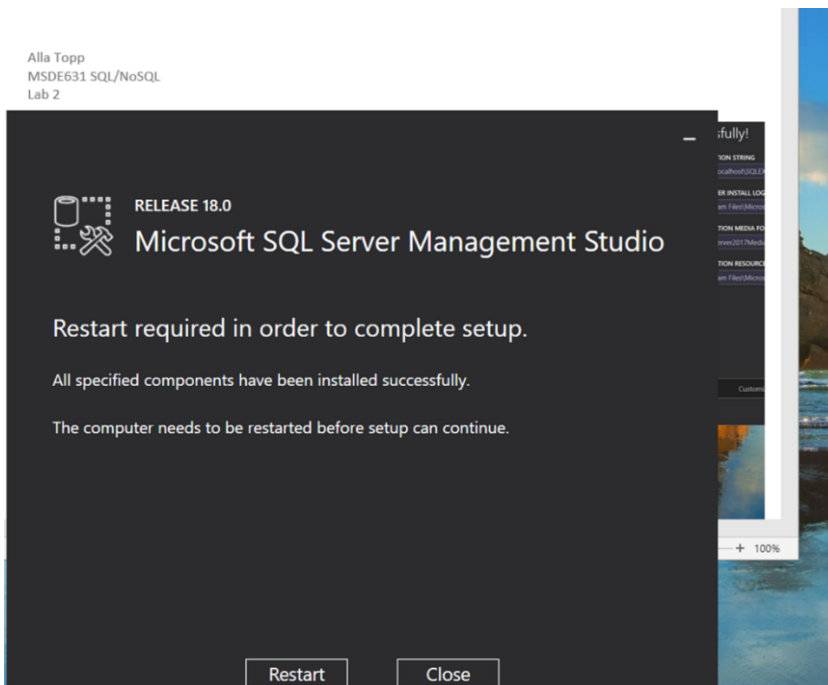
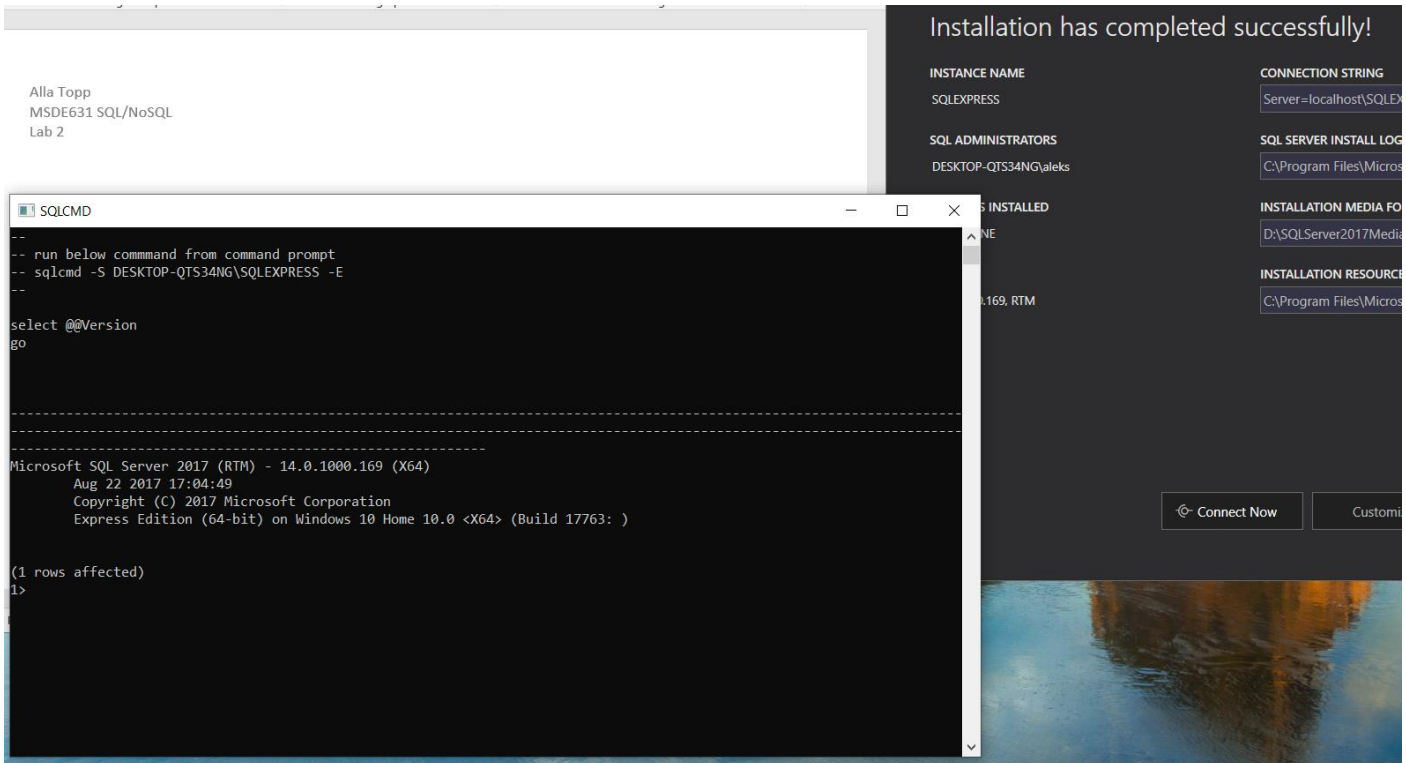


## LAB 2 Part 1. Installing and using the MS SQL Server environment

### Installation of MS SQL Server.



Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2

File Home

New Query Execute

Object Explorer

Connect

- Tables
  - System Tables
  - FileTables
  - External Tables
  - Graph Tables
  - dbo.AWBuildVersion
  - dbo.DatabaseLog
  - dbo.ErrorLog
  - HumanResources.Department
  - HumanResources.Employee
  - HumanResources.EmployeeD
  - HumanResources.EmployeeP
  - HumanResources.JobCandidate
  - HumanResources.Shift
  - Person.Address
  - Person.AddressType
  - Person.BusinessEntity
  - Person.BusinessEntityAddress
  - Person.BusinessEntityContact
  - Person.ContactType
  - Person.CountryRegion
  - Person.EmailAddress
  - Person.Password
  - Person.Person
  - Person.PersonPhone
  - Person.PhoneNumberType
  - Person.StateProvince
  - Production.BillOfMaterials
  - Production.Culture
  - Production.Document
  - Production.Illustration
  - Production.Location
  - Production.Product
  - Production.ProductCategory
  - Production.ProductCostHisto
  - Production.ProductDescriptio
  - Production.ProductDocumen

SQLCMD

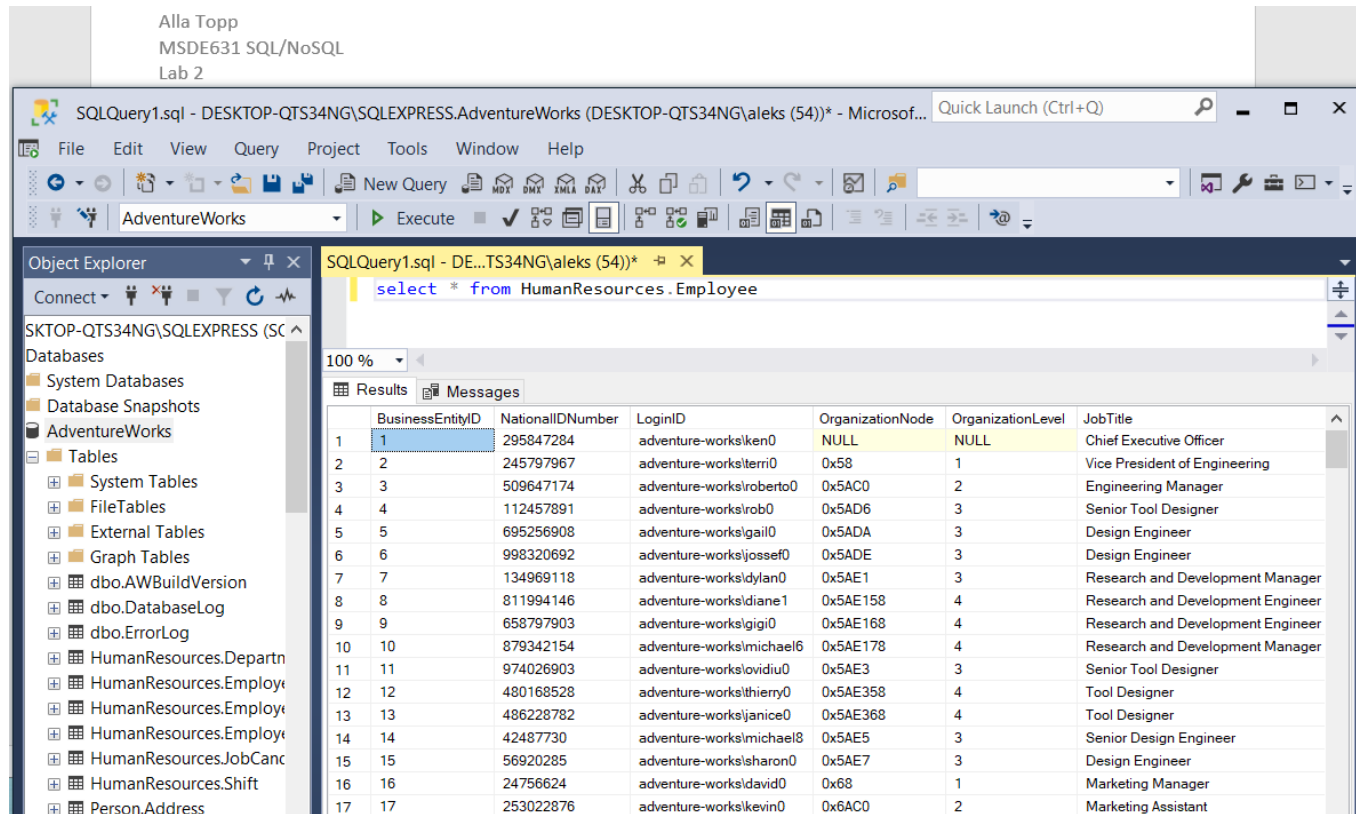
```
-- run below comm
-- sqlcmd -S DESKTO
--
select @@Version
go
.....
Microsoft SQL Servi
Aug 22 201
Copyright
Express Ed
(1 rows affected)
1>
```

Page 1 of 8 12 of

Ready

## LAB 2 Part 2. Query MS SQL Server

Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2



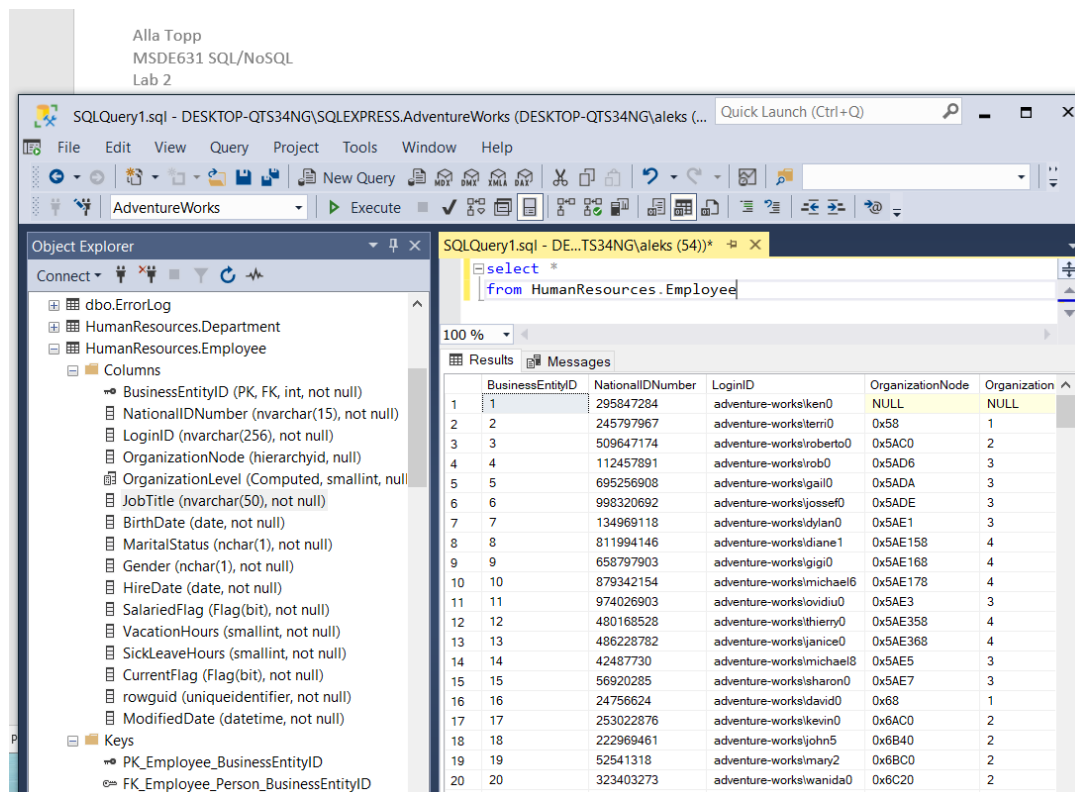
The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for AdventureWorks, including System Databases, Database Snapshots, and Tables. The main window shows a query window with the following SQL query:

```
select * from HumanResources.Employee
```

The query results are displayed in a table with the following columns: BusinessEntityID, NationalIDNumber, LoginID, OrganizationNode, OrganizationLevel, and JobTitle. The results show 17 rows of data.

	BusinessEntityID	NationalIDNumber	LoginID	OrganizationNode	OrganizationLevel	JobTitle
1	1	295847284	adventure-works\ken0	NULL	NULL	Chief Executive Officer
2	2	245797967	adventure-works\terri0	0x58	1	Vice President of Engineering
3	3	509647174	adventure-works\roberto0	0x5AC0	2	Engineering Manager
4	4	112457891	adventure-works\rob0	0x5AD6	3	Senior Tool Designer
5	5	695256908	adventure-works\gail0	0x5ADA	3	Design Engineer
6	6	998320692	adventure-works\jossef0	0x5ADE	3	Design Engineer
7	7	134969118	adventure-works\dylan0	0x5AE1	3	Research and Development Manager
8	8	811994146	adventure-works\diene1	0x5AE158	4	Research and Development Engineer
9	9	658797903	adventure-works\gigi0	0x5AE168	4	Research and Development Engineer
10	10	879342154	adventure-works\michael6	0x5AE178	4	Research and Development Manager
11	11	974026903	adventure-works\lovidiu0	0x5AE3	3	Senior Tool Designer
12	12	480168528	adventure-works\thierry0	0x5AE358	4	Tool Designer
13	13	486228782	adventure-works\janice0	0x5AE368	4	Tool Designer
14	14	42487730	adventure-works\michael8	0x5AE5	3	Senior Design Engineer
15	15	56920285	adventure-works\sharon0	0x5AE7	3	Design Engineer
16	16	24756624	adventure-works\david0	0x68	1	Marketing Manager
17	17	253022876	adventure-works\kevin0	0x6AC0	2	Marketing Assistant

Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2



The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for AdventureWorks, including System Databases, Database Snapshots, and Tables. The main window shows a query window with the following SQL query:

```
select * from HumanResources.Employee
```

The query results are displayed in a table with the following columns: BusinessEntityID, NationalIDNumber, LoginID, OrganizationNode, OrganizationLevel, and JobTitle. The results show 17 rows of data.

	BusinessEntityID	NationalIDNumber	LoginID	OrganizationNode	OrganizationLevel	JobTitle
1	1	295847284	adventure-works\ken0	NULL	NULL	Chief Executive Officer
2	2	245797967	adventure-works\terri0	0x58	1	Vice President of Engineering
3	3	509647174	adventure-works\roberto0	0x5AC0	2	Engineering Manager
4	4	112457891	adventure-works\rob0	0x5AD6	3	Senior Tool Designer
5	5	695256908	adventure-works\gail0	0x5ADA	3	Design Engineer
6	6	998320692	adventure-works\jossef0	0x5ADE	3	Design Engineer
7	7	134969118	adventure-works\dylan0	0x5AE1	3	Research and Development Manager
8	8	811994146	adventure-works\diene1	0x5AE158	4	Research and Development Engineer
9	9	658797903	adventure-works\gigi0	0x5AE168	4	Research and Development Engineer
10	10	879342154	adventure-works\michael6	0x5AE178	4	Research and Development Manager
11	11	974026903	adventure-works\lovidiu0	0x5AE3	3	Senior Tool Designer
12	12	480168528	adventure-works\thierry0	0x5AE358	4	Tool Designer
13	13	486228782	adventure-works\janice0	0x5AE368	4	Tool Designer
14	14	42487730	adventure-works\michael8	0x5AE5	3	Senior Design Engineer
15	15	56920285	adventure-works\sharon0	0x5AE7	3	Design Engineer
16	16	24756624	adventure-works\david0	0x68	1	Marketing Manager
17	17	253022876	adventure-works\kevin0	0x6AC0	2	Marketing Assistant

We will change the asterisk in the query to column names of JobTitle, HireDate and Gender.

The screenshot shows the SQL Server Enterprise Manager interface. On the left, the Object Explorer displays the database structure for 'AdventureWorks', including tables like 'HumanResources.Employee' and its columns. The main window shows a SQL query in 'SQLQuery1.sql' with the following text:

```
select JobTitle, HireDate, Gender
from HumanResources.Employee
```

Below the query editor, the 'Results' pane displays a table with 24 rows and 3 columns: JobTitle, HireDate, and Gender. The data includes various job titles such as 'Chief Executive Officer', 'Vice President of Engineering', 'Engineering Manager', 'Senior Tool Designer', 'Design Engineer', 'Research and Development Manager', 'Marketing Manager', 'Marketing Assistant', and 'Marketing Specialist'.

We added a where clause for JobTitle like '%Marketing%'

The screenshot shows the SQL Server Enterprise Manager interface with the same query editor as before, but now with a 'where' clause added:

```
select JobTitle, HireDate, Gender
from HumanResources.Employee
where JobTitle like '%Marketing%'
```

The 'Results' pane now displays only 9 rows of data, all of which have job titles containing the word 'Marketing'. The data includes 'Marketing Manager', 'Marketing Assistant', and 'Marketing Specialist'.

Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2

Change the word Marketing to upper case. We see no difference. Transact SQL does not care about the case of the text for the comparison and compares upper case and lower case.

Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2

SQLQuery1.sql - DESKTOP-QTS34NG\SQLEXPRESS.AdventureWorks (DESKTOP-QTS34NG\aleks (... Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

AdventureWorks Execute

Object Explorer

- dbo.ErrorLog
- HumanResources.Department
- HumanResources.Employee
- Columns
  - BusinessEntityID (PK, FK, int, not null)
  - NationalIDNumber (nvarchar(15), not null)
  - LoginID (nvarchar(256), not null)
  - OrganizationNode (hierarchyid, null)
  - OrganizationLevel (Computed, smallint, null)
  - JobTitle (nvarchar(50), not null)
  - BirthDate (date, not null)
  - MaritalStatus (nchar(1), not null)
  - Gender (nchar(1), not null)

SQLQuery1.sql - DE...TS34NG\aleks (54))\*

```
select JobTitle, HireDate, Gender
from HumanResources.Employee
where JobTitle like '%MARKETING%'
```

100 %

Results Messages

	JobTitle	HireDate	Gender
1	Marketing Manager	2007-12-20	M
2	Marketing Assistant	2007-01-26	M
3	Marketing Specialist	2011-02-07	M
4	Marketing Assistant	2011-02-14	F
5	Marketing Assistant	2011-01-07	F
6	Marketing Specialist	2009-03-02	M
7	Marketing Specialist	2008-12-12	M
8	Marketing Specialist	2009-01-12	F
9	Marketing Specialist	2009-01-18	F

Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2

SQLQuery1.sql - DESKTOP-QTS34NG\SQLEXPRESS.AdventureWorks (DESKTOP-QTS34NG\aleks (... Quick Launch (Ctrl+Q)

File Edit View Query Project Tools Window Help

AdventureWorks Execute

Object Explorer

- dbo.ErrorLog
- HumanResources.Department
- HumanResources.Employee
- Columns
  - BusinessEntityID (PK, FK, int, not null)
  - NationalIDNumber (nvarchar(15), not null)
  - LoginID (nvarchar(256), not null)
  - OrganizationNode (hierarchyid, null)
  - OrganizationLevel (Computed, smallint, null)
  - JobTitle (nvarchar(50), not null)
  - BirthDate (date, not null)
  - MaritalStatus (nchar(1), not null)
  - Gender (nchar(1), not null)
  - HireDate (date, not null)
  - SalariedFlag (Flag(bit), not null)
  - VacationHours (smallint, not null)
  - SickLeaveHours (smallint, not null)
  - CurrentFlag (Flag(bit), not null)
  - rowguid (uniqueidentifier, not null)
  - ModifiedDate (datetime, not null)

SQLQuery1.sql - DE...TS34NG\aleks (54))\*

```
select JobTitle, HireDate, Gender
from HumanResources.Employee
where JobTitle like '%manager%'
```

100 %

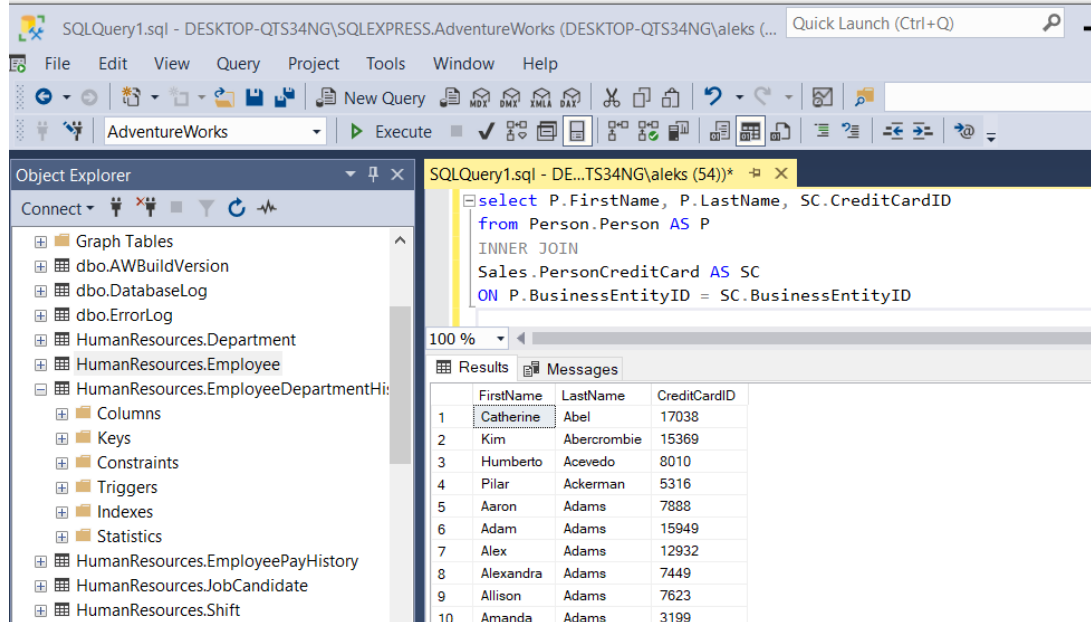
Results Messages

	JobTitle	HireDate	Gender
1	Engineering Manager	2007-11-11	M
2	Research and Development Manager	2009-02-08	M
3	Research and Development Manager	2009-05-03	M
4	Marketing Manager	2007-12-20	M
5	Production Control Manager	2008-12-01	M
6	Quality Assurance Manager	2009-02-28	M
7	Document Control Manager	2009-01-04	M
8	Facilities Manager	2009-12-02	M
9	Human Resources Manager	2008-12-06	F
10	Accounts Manager	2009-01-30	M
11	Finance Manager	2008-12-25	F
12	Purchasing Manager	2011-02-25	F
13	Information Services Manager	2008-12-11	F
14	Network Manager	2009-02-04	F
15	North American Sales Manager	2011-01-04	M
16	Pacific Sales Manager	2013-03-14	M
17	European Sales Manager	2012-04-16	F

## Join Tables

We can join the three tables of Department, Employee and EmployeeDepartmentHistory using the following SQL.

Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2



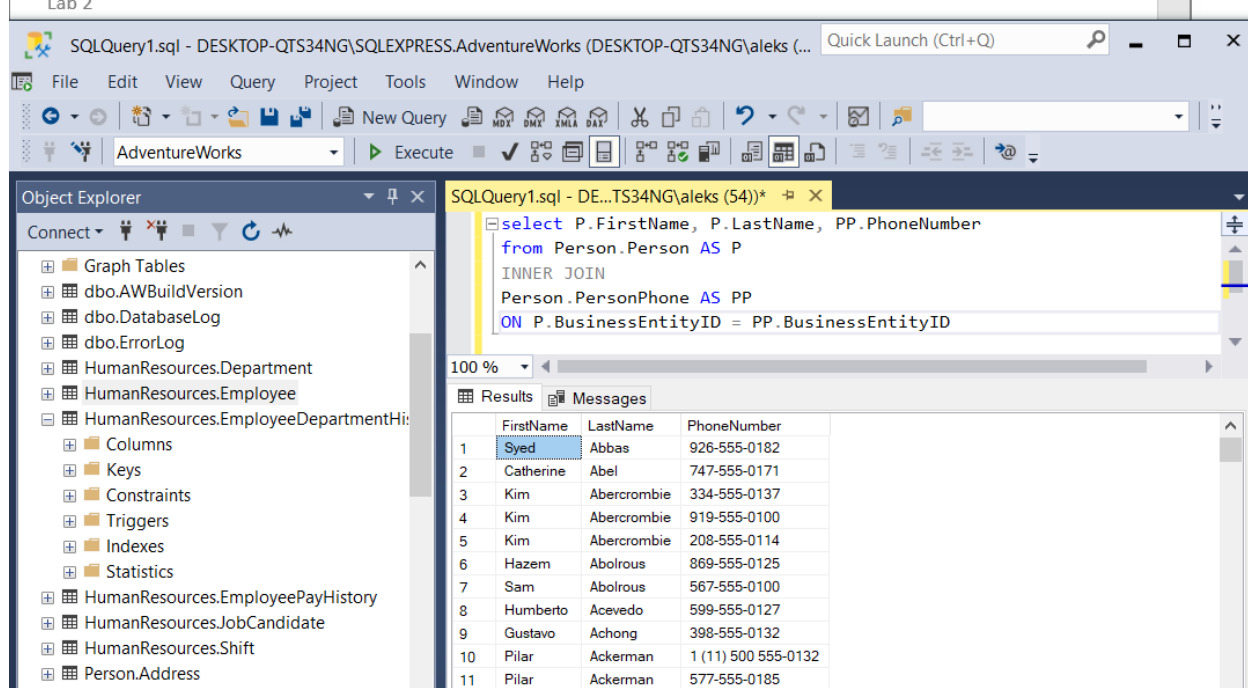
The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure, including tables like HumanResources.Department, HumanResources.Employee, and HumanResources.EmployeeDepartmentHistory. The central pane shows a SQL query in the 'SQLQuery1.sql' file, which joins the Person, Sales, and EmployeeDepartmentHistory tables. The Results pane on the right displays the query output, showing a list of employees with their first and last names and credit card IDs.

```
select P.FirstName, P.LastName, SC.CreditCardID
from Person.Person AS P
INNER JOIN
Sales.PersonCreditCard AS SC
ON P.BusinessEntityID = SC.BusinessEntityID
```

	FirstName	LastName	CreditCardID
1	Catherine	Abel	17038
2	Kim	Abercrombie	15369
3	Humberto	Acevedo	8010
4	Pilar	Ackerman	5316
5	Aaron	Adams	7888
6	Adam	Adams	15949
7	Alex	Adams	12932
8	Alexandra	Adams	7449
9	Allison	Adams	7623
10	Amanda	Adams	3199

We are going to join against another table called Person.PhoneNumberType instead of CreditCardID.

Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2



The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure, including tables like HumanResources.Department, HumanResources.Employee, and HumanResources.EmployeeDepartmentHistory. The central pane shows a SQL query in the 'SQLQuery1.sql' file, which joins the Person and PersonPhone tables. The Results pane on the right displays the query output, showing a list of employees with their first and last names and phone numbers.

```
select P.FirstName, P.LastName, PP.PhoneNumber
from Person.Person AS P
INNER JOIN
Person.PersonPhone AS PP
ON P.BusinessEntityID = PP.BusinessEntityID
```

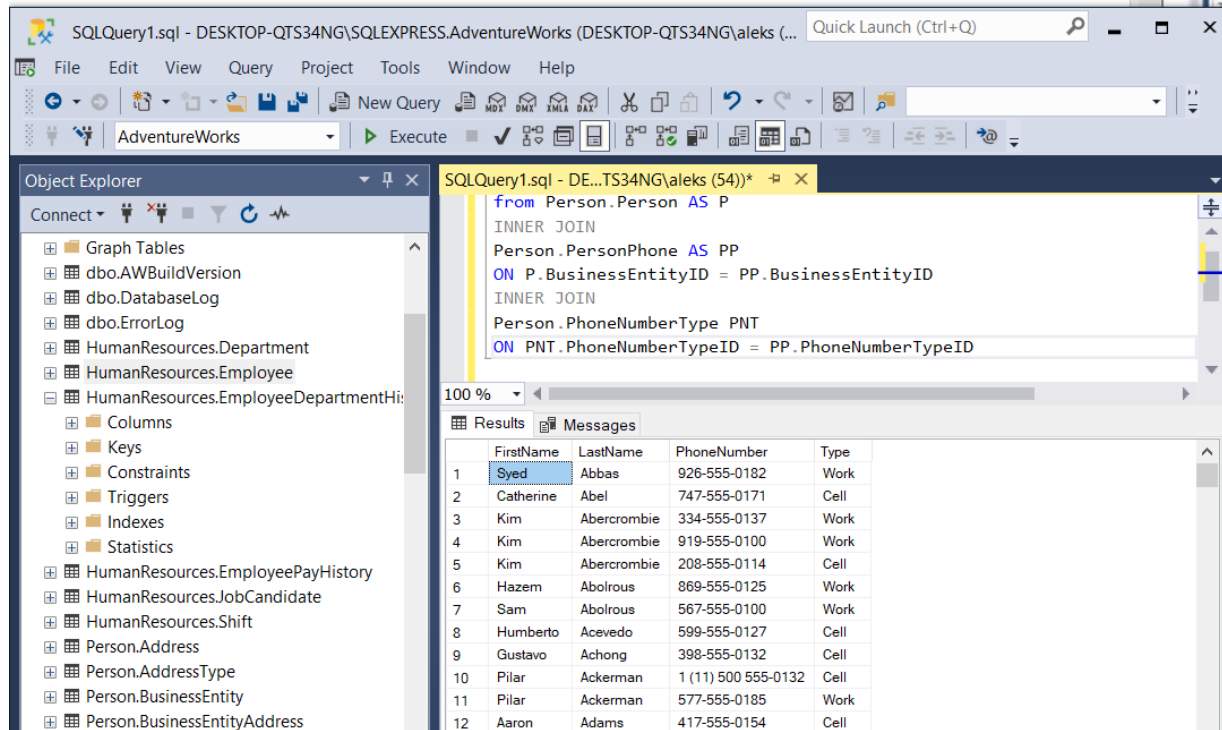
	FirstName	LastName	PhoneNumber
1	Syed	Abbas	926-555-0182
2	Catherine	Abel	747-555-0171
3	Kim	Abercrombie	334-555-0137
4	Kim	Abercrombie	919-555-0100
5	Kim	Abercrombie	208-555-0114
6	Hazem	Abolrous	869-555-0125
7	Sam	Abolrous	567-555-0100
8	Humberto	Acevedo	599-555-0127
9	Gustavo	Achong	398-555-0132
10	Pilar	Ackerman	1 (11) 500 555-0132
11	Pilar	Ackerman	577-555-0185



Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2

Next, we added another column Name Type and called the column Type, so it is clearer because we already have a person's name. My screenshot does not show the first line of the code, it should be select P.FirstName, P.LastName, PP.PhoneNumber, PNT .Name Type.

Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2



The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure, including tables like Person, PersonPhone, and PersonNumberType. The central pane shows a SQL query with the following text:

```
from Person.Person AS P
INNER JOIN
Person.PersonPhone AS PP
ON P.BusinessEntityID = PP.BusinessEntityID
INNER JOIN
Person.PhoneNumberType PNT
ON PNT.PhoneNumberTypeID = PP.PhoneNumberTypeID
```

Below the query, the Results pane displays a table with 12 rows and 4 columns: FirstName, LastName, PhoneNumber, and Type. The data is as follows:

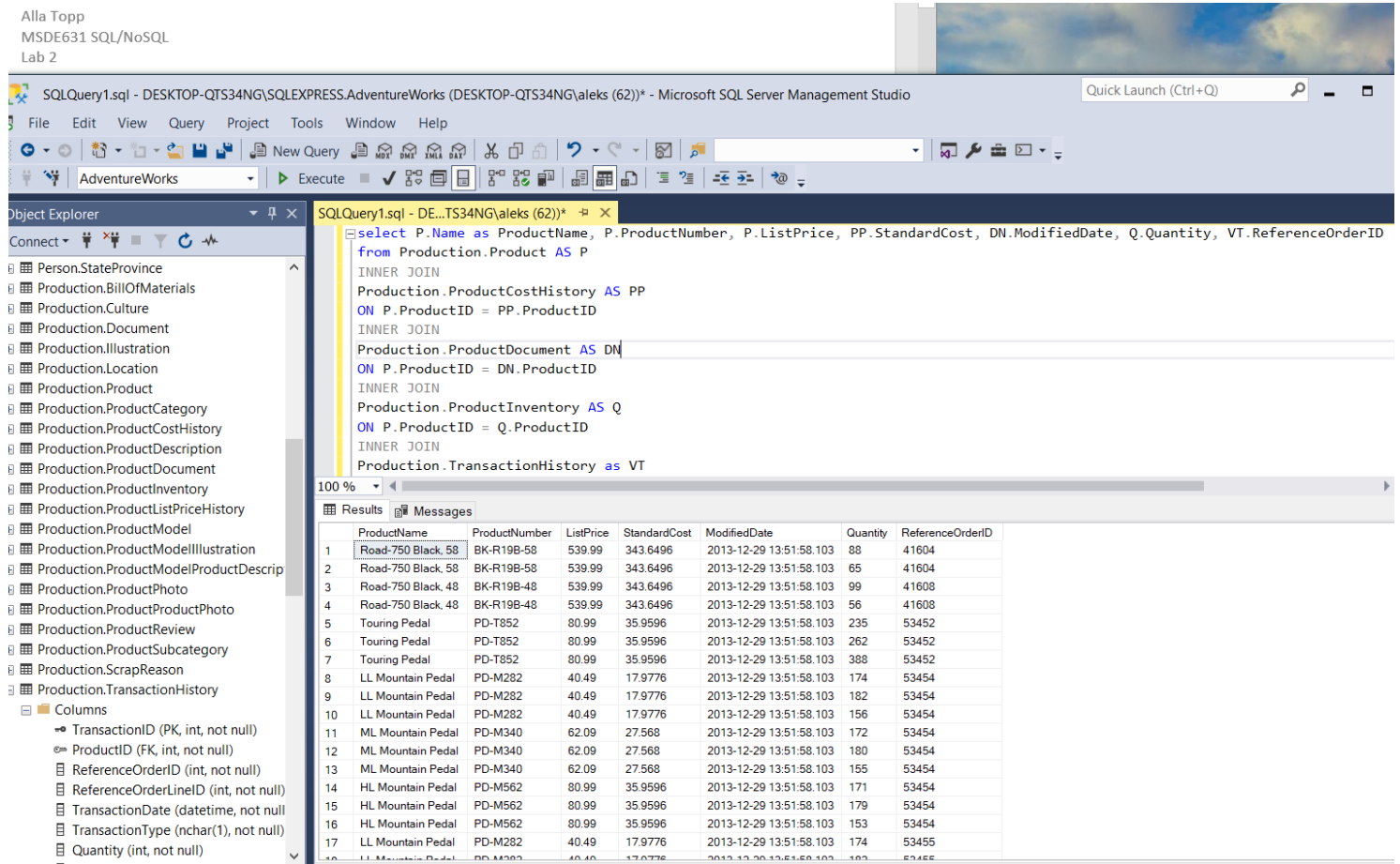
	FirstName	LastName	PhoneNumber	Type
1	Syed	Abbas	926-555-0182	Work
2	Catherine	Abel	747-555-0171	Cell
3	Kim	Abercrombie	334-555-0137	Work
4	Kim	Abercrombie	919-555-0100	Work
5	Kim	Abercrombie	208-555-0114	Cell
6	Hazem	Abolrous	869-555-0125	Work
7	Sam	Abolrous	567-555-0100	Work
8	Humberto	Acevedo	599-555-0127	Cell
9	Gustavo	Achong	398-555-0132	Cell
10	Pilar	Ackerman	1 (11) 500 555-0132	Cell
11	Pilar	Ackerman	577-555-0185	Work
12	Aaron	Adams	417-555-0154	Cell

**Assignment:**

At this point, you will create your own query that will join 5 tables together. The Production.Product has four foreign keys as shown below. Write a query that joins the Product table with the four other tables using INNER JOINS. Select the columns from other tables that will validate that the joins to all four tables are working correctly; make the query results make sense. Include the Product name, Product Number, and ListPrice as part of the result set (plus at least one other column from the four other tables).

Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2

Alla Topp  
MSDE631 SQL/NoSQL  
Lab 2



The screenshot displays the Microsoft SQL Server Management Studio interface. The title bar indicates the connection is to 'AdventureWorks'. The 'Object Explorer' on the left shows the database structure, including tables like 'Production.Product' and 'Production.TransactionHistory'. The 'SQL Query1.sql' editor in the center contains a complex SQL query that joins five tables: 'Production.Product', 'Production.ProductCostHistory', 'Production.ProductDocument', 'Production.ProductInventory', and 'Production.TransactionHistory'. The query selects columns: 'P.Name as ProductName', 'P.ProductNumber', 'P.ListPrice', 'PP.StandardCost', 'DN.ModifiedDate', 'Q.Quantity', and 'VT.ReferenceOrderID'. The 'Results' pane at the bottom shows the output of the query, which is a table with 7 columns and 17 rows of data. The data includes product names like 'Road-750 Black, 58', 'Road-750 Black, 48', 'Touring Pedal', and 'LL Mountain Pedal', along with their respective product numbers, prices, costs, dates, quantities, and reference order IDs.

```
select P.Name as ProductName, P.ProductNumber, P.ListPrice, PP.StandardCost, DN.ModifiedDate, Q.Quantity, VT.ReferenceOrderID
from Production.Product AS P
INNER JOIN
Production.ProductCostHistory AS PP
ON P.ProductID = PP.ProductID
INNER JOIN
Production.ProductDocument AS DN
ON P.ProductID = DN.ProductID
INNER JOIN
Production.ProductInventory AS Q
ON P.ProductID = Q.ProductID
INNER JOIN
Production.TransactionHistory as VT
```

	ProductName	ProductNumber	ListPrice	StandardCost	ModifiedDate	Quantity	ReferenceOrderID
1	Road-750 Black, 58	BK-R198-58	539.99	343.6496	2013-12-29 13:51:58.103	88	41604
2	Road-750 Black, 58	BK-R198-58	539.99	343.6496	2013-12-29 13:51:58.103	65	41604
3	Road-750 Black, 48	BK-R198-48	539.99	343.6496	2013-12-29 13:51:58.103	99	41608
4	Road-750 Black, 48	BK-R198-48	539.99	343.6496	2013-12-29 13:51:58.103	56	41608
5	Touring Pedal	PD-T852	80.99	35.9596	2013-12-29 13:51:58.103	235	53452
6	Touring Pedal	PD-T852	80.99	35.9596	2013-12-29 13:51:58.103	262	53452
7	Touring Pedal	PD-T852	80.99	35.9596	2013-12-29 13:51:58.103	388	53452
8	LL Mountain Pedal	PD-M282	40.49	17.9776	2013-12-29 13:51:58.103	174	53454
9	LL Mountain Pedal	PD-M282	40.49	17.9776	2013-12-29 13:51:58.103	182	53454
10	LL Mountain Pedal	PD-M282	40.49	17.9776	2013-12-29 13:51:58.103	156	53454
11	ML Mountain Pedal	PD-M340	62.09	27.568	2013-12-29 13:51:58.103	172	53454
12	ML Mountain Pedal	PD-M340	62.09	27.568	2013-12-29 13:51:58.103	180	53454
13	ML Mountain Pedal	PD-M340	62.09	27.568	2013-12-29 13:51:58.103	155	53454
14	HL Mountain Pedal	PD-M562	80.99	35.9596	2013-12-29 13:51:58.103	171	53454
15	HL Mountain Pedal	PD-M562	80.99	35.9596	2013-12-29 13:51:58.103	179	53454
16	HL Mountain Pedal	PD-M562	80.99	35.9596	2013-12-29 13:51:58.103	153	53454
17	LL Mountain Pedal	PD-M282	40.49	17.9776	2013-12-29 13:51:58.103	174	53455

Here I created 7 columns(ProductName,ProductNumber,ListPrice,StandardCost, ModifiedDate, Quantity and ReferenceOrderID) by joining 5 tables Production.Product, Production.ProductCostHistory, Production.ProductDocument, Production.ProductInventory and Production.TransactionHistory. All of the tables were joined based on the ProductID key.