

## CST2355 – Database Systems      Lab Assignment 5

Student Name:      Fei Lan  
Student ID:        041055048  
Student email:     lan00012@algonquinlive.com

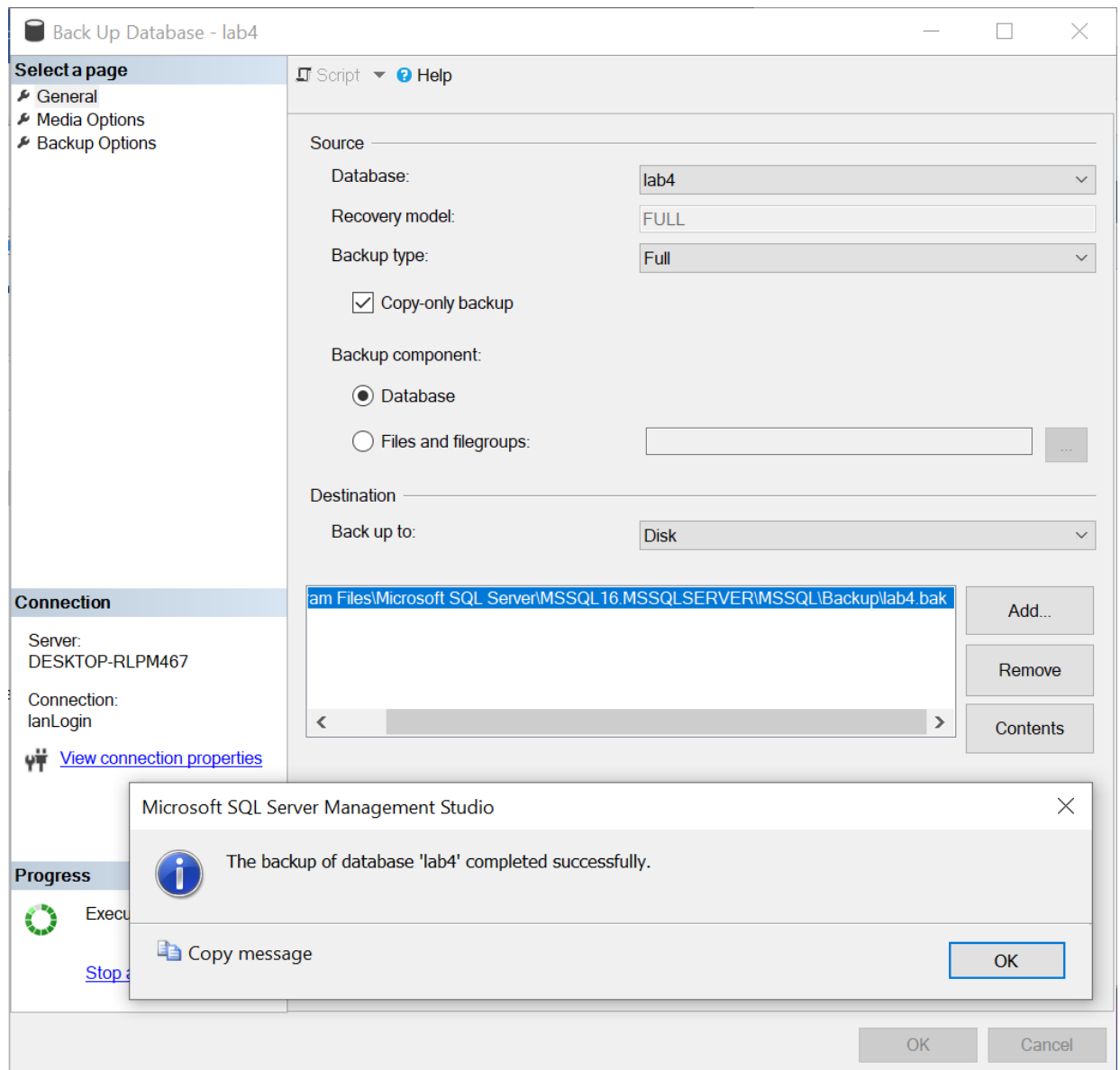
## Hand-in:

1. The lab assignment will be graded out of a maximum 4 points.
2. This template should be used to submit your lab assignment.
3. Make sure you have enough screenshots to completely document that you have completed all the steps.

## Activities (Steps):

In this lab, you will be using the Microsoft SQL Server Management Studio, building functions, stored procedures, and triggers.

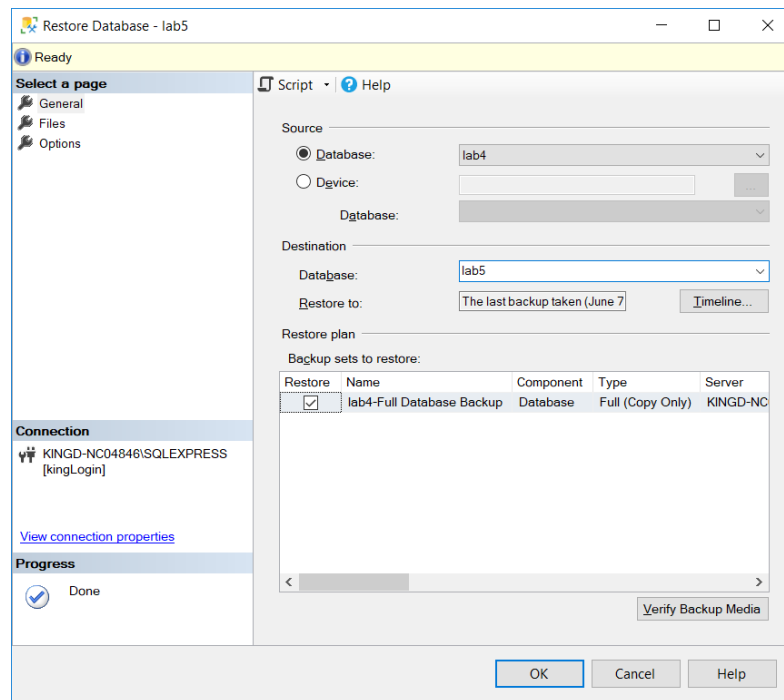
1. First you will use SQL Server Management Studio to create a new database that is owned by your own personal account by performing the following tasks:
  - 1.1. Create a new database named “lab5”, owned by *yourlastnameLogin*, by following these steps:
    - 1.1.1. Connect using your Windows login account (so that you have all the required privileges).
    - 1.1.2. Create a backup of the “lab4” database.
      - 1.1.2.1. Select lab4 > Tasks > Back Up ...
      - 1.1.2.2. Make sure you select the “Copy Only” option.



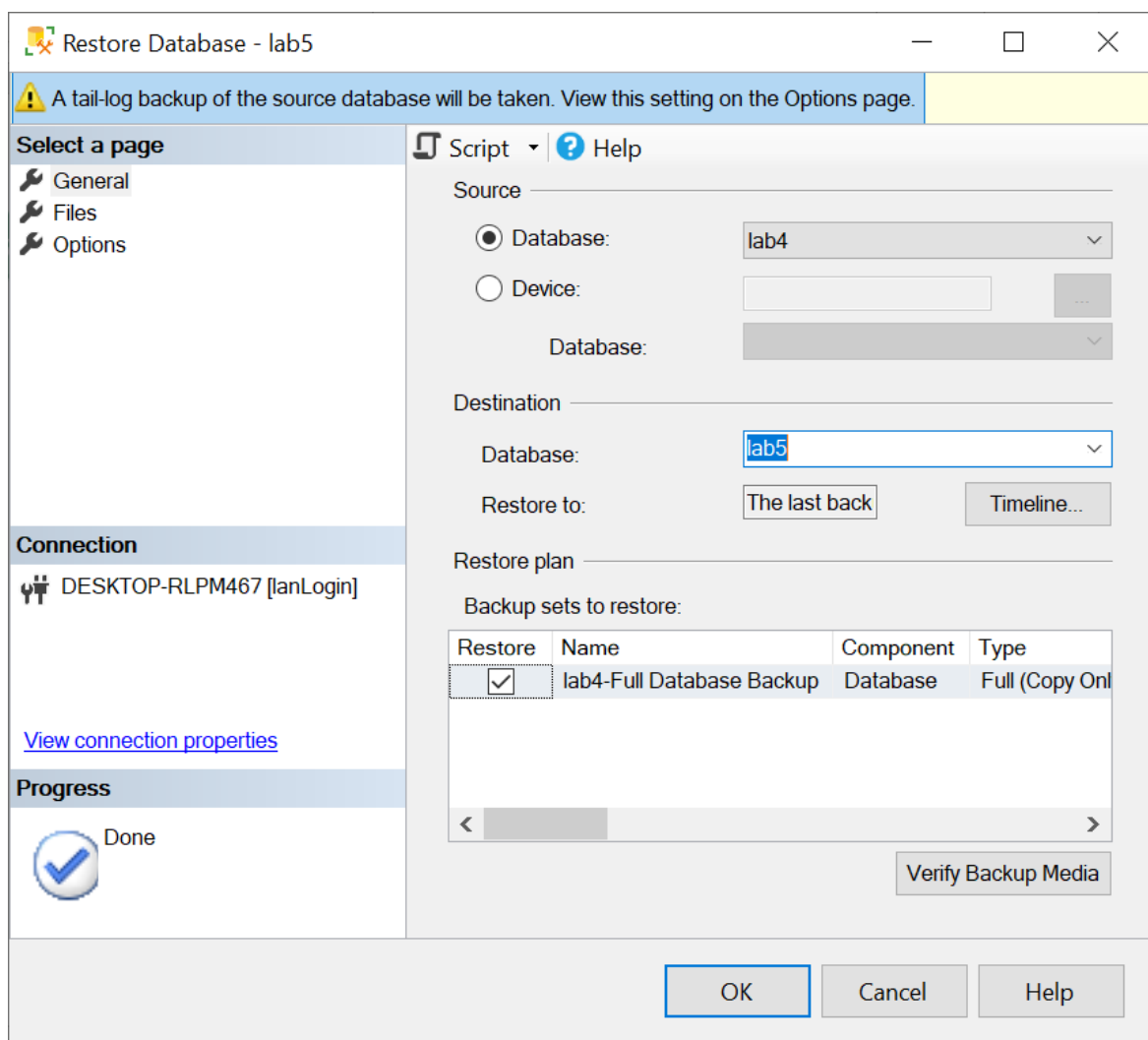
1.1.3. Restore the backup to a new database called “lab5”.

1.1.3.1. Select Databases > Restore Database ...

1.1.3.2. Provide the source database as “lab4”. Then a list of backups will appear. Select the destination as “lab5”. You should have something similar to the following:



Paste your screenshot here:

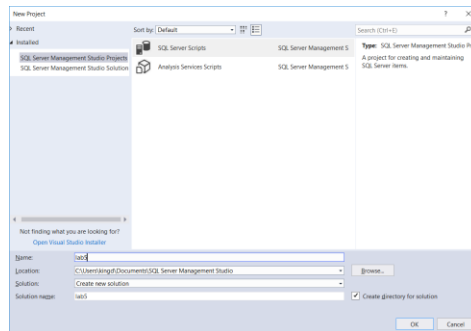


1.1.3.3. Then click OK to create the new database (including all constraints, etc.)

1.1.3.4. Once restored, you will also need to change owner (by executing the following query script – you need to change the login name...not likely kingLogin!)

```
USE lab5;
EXEC sp_changedbowner 'kingLogin', 'true';
```

2. Now, create a new project for your scripts. Use File > New > Project. Make sure the “Create Directory” option is selected. Here is mine:



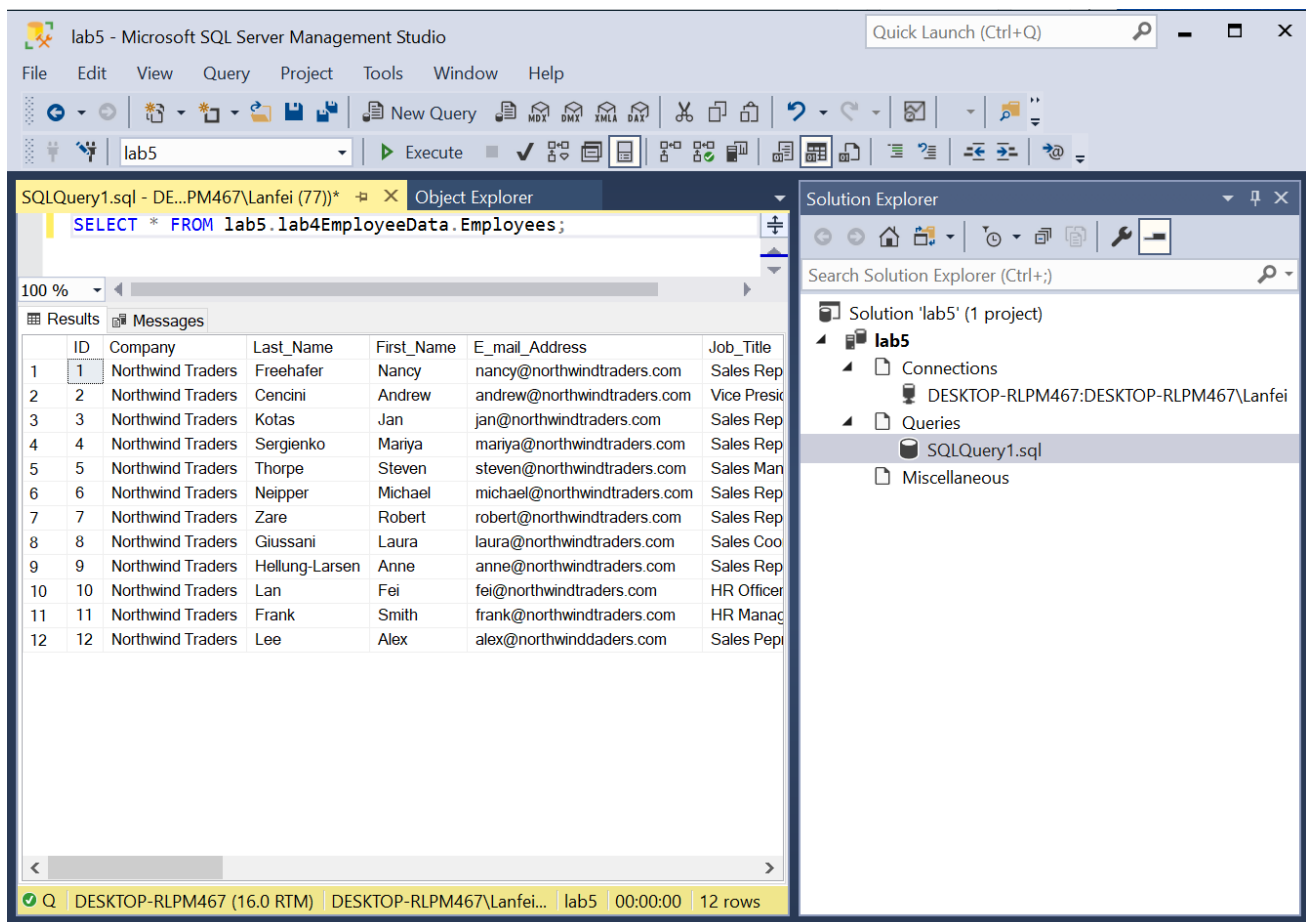
3. .You will now create a user-defined function and use it.

3.1. . Create a new query using Project > New Query

3.1.1.1. Try executing the following query:

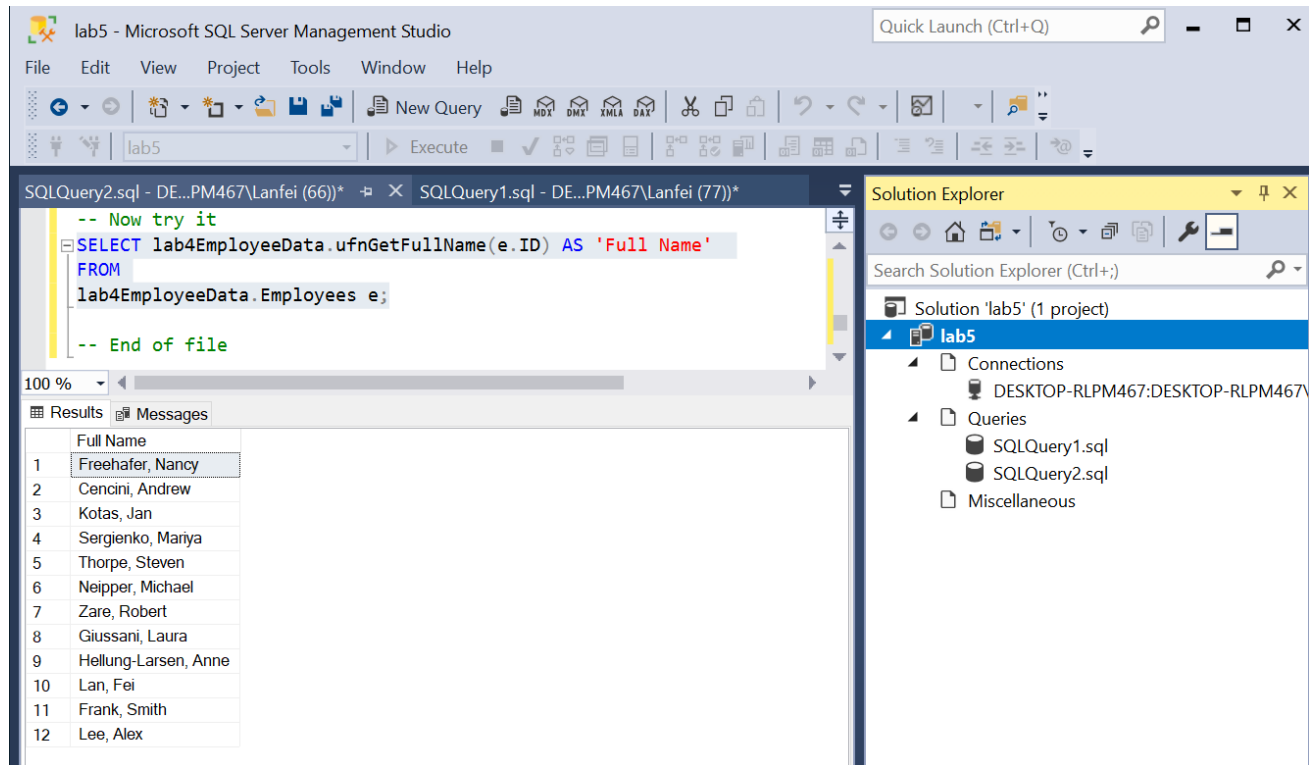
```
SELECT * FROM lab5.lab4EmployeeData.Employees;
```

Provide a screenshot:



3.2. Use the function definition example provided in lab5ExampleFunction.sql (it is almost ready to use...) and run it in SSMS.

3.3. Paste a screenshot showing it working to provide the employees full name.



4. You will now create a stored procedure and use it.

4.1. . Create a new query using Project > New Query

4.1.1. Now try the following (which only retrieves Hobby data for employees with NULL for city) You can run the commands individually by highlighting them in SSMS. (Make sure that you have at least one employee with a NULL City field for testing.)

```
-- returns firstname, lastname, hobby for matching employees
-- with the added restriction that the city field must be NULL
USE lab5;
GO
CREATE PROCEDURE lab4EmployeeData.getAlienData
    @LastName nvarchar(50),
    @FirstName nvarchar(50)
AS
SET NOCOUNT ON;
SELECT lab4EmployeeData.ufnGetFullName(e.ID) AS 'Full Name' , Hobby
FROM lab4EmployeeData.Employees e
LEFT JOIN lab4EmployeeData.EmployeeHobby
ON e.ID = lab4EmployeeData.EmployeeHobby.EmployeeID
```

```

LEFT JOIN lab4EmployeeData.Hobbies
ON lab4EmployeeData.EmployeeHobby.HobbyID =
lab4EmployeeData.Hobbies.HobbyID
WHERE e.[First Name] = @FirstName AND e.[Last Name] = @LastName
AND City IS NULL;

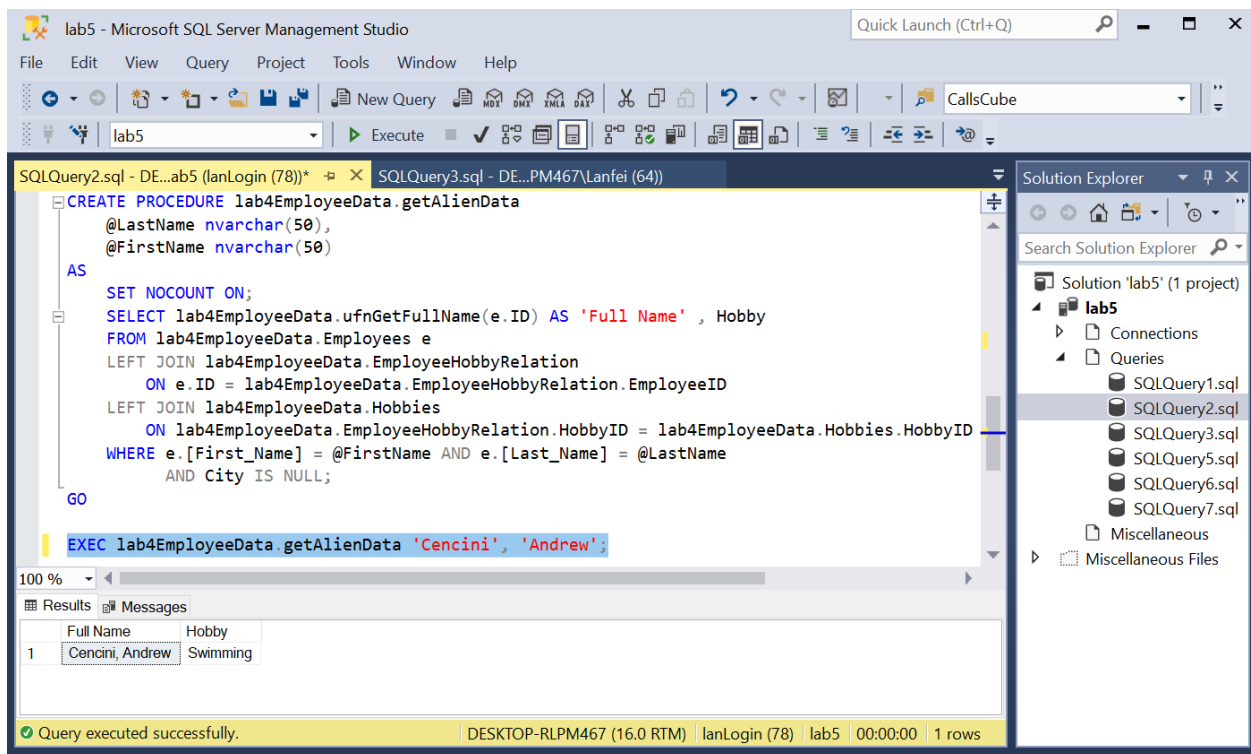
GO

```

4.1.2. Try using the stored procedure.

```
EXEC lab4EmployeeData.getAlienData 'King', 'Douglas';
```

4.1.3. Provide a screenshot of your stored procedure working on your data:



4.2. Now you will build a “real” stored procedure that creates a new table from data that has been imported into the database. This is typically one of the last steps of the workflow in an Extract-Transform-Load scenario.

4.2.1. Build a “cube” (note: this is not the same as a cube in the SQL Server Analysis tools...) of data for your `lab4EmployeeData` schema as a stored procedure. Whenever it needs to be re-built, the stored procedure can be executed.

4.2.1.1. Modify the example provided in the “`lab5ExampleCalls.sql`” file to create one or more queries that will create the `CubeData` table and then build the “cube” for your data called “`CallsCube`”. Your output table should use joined-in data to show the “Calls” totals for employees by name in each named department:

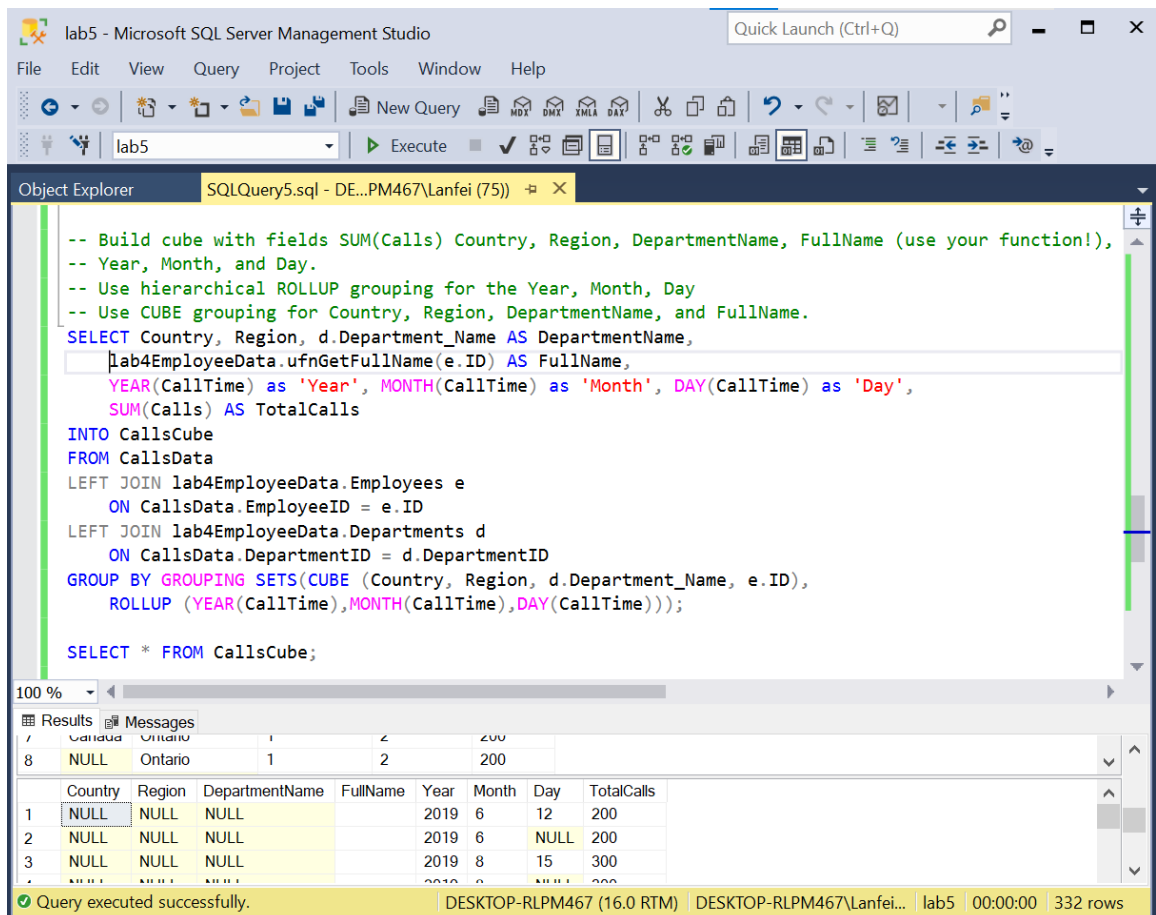
4.2.1.1.1. Include the following fields in your cube, **in addition to the SUM(Calls) field:**

Country, Region, DepartmentName, FullName (use your function!), Year, Month, and Day

4.2.1.1.2. Use hierarchical ROLLUP grouping for the Year, Month, Day.

4.2.1.1.3. Use CUBE grouping for Country, Region, DepartmentName, and FullName

4.2.1.2. Provide screenshot(s) that show the SQL: how you built the “cube”.



The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor displays the following SQL code:

```
-- Build cube with fields SUM(Calls) Country, Region, DepartmentName, FullName (use your function!),
-- Year, Month, and Day.
-- Use hierarchical ROLLUP grouping for the Year, Month, Day
-- Use CUBE grouping for Country, Region, DepartmentName, and FullName.
SELECT Country, Region, d.Department_Name AS DepartmentName,
       lab4EmployeeData.ufnGetFullName(e.ID) AS FullName,
       YEAR(CallTime) as 'Year', MONTH(CallTime) as 'Month', DAY(CallTime) as 'Day',
       SUM(Calls) AS TotalCalls
INTO CallsCube
FROM CallsData
LEFT JOIN lab4EmployeeData.Employees e
  ON CallsData.EmployeeID = e.ID
LEFT JOIN lab4EmployeeData.Departments d
  ON CallsData.DepartmentID = d.DepartmentID
GROUP BY GROUPING SETS(CUBE (Country, Region, d.Department_Name, e.ID),
  ROLLUP (YEAR(CallTime),MONTH(CallTime),DAY(CallTime)));

SELECT * FROM CallsCube;
```

The Results pane shows the following data:

Country	Region	DepartmentName	FullName	Year	Month	Day	TotalCalls
1	NULL	NULL	NULL	2019	6	12	200
2	NULL	NULL	NULL	2019	6	NULL	200
3	NULL	NULL	NULL	2019	8	15	300

The status bar at the bottom indicates: Query executed successfully. DESKTOP-RLPM467 (16.0 RTM) DESKTOP-RLPM467\Lanfei... lab5 00:00:00 332 rows

4.2.1.3. Provide a screenshot showing the resulting cube data (“SELECT \* FROM CallsCube;”)

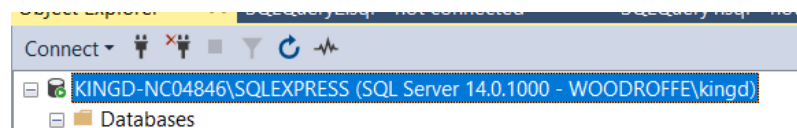


The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The title bar indicates the connection is 'lab5 - Microsoft SQL Server Management Studio'. The menu bar includes File, Edit, View, Query, Project, Tools, Window, and Help. The toolbar contains various icons for file operations, query execution, and formatting. The Object Explorer on the left shows the 'lab5' database selected. The query editor displays the SQL statement: `SELECT * FROM CallsCube;`. The Results pane shows a table with 24 rows and 9 columns: Country, Region, DepartmentName, FullName, Year, Month, Day, and TotalCalls. The data is as follows:

	Country	Region	DepartmentName	FullName	Year	Month	Day	TotalCalls
1	NULL	NULL	NULL		2019	6	12	200
2	NULL	NULL	NULL		2019	6	NULL	200
3	NULL	NULL	NULL		2019	8	15	300
4	NULL	NULL	NULL		2019	8	NULL	300
5	NULL	NULL	NULL		2019	11	15	100
6	NULL	NULL	NULL		2019	11	NULL	100
7	NULL	NULL	NULL		2019	NULL	NULL	600
8	NULL	NULL	NULL		2020	4	15	200
9	NULL	NULL	NULL		2020	4	NULL	200
10	NULL	NULL	NULL		2020	5	15	300
11	NULL	NULL	NULL		2020	5	NULL	300
12	NULL	NULL	NULL		2020	7	15	100
13	NULL	NULL	NULL		2020	7	NULL	100
14	NULL	NULL	NULL		2020	8	15	100
15	NULL	NULL	NULL		2020	8	NULL	100
16	NULL	NULL	NULL		2020	NULL	NULL	700
17	NULL	NULL	NULL		2021	8	15	100
18	NULL	NULL	NULL		2021	8	NULL	100
19	NULL	NULL	NULL		2021	NULL	NULL	100
20	NULL	NULL	NULL		NU...	NULL	NULL	1400
21	Canada	Britis...	Human Resource	Freehaf...	NU...	NULL	NULL	300
22	NULL	Britis...	Human Resource	Freehaf...	NU...	NULL	NULL	300
23	NULL	NULL	Human Resource	Freehaf...	NU...	NULL	NULL	300
24	NULL	NULL	NULL	Freehaf...	NU...	NULL	NULL	300

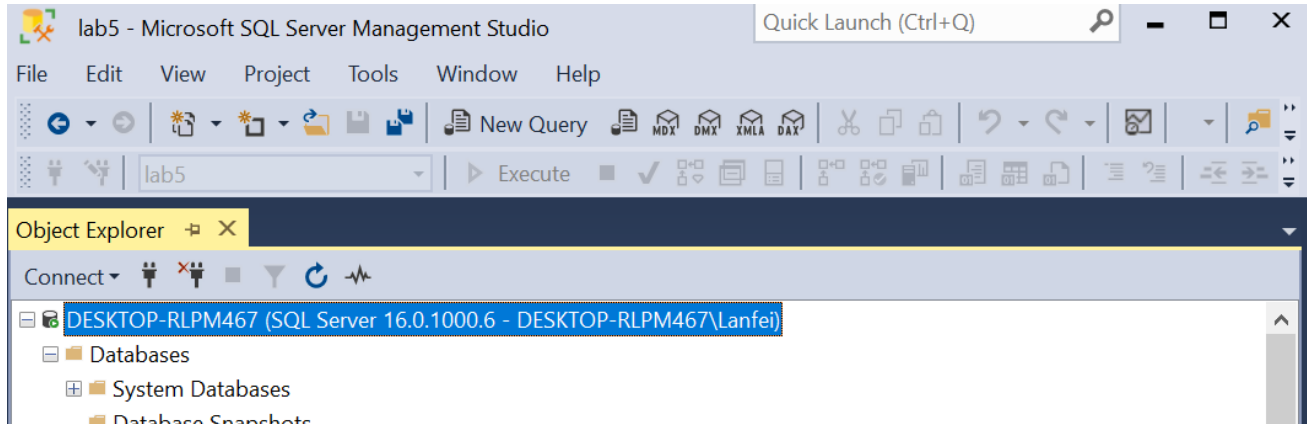
5. Now we will look at an actual data warehouse example. Download the AdventureWorks **datawarehouse** from Microsoft by using the following steps:

5.1. First, determine the version of SQL server you are connected to in SSMS. If you look at the connection information (here is mine):



5.2. It will show if it is a version 14 or 15 server. If it is a version 14 (like mine) then you can use the AdventureWorksDW2017 and AdventureWorks2017 databases. If it is a version 15, then you can use the AdventureWorksDW2019 and AdventureWorks2019 databases.

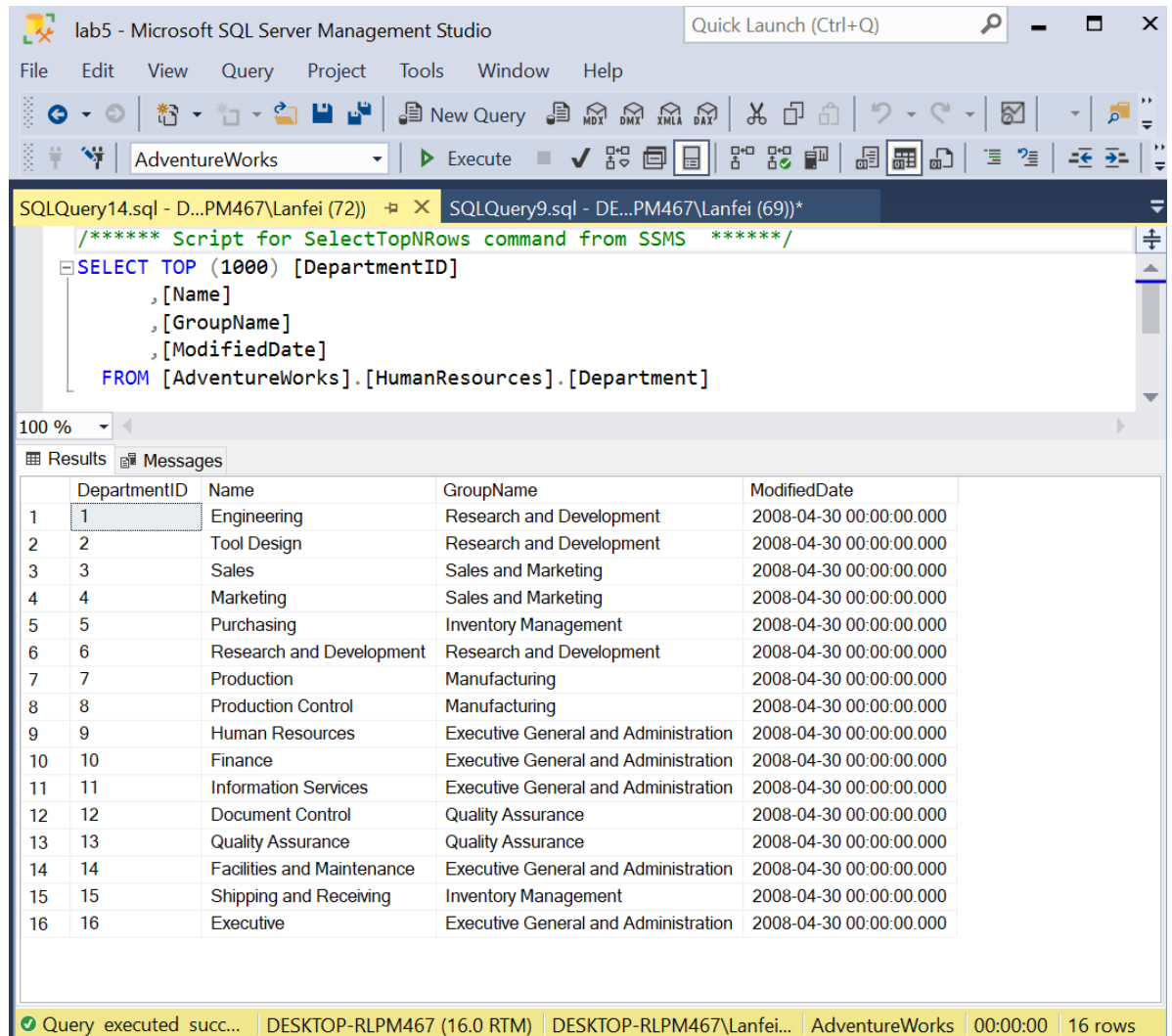
Provide a screenshot of your version here:



- 5.3. Navigate to the Microsoft site that has backups stored for various versions of the AdventureWorks example database: <https://docs.microsoft.com/en-us/sql/samples/adventureworks-install-configure?view=sql-server-ver15&tabs=ssms>
- 5.4. Download the AdventureWorksDW2017.bak or AdventureWorksDW2019.bak backup file. You are going to restore this to your SQL Server 2019 server.
- 5.5. Download the AdventureWorks2017.bak or AdventureWorks2019.bak backup file. You are going to restore this to your SQL Server 2019 server.
- 5.6. You need to restore and set the new owner for the AdventureWorksDW2017 database
  - 5.6.1. Restore: the first part of the following video shows the installation steps in detail:  
<https://www.youtube.com/watch?v=90DVrlzl8bs> is a nice tutorial
  - 5.6.2. The important step (in the video) is finding the default directory where backups are stored. Once you find it, you should copy both of the AdventureWorks backup files to that directory and restore the databases using your Windows login connection.
  - 5.6.3. Once restored, you will also need to change owner (by executing two scripts similar to the following query script – you need to change the login name...not likely kingLogin!)
 

```
USE AdventureWorksDW2017;
EXEC sp_changedbowner 'kingLogin', 'true';
```
  - 5.6.4. The databases should be available now. Try a simple query on each database to show they are installed correctly:

AdventureWorks screenshot of query:



lab5 - Microsoft SQL Server Management Studio

File Edit View Query Project Tools Window Help

AdventureWorks Execute

SQLQuery14.sql - D...PM467\Lanfei (72) SQLQuery9.sql - DE...PM467\Lanfei (69)\*

```

/***** Script for SelectTopNRRows command from SSMS *****/
SELECT TOP (1000) [DepartmentID]
, [Name]
, [GroupName]
, [ModifiedDate]
FROM [AdventureWorks].[HumanResources].[Department]
  
```

100 %

Results Messages

	DepartmentID	Name	GroupName	ModifiedDate
1	1	Engineering	Research and Development	2008-04-30 00:00:00.000
2	2	Tool Design	Research and Development	2008-04-30 00:00:00.000
3	3	Sales	Sales and Marketing	2008-04-30 00:00:00.000
4	4	Marketing	Sales and Marketing	2008-04-30 00:00:00.000
5	5	Purchasing	Inventory Management	2008-04-30 00:00:00.000
6	6	Research and Development	Research and Development	2008-04-30 00:00:00.000
7	7	Production	Manufacturing	2008-04-30 00:00:00.000
8	8	Production Control	Manufacturing	2008-04-30 00:00:00.000
9	9	Human Resources	Executive General and Administration	2008-04-30 00:00:00.000
10	10	Finance	Executive General and Administration	2008-04-30 00:00:00.000
11	11	Information Services	Executive General and Administration	2008-04-30 00:00:00.000
12	12	Document Control	Quality Assurance	2008-04-30 00:00:00.000
13	13	Quality Assurance	Quality Assurance	2008-04-30 00:00:00.000
14	14	Facilities and Maintenance	Executive General and Administration	2008-04-30 00:00:00.000
15	15	Shipping and Receiving	Inventory Management	2008-04-30 00:00:00.000
16	16	Executive	Executive General and Administration	2008-04-30 00:00:00.000

Query executed succ... DESKTOP-RLPM467 (16.0 RTM) DESKTOP-RLPM467\Lanfei... AdventureWorks 00:00:00 16 rows

AdventureWorksDW screenshot of query:

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The title bar indicates the connection is to 'lab5 - Microsoft SQL Server Management Studio'. The menu bar includes File, Edit, View, Query, Project, Tools, Window, and Help. The toolbar contains icons for New Query, Open, Save, Execute, and other standard database operations. The Object Explorer on the left shows the 'AdventureWorksDW' database selected. The main query window displays the following SQL script:

```

/***** Script for SelectTopNRows command from SSMS *****/
SELECT TOP (1000) [AccountKey]
, [ParentAccountKey]
, [AccountCodeAlternateKey]
, [ParentAccountCodeAlternateKey]
, [AccountDescription]
, [AccountType]
, [Operator]
, [CustomMembers]
, [ValueType]
, [CustomMemberOptions]
FROM [AdventureWorksDW].[dbo].[DimAccount]
  
```

The Results pane at the bottom shows the output of the query, displaying 99 rows. The first 12 rows are visible in the table below:

	AccountKey	ParentAccountKey	AccountCodeAlternateKey	ParentAccountCodeAlternateKey	AccountDes
1	1	NULL	1	NULL	Balance She
2	2	1	10	1	Assets
3	3	2	110	10	Current Ass
4	4	3	1110	110	Cash
5	5	3	1120	110	Receivables
6	6	5	1130	1120	Trade Recei
7	7	5	1140	1120	Other Recei
8	8	3	1150	110	Allowance fo
9	9	3	1160	110	Inventory
10	10	9	1162	1160	Raw Materiz
11	11	9	1164	1160	Work in Proc
12	12	9	1166	1160	Finished Go

The status bar at the bottom indicates the connection is to 'DESKTOP-RLPM467 (16.0 RTM)' and shows the query executed successfully, returning 99 rows.

6. Using SSMS, create a query on your AdventureWorksDW database that provides the following output:

6.1. A list of data showing the birth year numbers for the customers in each city:

6.1.1. The output should have a format similar to the following:

Birth Year	City	Number of Customers
1960	Ottawa	5

1961	Ottawa	1
...		
1981	Toronto	3
1983	Toronto	4
...		

6.2. STRONG HINT: The source of the data should be the table “DimGeography” linked with the associated table “DimCustomer”

6.3. Run the query and provide a screenshot:

The screenshot shows the Microsoft SQL Server Management Studio interface. The query editor displays the following SQL query:

```
USE AdventureWorksDW;

SELECT YEAR(c.BirthDate) AS 'Birth Year', g.City, count(*) AS 'Number of Customers'
FROM DimCustomer c
JOIN DimGeography g
    ON c.GeographyKey = g.GeographyKey
GROUP BY YEAR(c.BirthDate), g.City
ORDER BY g.City, YEAR(c.BirthDate);
```

The Results pane shows the following data:

	Birth Year	City	Number of Customers
1	1945	Ballard	1
2	1947	Ballard	2
3	1948	Ballard	1
4	1951	Ballard	1
5	1952	Ballard	1
6	1953	Ballard	1
7	1955	Ballard	2
8	1956	Ballard	1
9	1957	Ballard	1
10	1958	Ballard	2
11	1959	Ballard	3
12	1961	Ballard	2
13	1962	Ballard	1
14	1964	Ballard	3
15	1965	Ballard	2
16	1966	Ballard	2

The status bar at the bottom indicates: Query execute... | DESKTOP-RLPM467 (16.0 RTM) | DESKTOP-RLPM467\Lanfei... | AdventureWorksDW | 00:00:00 | 7,454 rows

7. Once you have embedded all of your screen shots, submit the file in Brightspace and you're done!

**OPTIONAL BONUS:** (to guarantee 100%)

8. Create a similar query to that in Step 6, except use the tables in the OLTP version; that is use either the schema "AdventureWorks2017" or "AdventureWorks2019".

- 8.1. You will have to discover where the data is stored. You can start with the following:

```
use AdventureWorks2017;
```

```
SELECT COLUMN_NAME AS 'Column_Name', TABLE_NAME AS 'Table_Name'  
FROM INFORMATION_SCHEMA.COLUMNS  
WHERE COLUMN_NAME LIKE '%BirthDate%'  
ORDER BY Table_Name, Column_Name;
```

```
SELECT COLUMN_NAME AS 'Column_Name', TABLE_NAME AS 'Table_Name'  
FROM INFORMATION_SCHEMA.COLUMNS  
WHERE COLUMN_NAME LIKE '%City%'  
ORDER BY Table_Name, Column_Name;
```

- 8.2. It will take some digging....

- 8.3. Provide a screenshot below to show your query and output:

lab5 - Microsoft SQL Server Management Studio

Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

AdventureWorks

Execute

lab5-6.3.sql - DESK...LPM467\Lanfei (53) SQLQuery9.sql - DE...PM467\Lanfei (69))\*

```
USE AdventureWorks;

SELECT YEAR(vd.BirthDate) AS 'Birth Year', vc.City, count(*) AS 'Number of Customers'
FROM Sales.vPersonDemographics vd
JOIN Sales.vIndividualCustomer vc
ON vd.BusinessEntityID= vc.BusinessEntityID
GROUP BY YEAR(vd.BirthDate), vc.City
ORDER BY vc.City, YEAR(vd.BirthDate);
```

100 %

Results Messages

	Birth Year	City	Number of Customers
1	1936	Ballard	1
2	1940	Ballard	1
3	1941	Ballard	1
4	1942	Ballard	1
5	1945	Ballard	1
6	1947	Ballard	2
7	1948	Ballard	1
8	1949	Ballard	2
9	1950	Ballard	1
10	1952	Ballard	2
11	1953	Ballard	3
12	1954	Ballard	2
13	1955	Ballard	2
14	1956	Ballard	2
15	1957	Ballard	1
16	1958	Ballard	2
17	1959	Ballard	2

Query executed successfully. DESKTOP-RLPM467 (16.0 RTM) | DESKTOP-RLPM467\Lanfei... | AdventureWorks | 00:00:00 | 7,537 rows