



Tecnológico de Estudios Superiores de Ecatepec

**División de Ingeniería en Sistemas
Computacionales**

Academia en Ciencias de la Ingeniería

Materia

Administración de Base de Datos

Grupo 5701

Alumno

Campero Granados Luis Daniel

Profesor

Jiménez Alfaro Abraham Jorge

Practica 3

Practica 3

Abrimos SQL Server Management Studio. Una vez abierto empezaremos con nuestras actividades en un nuevo Query.

Recuperar información acerca de los objetos de la base de datos

Ejecute el siguiente script y revisar los resultados de cada declaración SELECT.

- SELECT * FROM sys.databases
- SELECT * FROM sys.schemas
- SELECT * FROM sys.objects
- SELECT * FROM sys.tables
- SELECT * FROM sys.columns
- SELECT * FROM sys.identity_columns
- SELECT * FROM sys.foreign_keys
- SELECT * FROM sys.foreign_key_columns
- SELECT * FROM sys.default_constraints
- SELECT * FROM sys.check_constraints
- SELECT * FROM sys.indexes
- SELECT * FROM sys.index_columns
- SELECT * FROM sys.triggers
- SELECT * FROM sys.views
- SELECT * FROM sys.procedures

SQLQuery1.sql - DESKTOP-U06197U\SQLEXPRESS.master (DESKTOP-U06197U\Alan (53)) - Microsoft SQL Server Management Studio

Object Explorer

Connect -> DESKTOP-U06197U\SQLEXPRESS (SQL Server 15.0.20)

Database

Security

Server Objects

Replication

PolyBase

Management

Event Profiler

SQLQuery1.sql - DESKTOP-U06197U\Alan (53)

```

SELECT * FROM sys.databases
SELECT * FROM sys.schemas
SELECT * FROM sys.objects
SELECT * FROM sys.tables
SELECT * FROM sys.columns
SELECT * FROM sys.identity_columns
SELECT * FROM sys.foreign_keys
SELECT * FROM sys.foreign_key_columns
SELECT * FROM sys.default_constraints
SELECT * FROM sys.check_constraints
SELECT * FROM sys.indexes
SELECT * FROM sys.index_columns
SELECT * FROM sys.triggers
SELECT * FROM sys.views
SELECT * FROM sys.procedures

```

Results

object_id	name	column_id	system_type_id	user_type_id	max_length	precision	scale	collation_name	is_nullable	isansi_padded	is_rowguidcol	is_identity	is_computed	is_filestream	is_replicated	is_non_rep_subscribed	is_merge_published	is_dts_replicated	is_msdt_blob
4	nom	4	127	127	8	19	0	NULL	0	0	0	0	0	0	0	0	0	0	0
5	s	5	56	56	4	10	0	NULL	0	0	0	0	0	0	0	0	0	0	0
6	url	6	56	56	4	10	0	NULL	0	0	0	0	0	0	0	0	0	0	0
7	url	7	52	52	2	5	0	NULL	0	0	0	0	0	0	0	0	0	0	0
8	ma	8	52	52	2	5	0	NULL	0	0	0	0	0	0	0	0	0	0	0
9	stat	9	56	56	4	10	0	NULL	0	0	0	0	0	0	0	0	0	0	0
10	offset	10	56	56	4	10	0	NULL	0	0	0	0	0	0	0	0	0	0	0
11	urlbit	11	56	56	4	10	0	NULL	0	0	0	0	0	0	0	0	0	0	0

Query executed successfully.

DESKTOP-U06197U\SQLEXPRESS - DESKTOP-U06197U\Alan - master 00:00:01 1,825 rows

SQLQuery1.sql - DESKTOP-U06197U\SQLEXPRESS.master (DESKTOP-U06197U\Alan (53)) - Microsoft SQL Server Management Studio

Object Explorer

Connect -> DESKTOP-U06197U\SQLEXPRESS (SQL Server 15.0.20)

Database

Security

Server Objects

Replication

PolyBase

Management

Event Profiler

SQLQuery1.sql - DESKTOP-U06197U\Alan (53)

```

SELECT * FROM sys.databases
SELECT * FROM sys.schemas
SELECT * FROM sys.objects
SELECT * FROM sys.tables
SELECT * FROM sys.columns
SELECT * FROM sys.identity_columns
SELECT * FROM sys.foreign_keys
SELECT * FROM sys.foreign_key_columns
SELECT * FROM sys.default_constraints
SELECT * FROM sys.check_constraints
SELECT * FROM sys.indexes
SELECT * FROM sys.index_columns
SELECT * FROM sys.triggers
SELECT * FROM sys.views
SELECT * FROM sys.procedures

```

Results

object_id	name	index_id	type	file_desc	is_unique	data_space_id	group_order_id	is_primary_key	is_unique_constraint	is_factor	is_padded	is_disabled	is_hypothetical	is_ignored_in_optimization	allow_row_locks	allow_page_locks	has_filter	filter_definition	copy_on_sustain
7	nom	1	1	CLUSTERED	1	1	0	0	0	0	0	0	0	0	1	1	0	NULL	NULL
8	s	1	1	CLUSTERED	1	1	0	0	0	0	0	0	0	0	1	1	0	NULL	NULL
9	url	1	1	CLUSTERED	1	1	0	0	0	0	0	0	0	0	1	1	0	NULL	NULL
10	url	2	2	NONCLUSTERED	1	1	0	0	0	0	0	0	0	0	1	1	0	NULL	NULL
11	url	3	2	NONCLUSTERED	1	1	0	0	0	0	0	0	0	0	1	1	0	NULL	NULL
12	ma	1	1	CLUSTERED	1	1	0	0	0	0	0	0	0	0	1	1	0	NULL	NULL
13	stat	2	2	NONCLUSTERED	1	1	0	0	0	0	0	0	0	0	1	1	0	NULL	NULL
14	urlbit	1	1	CLUSTERED	1	1	0	0	0	0	0	0	0	0	1	1	0	NULL	NULL

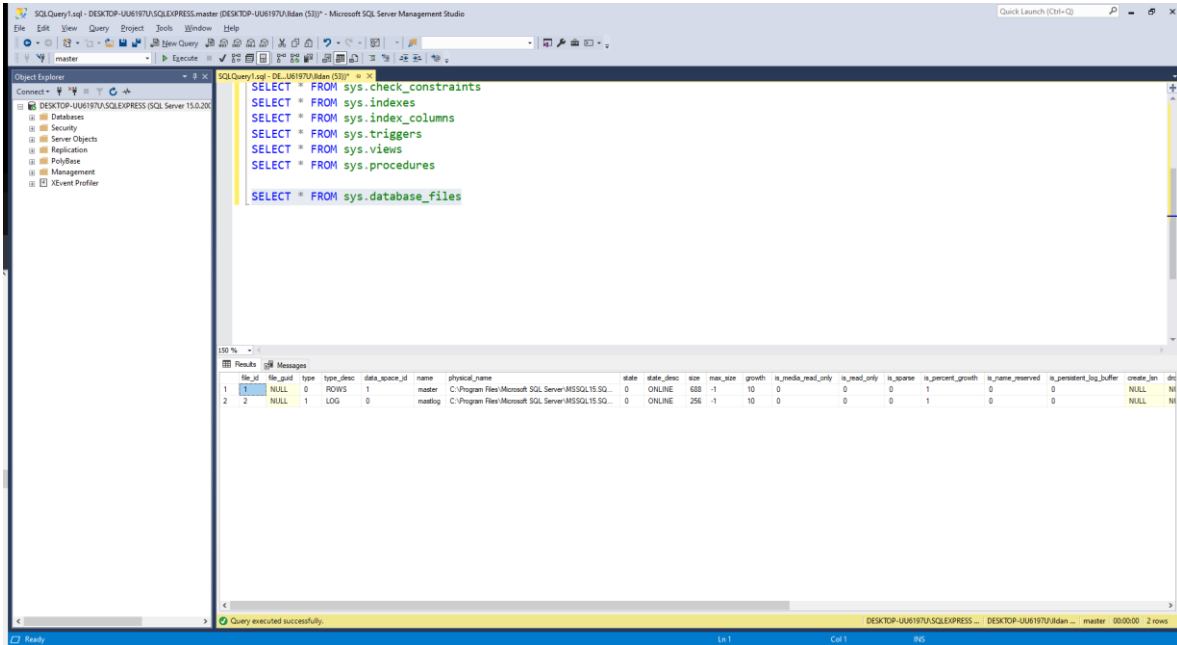
Query executed successfully.

DESKTOP-U06197U\SQLEXPRESS - DESKTOP-U06197U\Alan - master 00:00:01 1,825 rows

Recuperar Tamaños de bases de datos y objetos

Ejecutar la siguiente secuencia de comandos y revise los resultados de cada declaración

- `SELECT * FROM sys.database_files`



Ejecute el siguiente script y revise los resultados de cada declaración SELECT

- `SELECT * FROM sys.partitions`
- `SELECT * FROM sys.allocation_units`
- `SELECT object_name(a.object_id), c.name, SUM(rows) rows,`
- `SUM(total_pages) total_pages, SUM(used_pages) used_pages,`
- `SUM(data_pages) data_pages`
- `FROM sys.partitions a INNER JOIN sys.allocation_units b ON`
- `a.hobt_id =`
- `b.container_id`
- `INNER JOIN sys.indexes c ON a.object_id = c.object_id and`
- `a.index_id =`
- `c.index_id`
- `GROUP BY object_name(a.object_id), c.name`

- ORDER BY object_name(a.object_id), c.name

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The Query Editor window displays a SQL query that joins system tables to retrieve information about database files, partitions, allocation units, and indexes. The Results pane shows the output of the query, which is divided into two sections. The first section lists database files with columns like partition_id, object_id, index_id, partition_number, hobt_id, rows, filestream_flag, data_compression, and data_compression_desc. The second section lists allocation units with columns like allocation_unit_id, type, type_desc, container_id, data_space_id, total_pages, used_pages, and data_pages. The status bar at the bottom indicates that the query was executed successfully and returned 530 rows.

```

SELECT * FROM sys.database_files

SELECT * FROM sys.partitions
SELECT * FROM sys.allocation_units
SELECT object_name(a.object_id), c.name, SUM(rows) rows,
SUM(total_pages) total_pages, SUM(used_pages) used_pages,
SUM(data_pages) data_pages
FROM sys.partitions a INNER JOIN sys.allocation_units b ON a.hobt_id =
b.container_id
INNER JOIN sys.indexes c ON a.object_id = c.object_id and a.index_id =
c.index_id
GROUP BY object_name(a.object_id), c.name
ORDER BY object_name(a.object_id), c.name

```

partition_id	object_id	index_id	partition_number	hobt_id	rows	filestream_flag	data_compression	data_compression_desc
1	196608	3	1	196608	1339	0	0	NONE
2	327680	5	1	327680	173	0	0	NONE
3	458752	7	1	458752	204	0	0	NONE
4	524288	8	0	524288	2	0	0	NONE
5	7705425149952	117675467	0	7705425149952	0	0	0	NONE
6	875404889504	132675514	0	875404889504	0	0	0	NONE
7	9802594621056	149575571	0	9802594621056	0	0	0	NONE
8	28147877103872	6	1	28147877103872	0	0	0	NONE

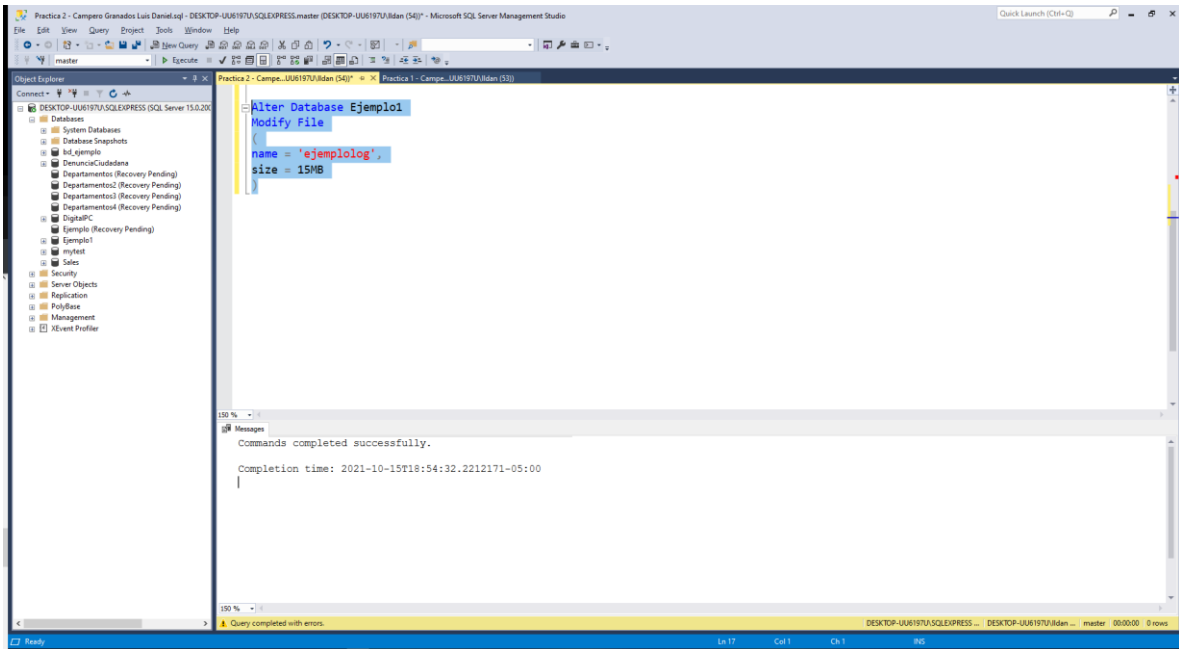
allocation_unit_id	type	type_desc	container_id	data_space_id	total_pages	used_pages	data_pages
1	IN_ROW_DATA	IN_ROW_DATA	196608	1	25	20	18
2	IN_ROW_DATA	IN_ROW_DATA	327680	1	4	4	2
3	IN_ROW_DATA	IN_ROW_DATA	458752	1	5	5	3
4	IN_ROW_DATA	IN_ROW_DATA	524288	1	2	2	1
5	IN_ROW_DATA	IN_ROW_DATA	7705425149952	1	0	0	0
6	IN_ROW_DATA	IN_ROW_DATA	875404889504	1	0	0	0
7	IN_ROW_DATA	IN_ROW_DATA	9802594621056	1	0	0	0
8	IN_ROW_DATA	IN_ROW_DATA	28147877103872	1	0	0	0

(No column name)	name	rows	total_pages	used_pages	data_pages
1	_tbl_trusted_assemblies	0	0	0	0

Indices.

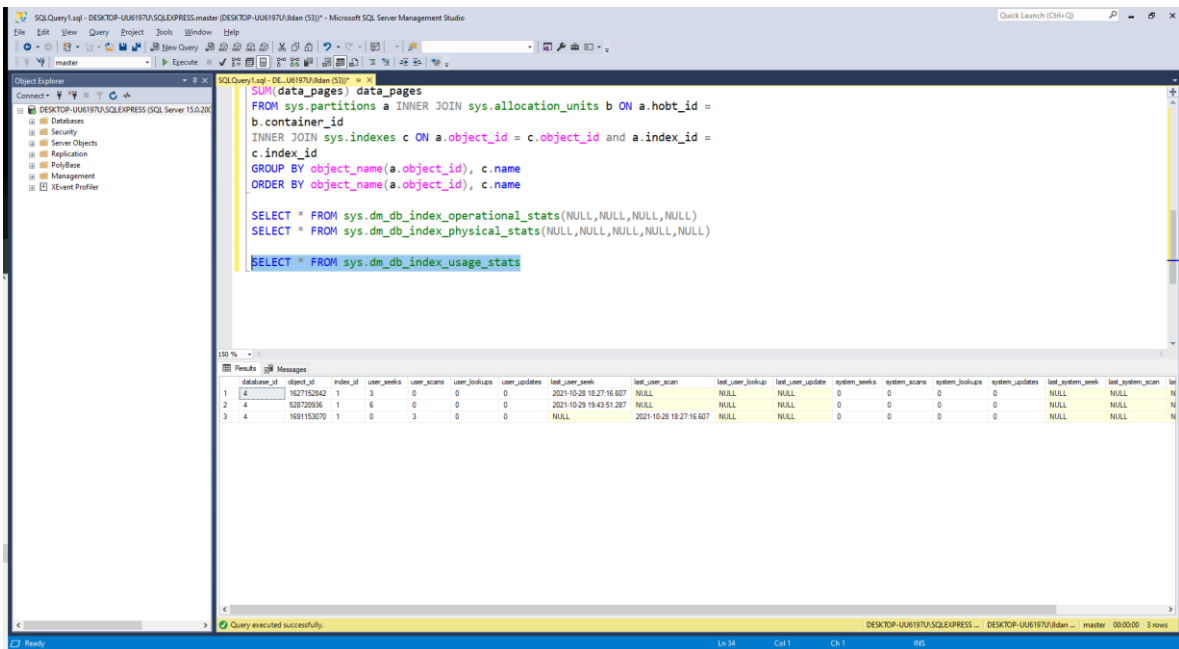
Ejecute la siguiente secuencia de comandos y revise los resultados

- SELECT * FROM sys.dm_db_index_operational_stats(NULL,NULL,NULL,NULL)
- SELECT * FROM sys.dm_db_index_physical_stats(NULL,NULL,NULL,NULL,NULL)



Ejecute la siguiente secuencia de comandos y revise los resultados Alter Database Ejemplo

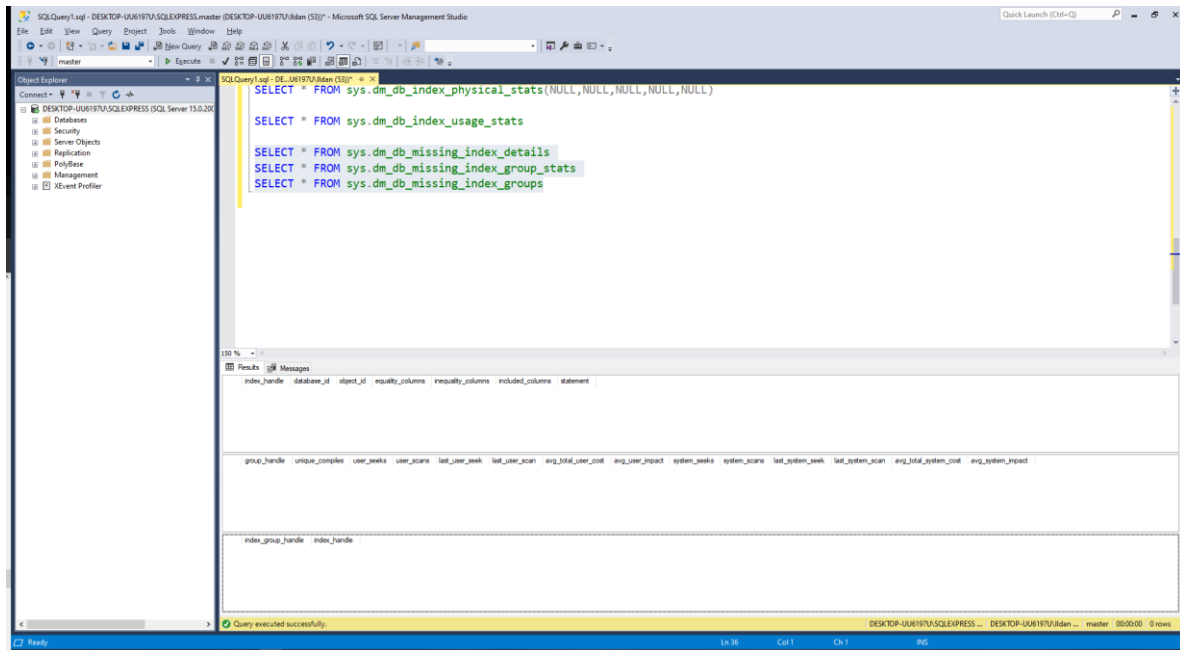
- SELECT * FROM sys.dm_db_index_usage_stats



Determinar los índices a crear

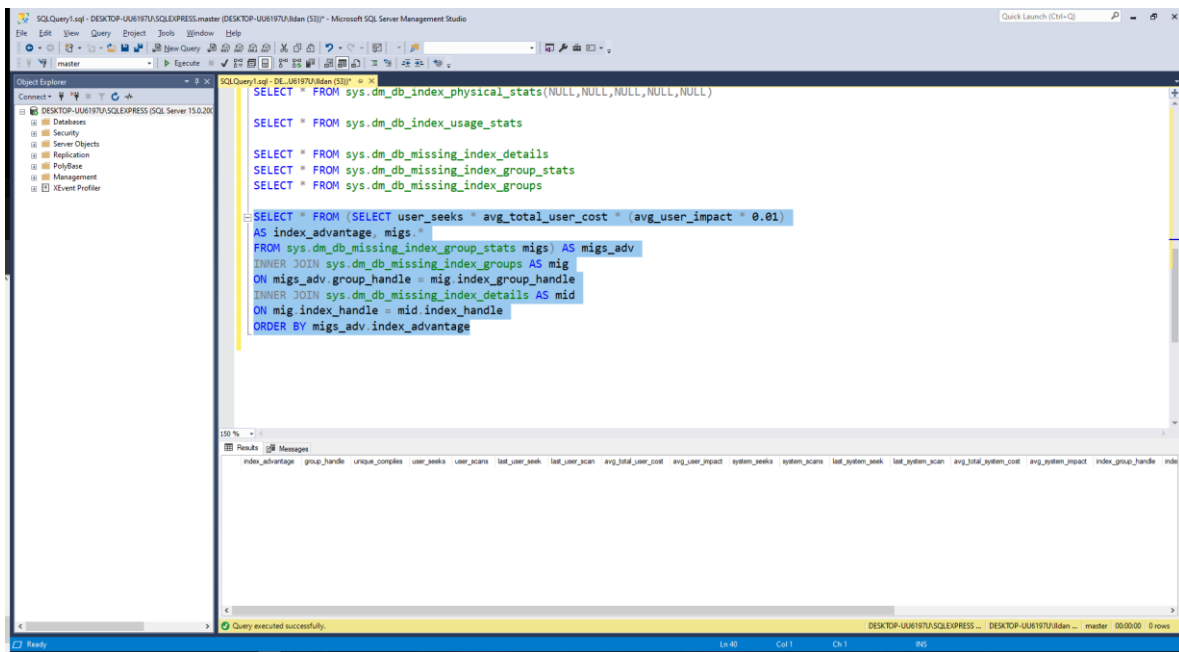
Ejecute la siguiente secuencia de comandos y revise los resultados

- `SELECT * FROM sys.dm_db_missing_index_details`
- `SELECT * FROM sys.dm_db_missing_index_group_stats`
- `SELECT * FROM sys.dm_db_missing_index_groups`



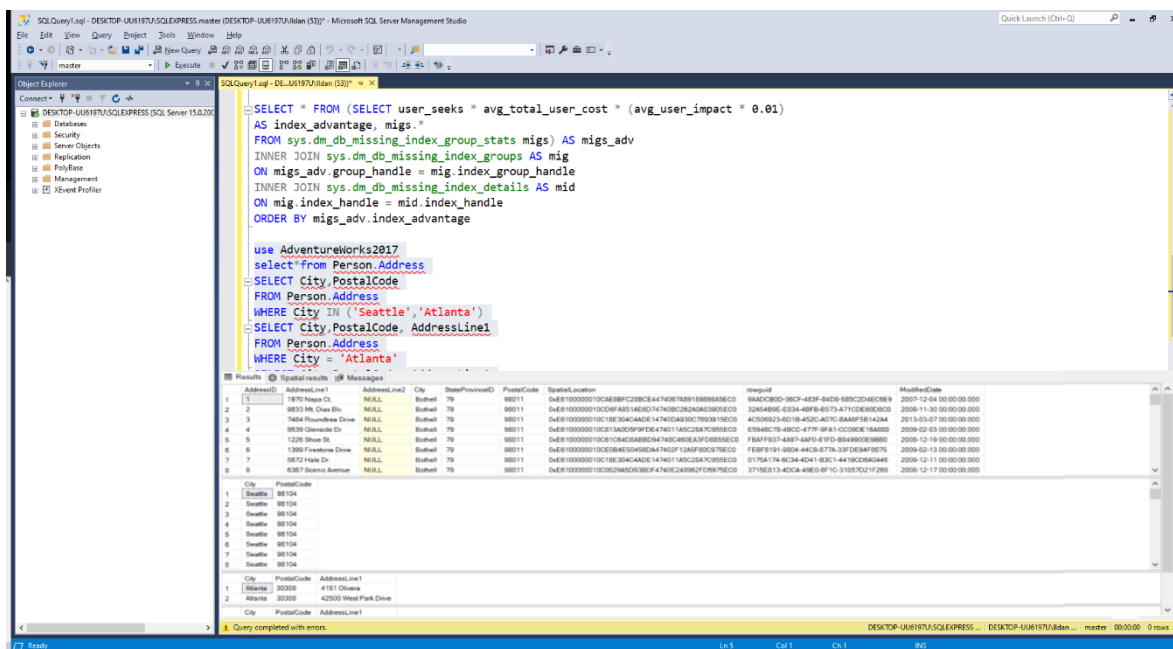
Ejecute el siguiente script de agregación y revise los resultados

- `SELECT * FROM (SELECT user_seeks * avg_total_user_cost * (avg_user_impact * 0.01)`
- `AS index_advantage, migs.*`
- `FROM sys.dm_db_missing_index_group_stats migs) AS migs_adv`
- `INNER JOIN sys.dm_db_missing_index_groups AS mig`
- `ON migs_adv.group_handle = mig.index_group_handle`
- `INNER JOIN sys.dm_db_missing_index_details AS mid`
- `ON mig.index_handle = mid.index_handle`
- `ORDER BY migs_adv.index_advantage`



Ejecute el siguiente código para forzar un índice perdido contra la base de datos AdventureWorks

- use AdventureWorks2017
- select*from Person.Address
- SELECT City,PostalCode
- FROM Person.Address
- WHERE City IN ('Seattle','Atlanta')
- SELECT City,PostalCode, AddressLine1
- FROM Person.Address
- WHERE City = 'Atlanta'
- SELECT City,PostalCode, AddressLine1
- FROM Person.Address
- WHERE City like 'Atlan%'



Ejecutar el script de agregación de nuevo y examine los resultados

- `SELECT*FROM (SELECT user_seeks * avg_total_user_cost * (avg_user_impact * 0.01)`
- `AS index_advantage,migs.*`
- `FROM sys.dm_db_missing_index_group_stats migs) AS migs_adv`
- `INNER JOIN sys.dm_db_missing_index_groups AS mig`
- `ON migs_adv.group_handle = mig.index_group_handle`
- `INNER JOIN sys.dm_db_missing_index_details AS mid`
- `ON mig.index_handle = mid.index_handle`
- `ORDER BY migs_adv.index_advantage`
- `select*from Person.CountryRegion`

The screenshot shows the SQL Server Enterprise Manager interface. The query editor displays the following SQL script:

```

SELECT * FROM (SELECT user_seeks * avg_total_user_cost * (avg_user_impact * 0.01)
AS index_advantage, migs.*
FROM sys.dm_db_missing_index_group_stats migs) AS migs_adv
INNER JOIN sys.dm_db_missing_index_groups AS mig
ON migs_adv.group_handle = mig.index_group_handle
INNER JOIN sys.dm_db_missing_index_details AS mid
ON mig.index_handle = mid.index_handle
ORDER BY migs_adv.index_advantage

select*from Person.CountryRegion
  
```

The Results pane shows the following data:

index_advantage	group_handle	unique_compiles	user_seeks	user_scans	last_user_seek	last_user_scan	avg_total_user_cost	avg_user_impact	system_seeks	system_scans	last_system_seek	last_system_scan	avg_total_system_cost	avg_system_cost
0.1987762338488	6	1	1	0	2021-10-28 14:07:58.983	NULL	0.197673275555556	85.33	0	0	NULL	NULL	0	0
0.181953714862857	2	1	0	0	2021-10-28 14:07:58.973	NULL	0.197673275555556	81.93	0	0	NULL	NULL	0	0
0.185955190574222	4	1	1	0	2021-10-28 14:07:58.980	NULL	0.198627675555556	97.96	0	0	NULL	NULL	0	0

The bottom pane shows the results of the `select*from Person.CountryRegion` query, displaying a list of countries and regions with their codes and names.

Ejecute el siguiente script para ejecutar repetidamente una declaración SELECT y revise los nuevos resultados de la secuencia de comandos de agregación.

- SELECT City,PostalCode,AddressLine1
- FROM Person.Address
- WHERE City like 'Atlan%'
- GO 100
- SELECT*
- FROM (SELECT user_seeks * avg_total_user_cost *
(avg_user_impact * 0.01)
- AS index_advantage,migs.*
- FROM sys.dm_db_missing_index_group_stats migs) AS
migs_adv
- INNER JOIN sys.dm_db_missing_index_groups AS mig
- ON migs_adv.group_handle = mig.index_group_handle
- INNER JOIN sys.dm_db_missing_index_details AS mid
- ON mig.index_handle = mid.index_handle
- ORDER BY migs_adv.index_advantage
- select*from Person.StateProvince

