION FUNCTION [PartitioningByQuarters]()
```

100 %

Results | Messages | Execution plan

	PartitionNumber	PartitionFilegroup	NumberOfRows
1	1	Q1	59
2	3	Q2	90
3	4	Q3	90
4	5	Q4	96
5	2	Q1M1	30

Query executed successfully. DESKTOP-QJLVPE3 (14.0 RTM)

Merge:

Below query is the demonstration of Merge in Partitioning where we have merged the datatable we partitioned in the previous step of split partition.

```
ALTER PARTITION FUNCTION [PartitioningByQuarters]()
merge RANGE (60)

SELECT
p.partition_number AS PartitionNumber,
f.name AS PartitionFilegroup,
p.rows AS NumberOfRows
FROM sys.partitions p
JOIN sys.destination_data_spaces dds ON p.partition_number = dds.destination_id
JOIN sys.filegroups f ON dds.data_space_id = f.data_space_id
WHERE OBJECT_NAME(OBJECT_ID) = 'Reports'

CREATE TABLE ReportsArchive
```

100 %

Results | Messages | Execution plan

	PartitionNumber	PartitionFilegroup	NumberOfRows
1	1	Q1	89
2	2	Q2	90
3	3	Q3	90
4	4	Q4	96

Query executed successfully. DESKTOP-QJLVPE3 (14.0 RTM)

Switch:

Below query is the demonstration of Switch in Partitioning where we have moved the data in partition 1 into employee archive table created.

```
SELECT
p.partition_number AS PartitionNumber,
f.name AS PartitionFilegroup,
p.rows AS NumberOfRows
FROM sys.partitions p
JOIN sys.destination_data_spaces dds ON p.partition_number = dds.destination_id
JOIN sys.filegroups f ON dds.data_space_id = f.data_space_id
WHERE OBJECT_NAME(OBJECT_ID) = 'Reports'

CREATE TABLE ReportsArchive
```

100 %

Results | Messages | Execution plan

	PartitionNumber	PartitionFilegroup	NumberOfRows
1	1	Q1	0
2	2	Q2	90
3	3	Q3	90
4	4	Q4	96

Query executed successfully. DESKTOP-QJLVPE3 (14.0 RTM)

Creating archive table.

```
CREATE TABLE ReportsArchive
(
EmployeeName varchar (100),
EmployeeSalary varchar (20),
EmployeeSalaryDay int,
SalaryLink varchar(100))
ON Q1 ;
GO

ALTER Table Reports
Switch Partition 1 to ReportsArchive
select * from ReportsArchive
```

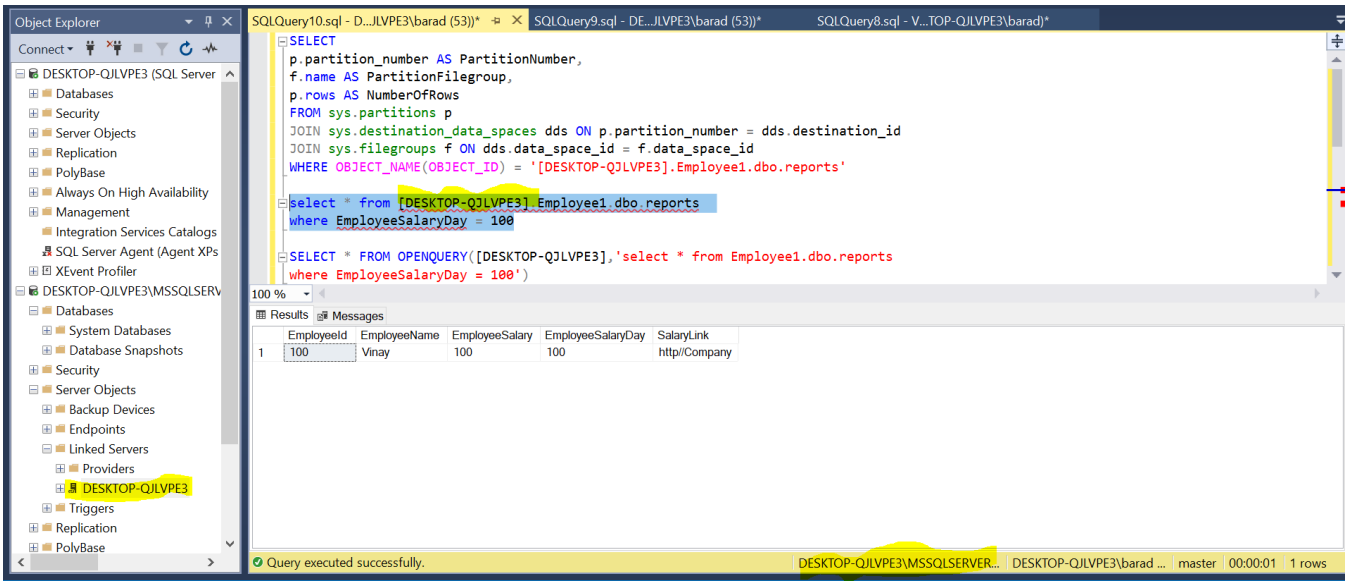
100 %

Results | Messages | Execution plan

	EmployeeName	EmployeeSalary	EmployeeSalaryDay	SalaryLink
1	Vinay	100	1	http//Company

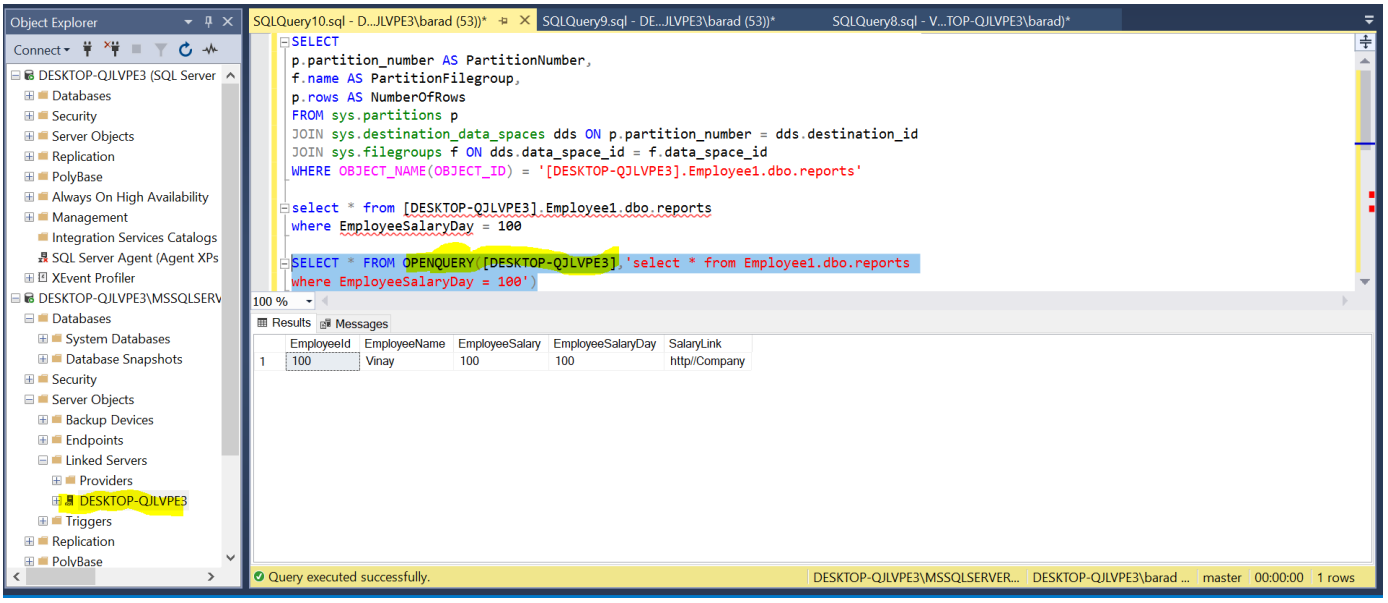
Linked Servers:

Here we have executed query in new server linked to existing server. Below screenshot shows the execution of query we are using for analyzing the past instances to extract salary details of employee Vinay on his 100th day being run in a new query window of new server.



Open Query:

Here we have executed query in new server linked to existing server by OpenQuery. Below screenshot shows the execution of query we are using for analyzing the past instances to extract salary details of employee Vinay on his 100th day being run in a new query window of new server.



Query using registered Servers:

Here we have executed query in registered server group. Below screenshot shows the execution of query we are using for analyzing the past instances to extract salary details of employee Vinay on his 100th day being run in a registered server query window.