Project 3

Data Reporting and Analysis with T-SQL

Prepared By

Abhinandan Patil

Data Science Intern at AINE.AI

Abhinandanpatil30@gmail.com

Project: Product sales performance analysis using T-SQL.

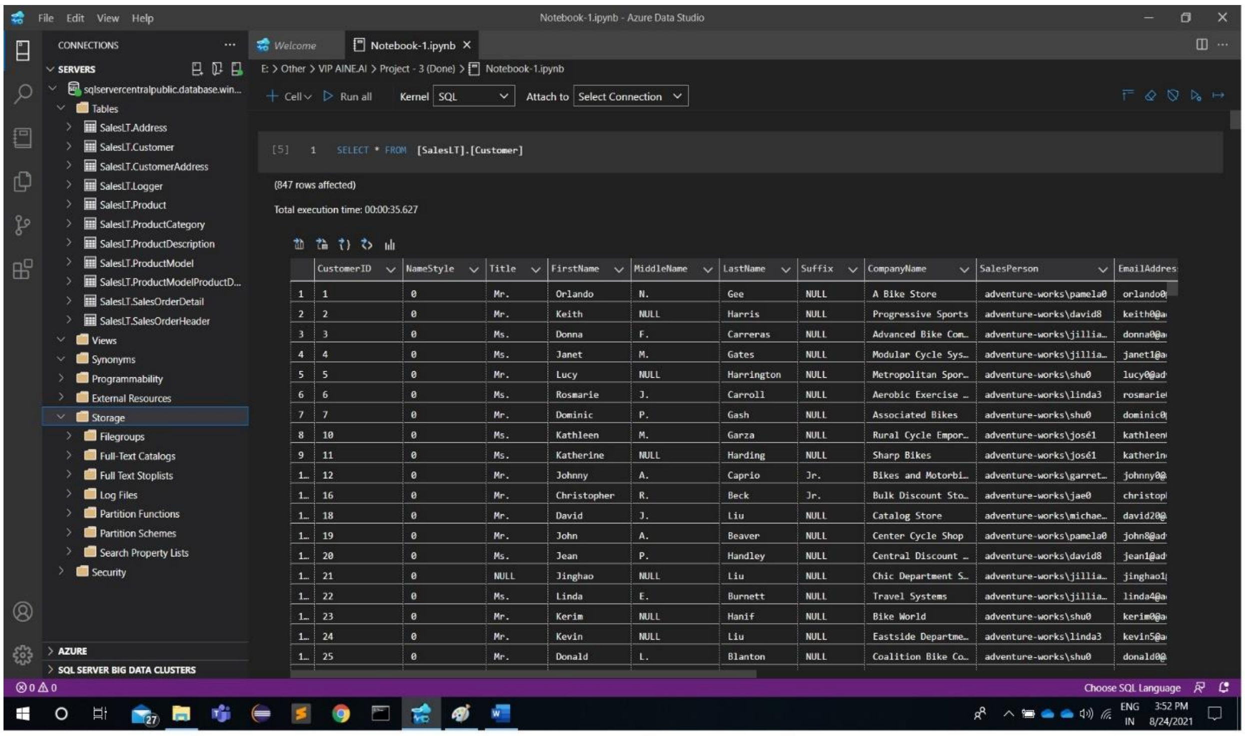
About the project: Analysis of sales of various products by customers demographics and product categories for Adventure Works Cycles using T-SQL programming on Azure Data Studio.

Tools to Use: Azure Data Studio

Aim: Using T-SQL programming to summarize the sales of Adventure Works Cycles with respect to product characteristics, promotion cost and customer demographics.

2.a) Retrieve customer details

Familiarize yourself with the Customer table by writing a Transact-SQL query that retrieves all columns for all customers.



```sql
SELECT * FROM [SalesLT].[Customer]
```

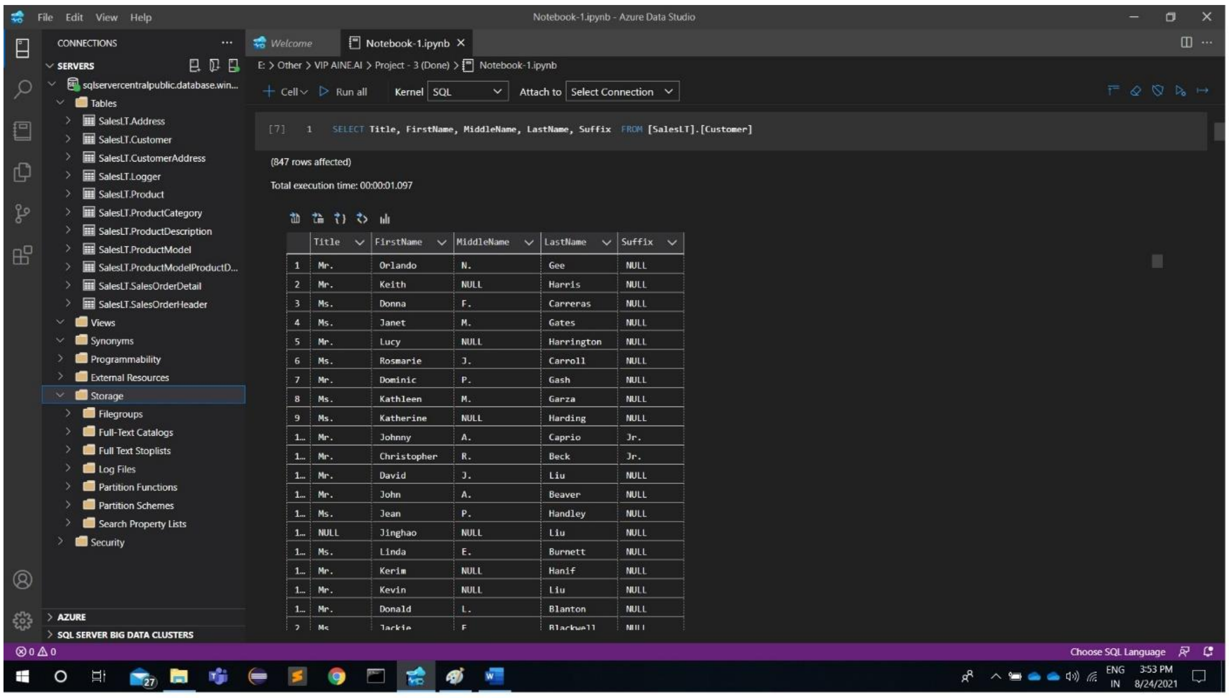
```
(847 rows affected)

Total execution time: 00:00:35.627
```

	CustomerID	NameStyle	Title	FirstName	MiddleName	LastName	Suffix	CompanyName	SalesPerson	EmailAddres
1	1	0	Mr.	Orlando	N.	Gee	NULL	A Bike Store	adventure-works\pamela0	orlando0
2	2	0	Mr.	Keith	NULL	Harris	NULL	Progressive Sports	adventure-works\david8	keith0@a
3	3	0	Ms.	Donna	F.	Carreras	NULL	Advanced Bike Com…	adventure-works\jillia…	donna0@a
4	4	0	Ms.	Janet	M.	Gates	NULL	Modular Cycle Sys…	adventure-works\jillia…	janet1@a
5	5	0	Mr.	Lucy	NULL	Harrington	NULL	Metropolitan Spor…	adventure-works\shu0	lucy0@ad
6	6	0	Ms.	Rosmarie	J.	Carroll	NULL	Aerobic Exercise …	adventure-works\linda3	rosmarie
7	7	0	Mr.	Dominic	P.	Gash	NULL	Associated Bikes	adventure-works\shu0	dominic0
8	10	0	Ms.	Kathleen	M.	Garza	NULL	Rural Cycle Empor…	adventure-works\josé1	kathleen
9	11	0	Ms.	Katherine	NULL	Harding	NULL	Sharp Bikes	adventure-works\josé1	katherin
1…	12	0	Mr.	Johnny	A.	Caprio	Jr.	Bikes and Motorbi…	adventure-works\garret…	johnny0@
1…	16	0	Mr.	Christopher	R.	Beck	Jr.	Bulk Discount Sto…	adventure-works\jae0	christop
1…	18	0	Mr.	David	J.	Liu	NULL	Catalog Store	adventure-works\michae…	david20@
1…	19	0	Mr.	John	A.	Beaver	NULL	Center Cycle Shop	adventure-works\pamela0	john8@ad
1…	20	0	Ms.	Jean	P.	Handley	NULL	Central Discount …	adventure-works\david8	jean1@ad
1…	21	0	NULL	Jinghao	NULL	Liu	NULL	Chic Department S…	adventure-works\jillia…	jinghao1
1…	22	0	Ms.	Linda	E.	Burnett	NULL	Travel Systems	adventure-works\jillia…	linda4@a
1…	23	0	Mr.	Kerim	NULL	Hanif	NULL	Bike World	adventure-works\shu0	kerim0@a
1…	24	0	Mr.	Kevin	NULL	Liu	NULL	Eastside Departme…	adventure-works\linda3	kevin5@a
1…	25	0	Mr.	Donald	L.	Blanton	NULL	Coalition Bike Co…	adventure-works\shu0	donald0@

As code shown in image I write a Transact-SQL query that retrieve all columns for all customers.

2.b) Retrieve customer name data

Create a list of all customer contact names that includes the title, first name, middle name (if any), last name, and suffix (if any) of all customers.



```sql
SELECT Title, FirstName, MiddleName, LastName, Suffix  FROM [SalesLT].[Customer]
```

```
(847 rows affected)

Total execution time: 00:00:01.097
```

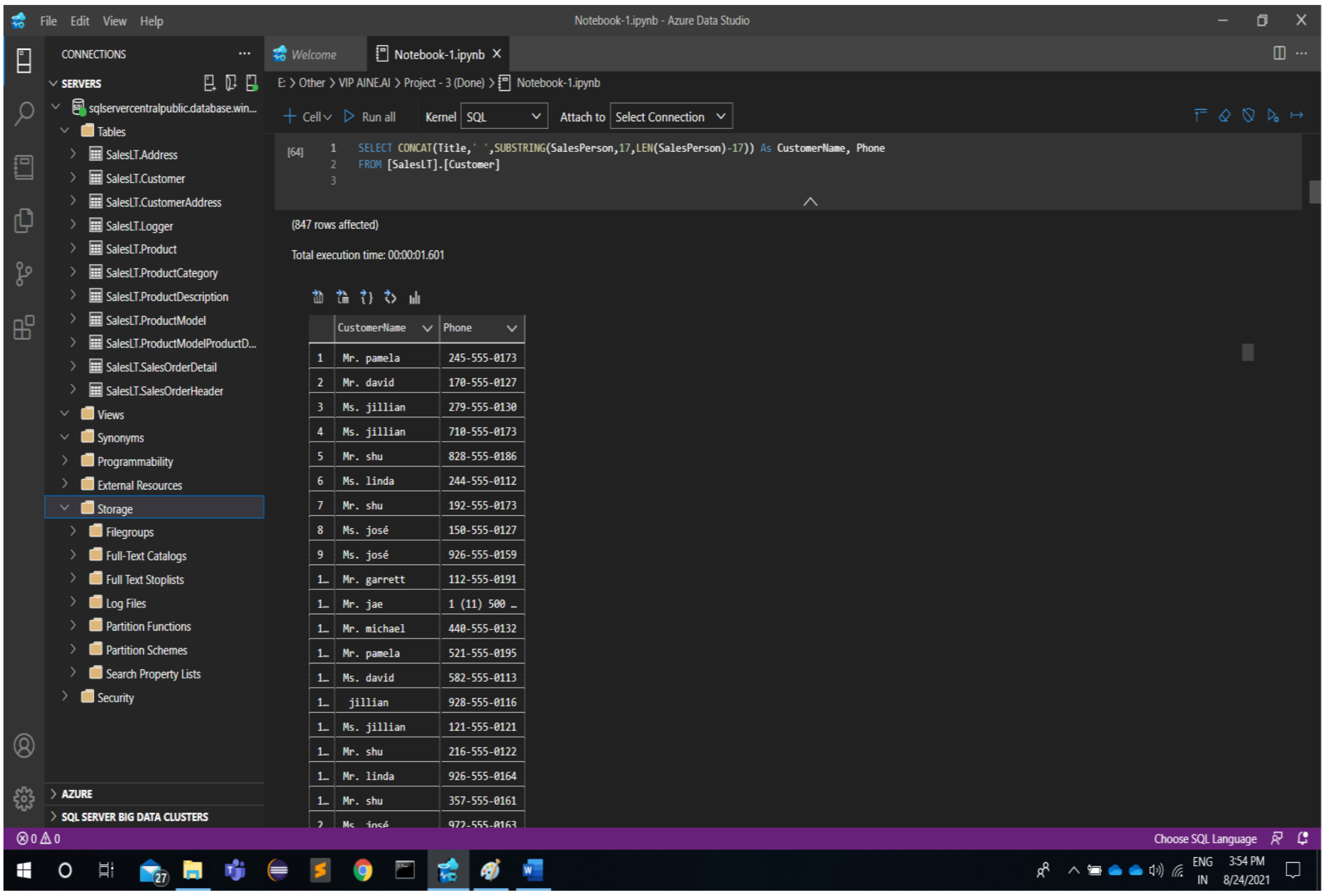
	Title	FirstName	MiddleName	LastName	Suffix
1	Mr.	Orlando	N.	Gee	NULL
2	Mr.	Keith	NULL	Harris	NULL
3	Ms.	Donna	F.	Carreras	NULL
4	Ms.	Janet	M.	Gates	NULL
5	Mr.	Lucy	NULL	Harrington	NULL
6	Ms.	Rosmarie	J.	Carroll	NULL
7	Mr.	Dominic	P.	Gash	NULL
8	Ms.	Kathleen	M.	Garza	NULL
9	Ms.	Katherine	NULL	Harding	NULL
1…	Mr.	Johnny	A.	Caprio	Jr.
1…	Mr.	Christopher	R.	Beck	Jr.
1…	Mr.	David	J.	Liu	NULL
1…	Mr.	John	A.	Beaver	NULL
1…	Ms.	Jean	P.	Handley	NULL
1…	NULL	Jinghao	NULL	Liu	NULL
1…	Ms.	Linda	E.	Burnett	NULL
1…	Mr.	Kerim	NULL	Hanif	NULL
1…	Mr.	Kevin	NULL	Liu	NULL
1…	Mr.	Donald	L.	Blanton	NULL
2	Ms	Jackie	E	Blackwell	NULL

Code shown in image I extract the title, first name, middle name (if any), last name, and suffix (if any) of all customers from [SalesLT].[Customer].

2.c) Retrieve customer names and phone numbers

Each customer has an assigned salesperson. You must write a query to create a call list sheet:

- The salesperson
- A column named CustomerName that displays how the customer contact should be greeted (for example, “Mr Smith”)
- The customer’s phone number.



```
SELECT CONCAT(Title,' ',SUBSTRING(SalesPerson,17,LEN(SalesPerson)-17)) As CustomerName, Phone
FROM [SalesLT].[Customer]
```

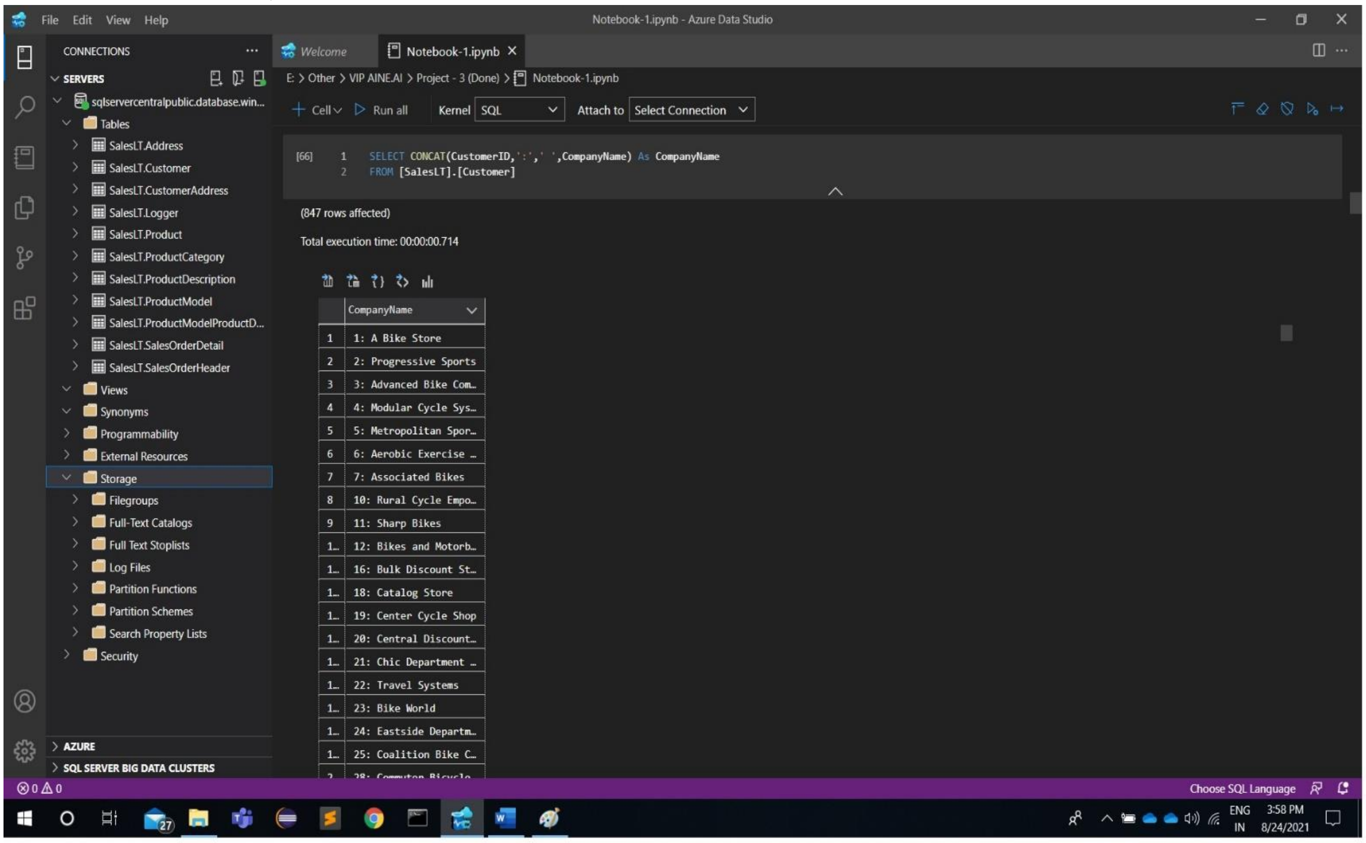
```
(847 rows affected)

Total execution time: 00:00:01.601
```

	CustomerName	Phone
1	Mr. pamela	245-555-0173
2	Mr. david	170-555-0127
3	Ms. jillian	279-555-0130
4	Ms. jillian	710-555-0173
5	Mr. shu	828-555-0186
6	Ms. linda	244-555-0112
7	Mr. shu	192-555-0173
8	Ms. josé	150-555-0127
9	Ms. josé	926-555-0159
1…	Mr. garrett	112-555-0191
1…	Mr. jae	1 (11) 500 …
1…	Mr. michael	440-555-0132
1…	Mr. pamela	521-555-0195
1…	Ms. david	582-555-0113
1…	jillian	928-555-0116
1…	Ms. jillian	121-555-0121
1…	Mr. shu	216-555-0122
1…	Mr. linda	926-555-0164
1…	Mr. shu	357-555-0161
2	Ms. josé	972-555-0163

As query shown in image I create call list sheet of column named CustomerName that displays how the customer contact should be greeted (for example, ”Mr Smith”) and customer phone number column.

3.a) Retrieve a list of customer companies

You have been asked to provide a list of all customer companies in the format : - for example, 78: Preferred Bikes.



```
SELECT CONCAT(CustomerID,':',' ',CompanyName) As CompanyName
FROM [SalesLT].[Customer]
```

```
(847 rows affected)

Total execution time: 00:00:00.714
```

	CompanyName
1	1: A Bike Store
2	2: Progressive Sports
3	3: Advanced Bike Com…
4	4: Modular Cycle Sys…
5	5: Metropolitan Spor…
6	6: Aerobic Exercise …
7	7: Associated Bikes
8	10: Rural Cycle Empo…
9	11: Sharp Bikes
1…	12: Bikes and Motorb…
1…	16: Bulk Discount St…
1…	18: Catalog Store
1…	19: Center Cycle Shop
1…	20: Central Discount…
1…	21: Chic Department …
1…	22: Travel Systems
1…	23: Bike World
1…	24: Eastside Departm…
1…	25: Coalition Bike C…

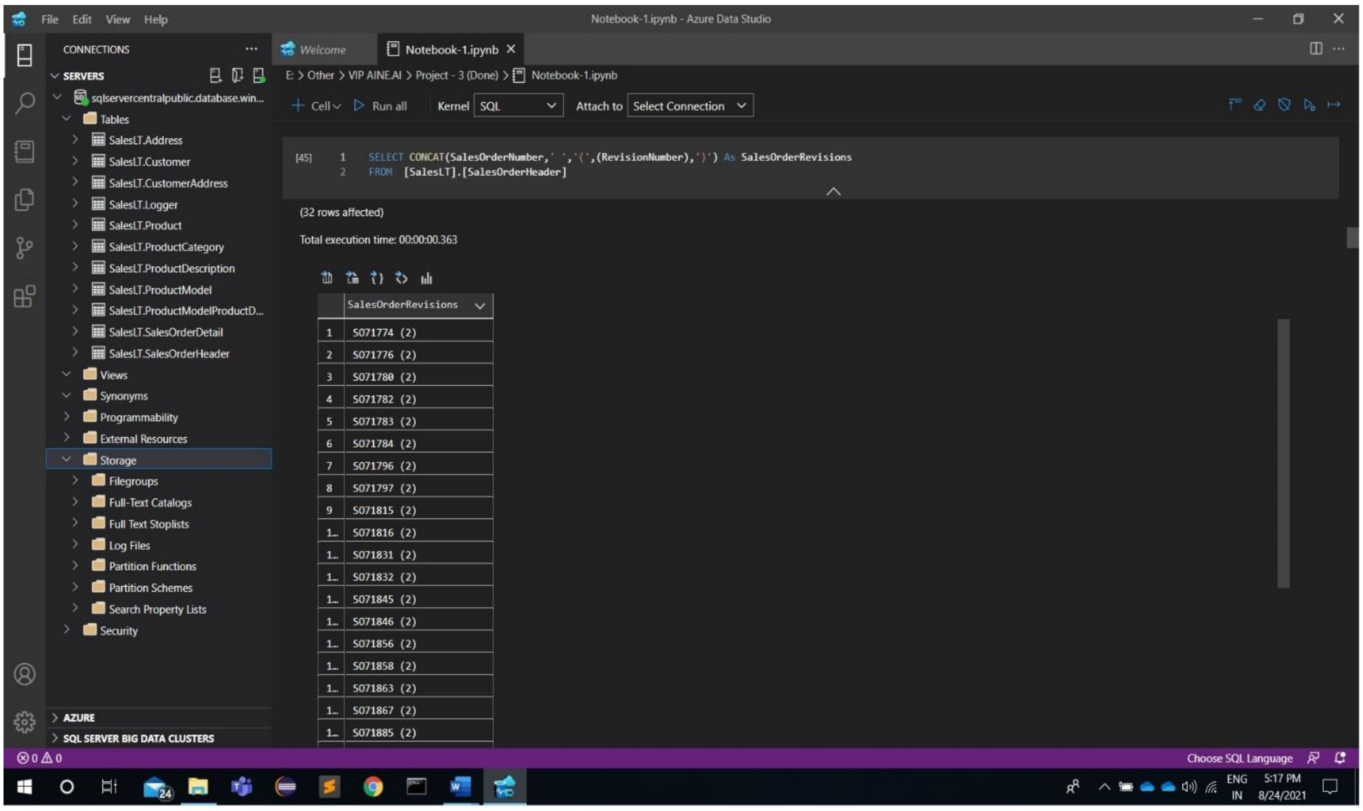
As query shown in image I retrieve the company names in the format :- for example, 78: Preferred Bikes.

3.b) Retrieve a list of sales order revisions

The SalesLT.SalesOrderHeader table contains records of sales orders. You have been asked to retrieve data for a report that shows:

• The sales order number and revision number in the format () – for example SO71774 (2).

• The order date converted to ANSI standard format (yyyy.mm.dd – for example 2015.01.31).



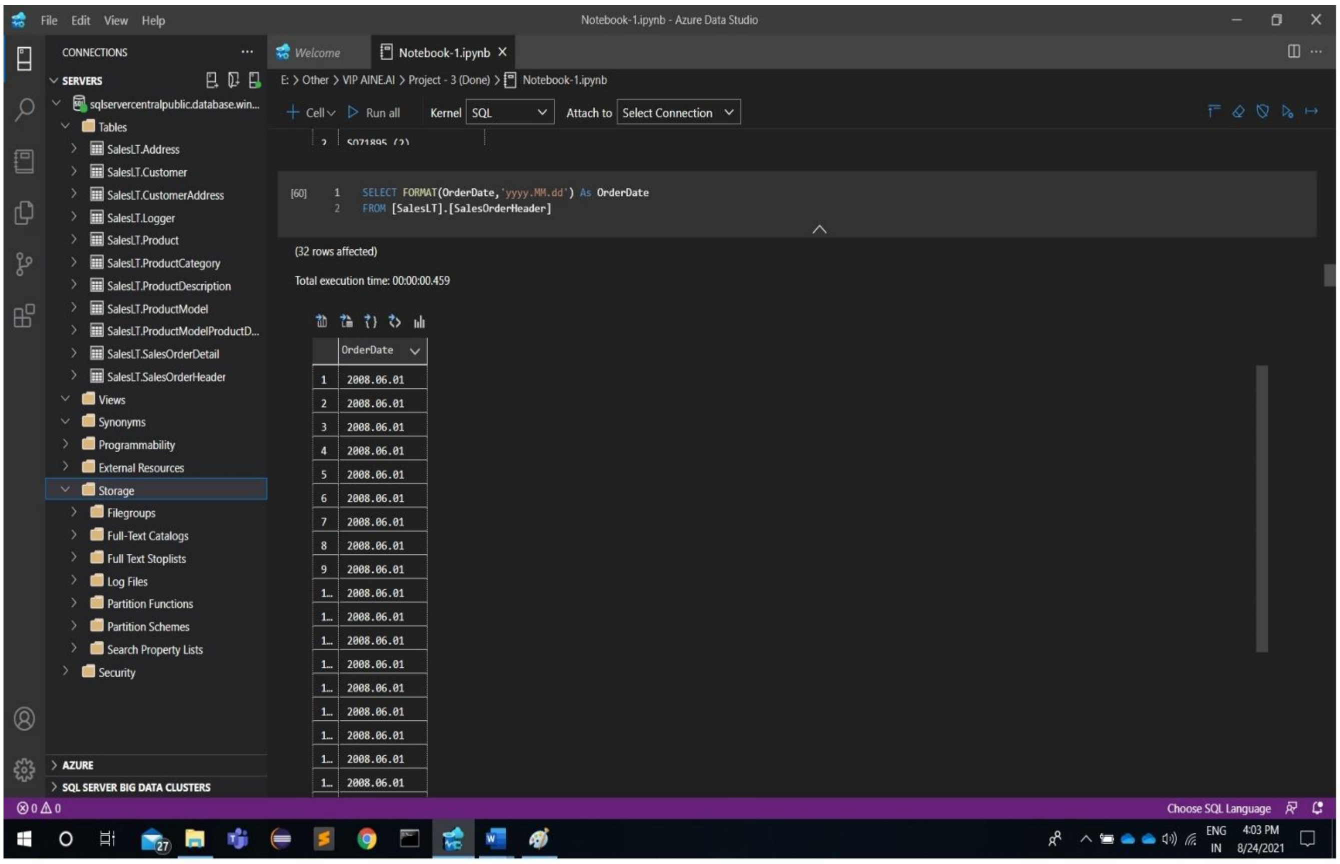
```sql
SELECT CONCAT(SalesOrderNumber,' ','(',(RevisionNumber),')') As SalesOrderRevisions
FROM  [SalesLT].[SalesOrderHeader]
```

(32 rows affected)

Total execution time: 00:00:00.363

	SalesOrderRevisions
1	SO71774 (2)
2	SO71776 (2)
3	SO71780 (2)
4	SO71782 (2)
5	SO71783 (2)
6	SO71784 (2)
7	SO71796 (2)
8	SO71797 (2)
9	SO71815 (2)
1…	SO71816 (2)
1…	SO71831 (2)
1…	SO71832 (2)
1…	SO71845 (2)
1…	SO71846 (2)
1…	SO71856 (2)
1…	SO71858 (2)
1…	SO71863 (2)
1…	SO71867 (2)
1…	SO71885 (2)

From above query shown in image I retrieve the data that shows the sales order number and revision number in the format for example, SO71774 (2).



```sql
SELECT FORMAT(OrderDate,'yyyy.MM.dd') As OrderDate
FROM [SalesLT].[SalesOrderHeader]
```

(32 rows affected)

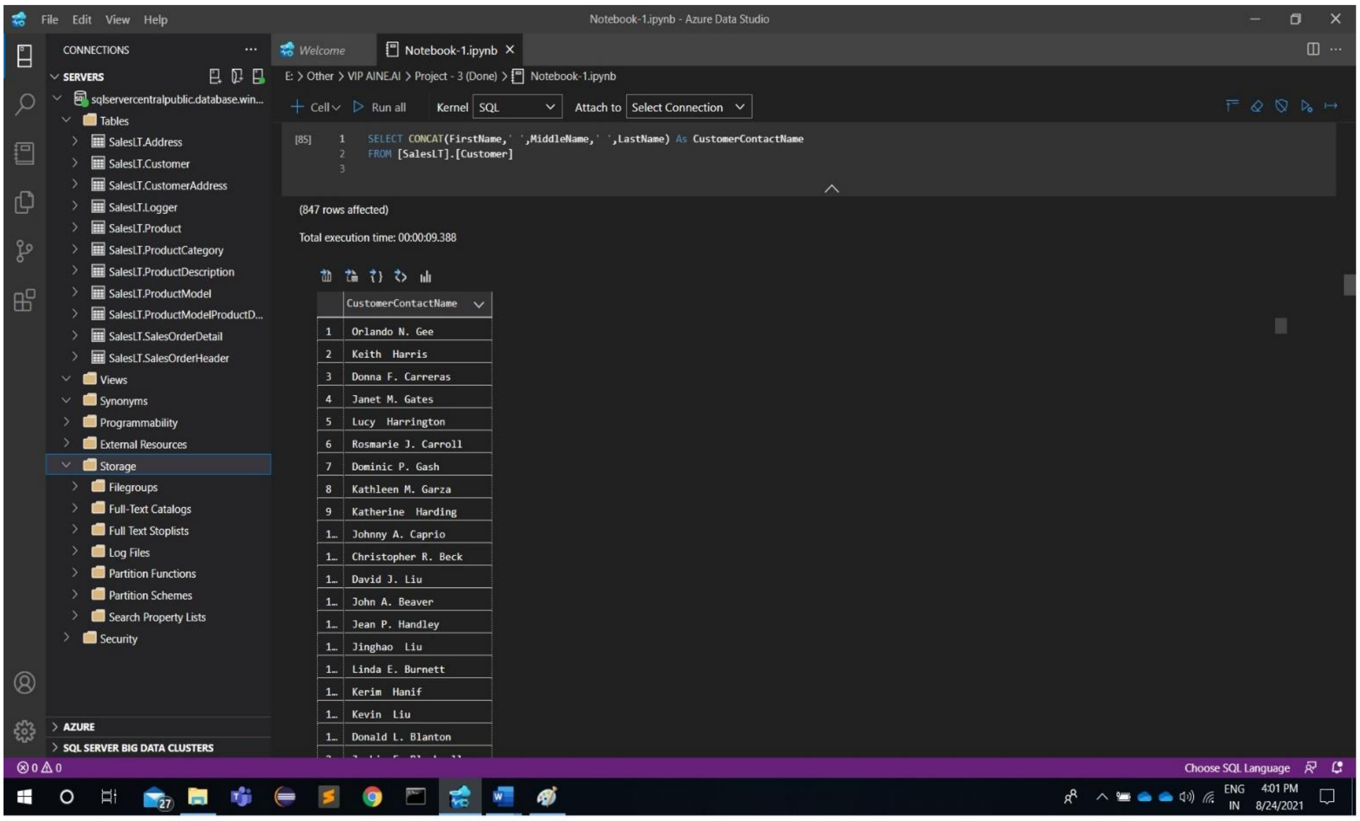
Total execution time: 00:00:00.459

	OrderDate
1	2008.06.01
2	2008.06.01
3	2008.06.01
4	2008.06.01
5	2008.06.01
6	2008.06.01
7	2008.06.01
8	2008.06.01
9	2008.06.01
1…	2008.06.01
1…	2008.06.01
1…	2008.06.01
1…	2008.06.01
1…	2008.06.01
1…	2008.06.01
1…	2008.06.01
1…	2008.06.01
1…	2008.06.01

From above query shown in image I retrieve the data that shows the order date converted to ANSI standard format (yyyy.mm.dd – for example 2015.01.31).

4.a) Retrieve customer contact names with middle names if known

You have been asked to write a query that returns a list of customer names. The list must consist of a single field in the format (for example Keith Harris) if the middle name is unknown, or (for example Jane M. Gates) if a middle name is stored in the database.


Notebook-1.ipynb - Azure Data Studio
E: > Other > VIP AINE.AI > Project - 3 (Done) > Notebook-1.ipynb
Kernel SQL
Attach to Select Connection

```
SELECT CONCAT(FirstName,' ',MiddleName,' ',LastName) As CustomerContactName
FROM [SalesLT].[Customer]
```

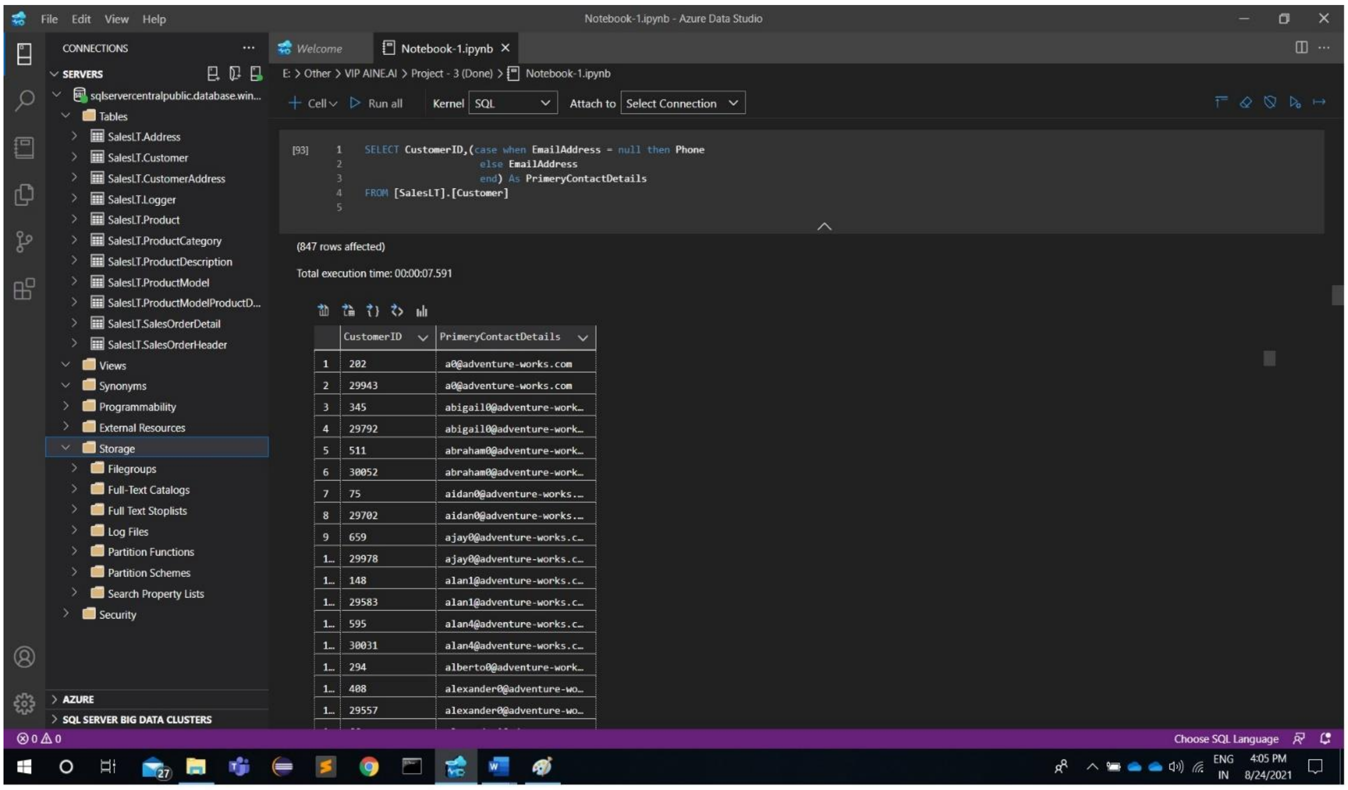
```
(847 rows affected)

Total execution time: 00:00:09.388
```

	CustomerContactName
1	Orlando N. Gee
2	Keith Harris
3	Donna F. Carreras
4	Janet M. Gates
5	Lucy Harrington
6	Rosmarie J. Carroll
7	Dominic P. Gash
8	Kathleen M. Garza
9	Katherine Harding
1...	Johnny A. Caprio
1...	Christopher R. Beck
1...	David J. Liu
1...	John A. Beaver
1...	Jean P. Handley
1...	Jinghao Liu
1...	Linda E. Burnett
1...	Kerim Hanif
1...	Kevin Liu
1...	Donald L. Blanton

From above query shown in image I retrieve customer contacts names with middle names if known.

4.b) Retrieve primary contact details

Customers may provide Adventure Works with an email address, a phone number, or both. If an email address is available, then it should be used as the primary contact method; if not, then the phone number should be used. You must write a query that returns a list of customer IDs in one column, and a second column named PrimaryContact that contains the email address if known, and otherwise the phone number.



```sql
SELECT CustomerID,(case when EmailAddress = null then Phone
                        else EmailAddress
                        end) As PrimeryContactDetails
FROM [SalesLT].[Customer]
```

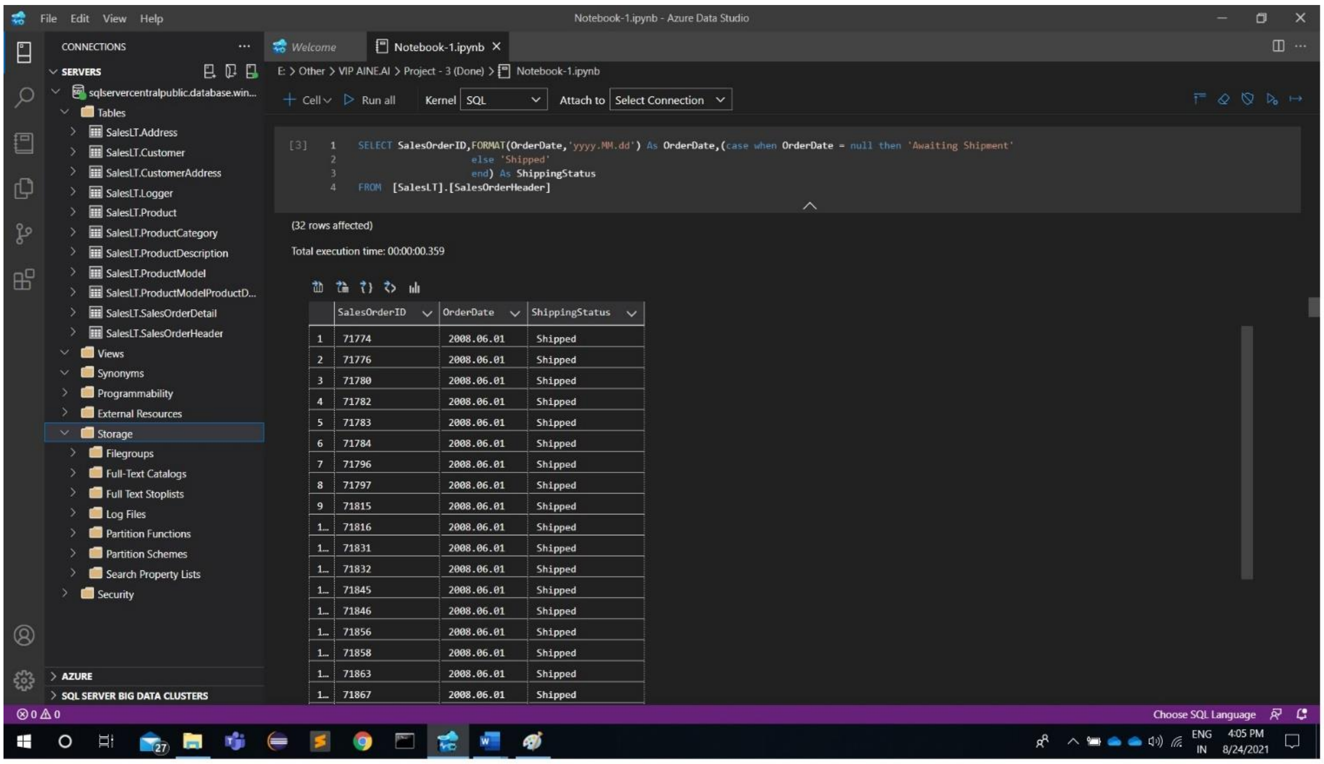
```
(847 rows affected)

Total execution time: 00:00:07.591
```

	CustomerID	PrimeryContactDetails
1	202	a0@adventure-works.com
2	29943	a0@adventure-works.com
3	345	abigail0@adventure-work...
4	29792	abigail0@adventure-work...
5	511	abraham0@adventure-work...
6	30052	abraham0@adventure-work...
7	75	aidan0@adventure-works....
8	29702	aidan0@adventure-works....
9	659	ajay0@adventure-works.c...
1...	29978	ajay0@adventure-works.c...
1...	148	alan1@adventure-works.c...
1...	29583	alan1@adventure-works.c...
1...	595	alan4@adventure-works.c...
1...	30031	alan4@adventure-works.c...
1...	294	alberto0@adventure-work...
1...	408	alexander0@adventure-wo...
1...	29557	alexander0@adventure-wo...

From above query shown in image I retrieve primary contact of customers. If an email address is available, then it should be used as the primary contact method; if not, then the phone number should be used.

4.c) Retrieve shipping status

You have been asked to create a query that returns a list of sales order IDs and order dates with a column named ShippingStatus that contains the text “Shipped” for orders with a known ship date, and “Awaiting Shipment” for orders with no ship date.



```sql
SELECT SalesOrderID,FORMAT(OrderDate,'yyyy.MM.dd') As OrderDate,(case when OrderDate = null then 'Awaiting Shipment'
                    else 'Shipped'
                    end) As ShippingStatus
FROM  [SalesLT].[SalesOrderHeader]
```

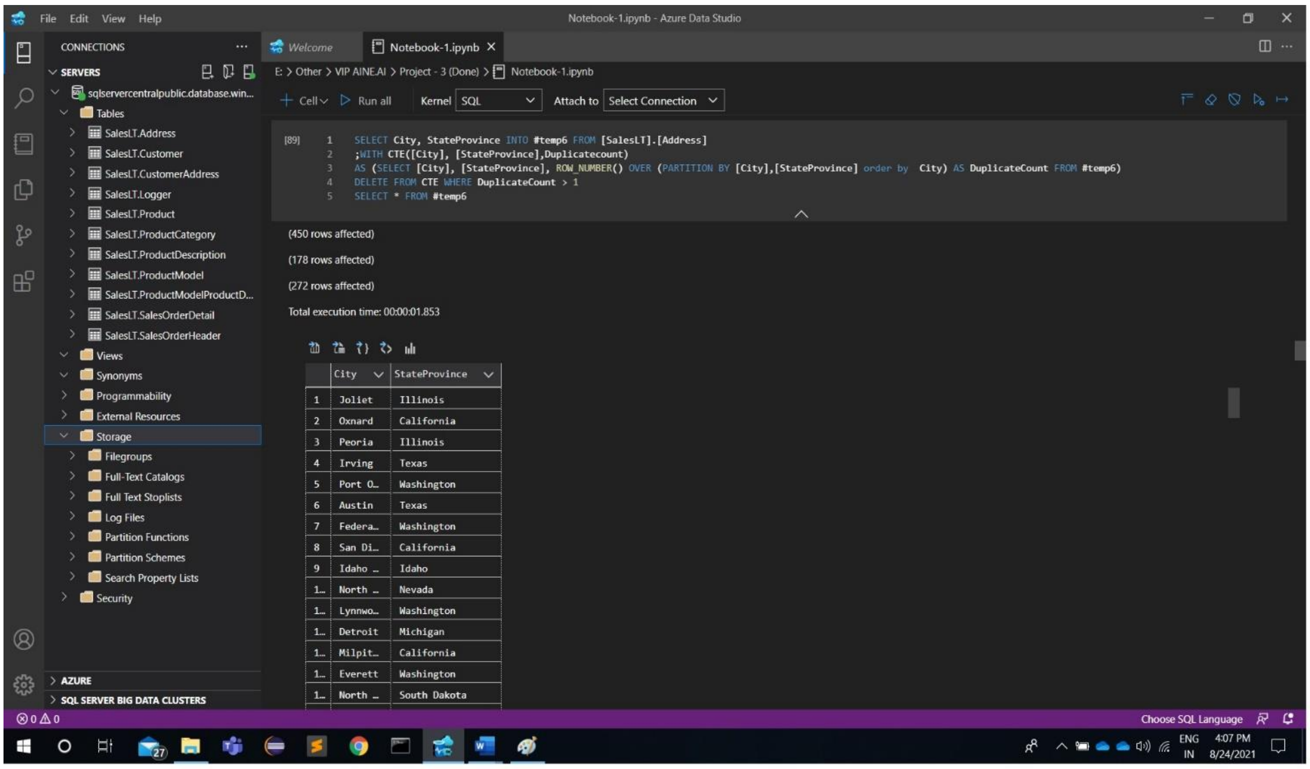
```
(32 rows affected)

Total execution time: 00:00:00.359
```

	SalesOrderID	OrderDate	ShippingStatus
1	71774	2008.06.01	Shipped
2	71776	2008.06.01	Shipped
3	71780	2008.06.01	Shipped
4	71782	2008.06.01	Shipped
5	71783	2008.06.01	Shipped
6	71784	2008.06.01	Shipped
7	71796	2008.06.01	Shipped
8	71797	2008.06.01	Shipped
9	71815	2008.06.01	Shipped
1...	71816	2008.06.01	Shipped
1...	71831	2008.06.01	Shipped
1...	71832	2008.06.01	Shipped
1...	71845	2008.06.01	Shipped
1...	71846	2008.06.01	Shipped
1...	71856	2008.06.01	Shipped
1...	71858	2008.06.01	Shipped
1...	71863	2008.06.01	Shipped
1...	71867	2008.06.01	Shipped

As query shown in image I retrieve shipping status. I return a list of sales OrderID and order dates with a column named ShippingStatus that contains the text “Shipped” for orders with a known ship date, and “Awaiting Shipment” for orders with no ship date.

5.a) Retrieve a list of cities

Initially, you need to produce a list of all of your customers' locations. Write a Transact-SQL query that queries the Address table and retrieves all values for City and StateProvince, removing duplicates.



```sql
SELECT City, StateProvince INTO #temp6 FROM [SalesLT].[Address]
;WITH CTE([City], [StateProvince],Duplicatecount)
AS (SELECT [City], [StateProvince], ROW_NUMBER() OVER (PARTITION BY [City],[StateProvince] order by  City) AS DuplicateCount FROM #temp6)
DELETE FROM CTE WHERE DuplicateCount > 1
SELECT * FROM #temp6
```

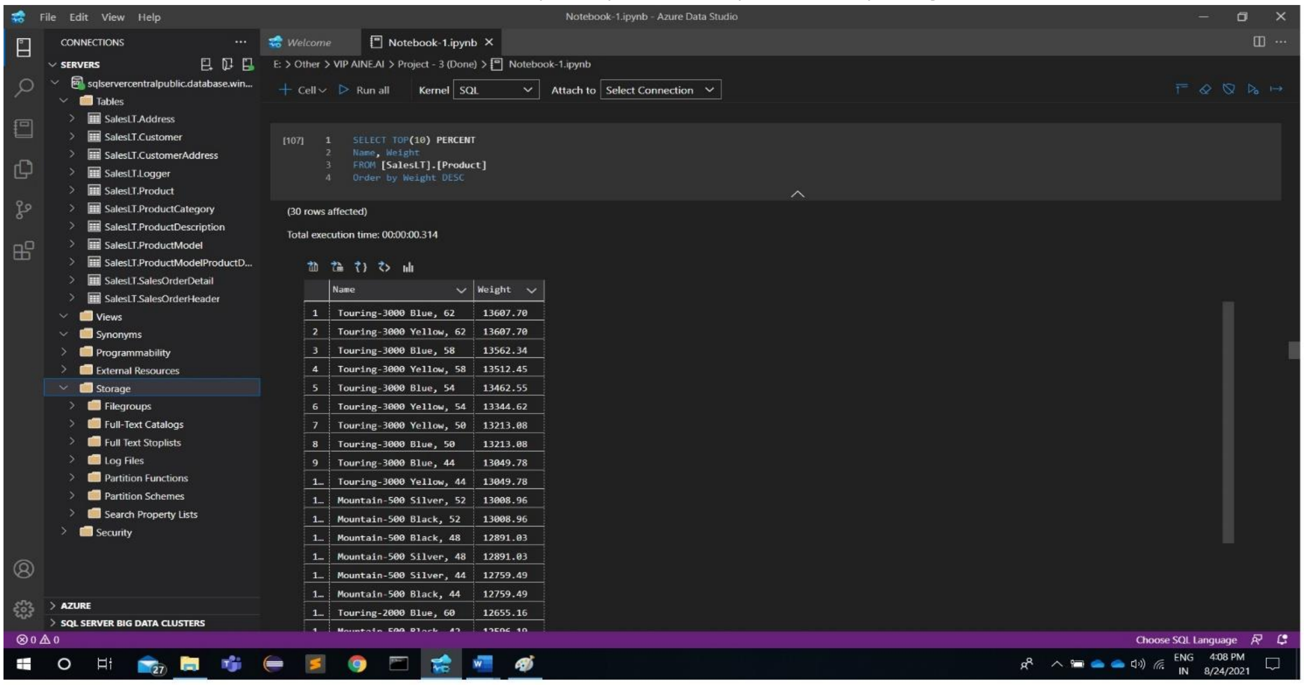
```
(450 rows affected)

(178 rows affected)

(272 rows affected)

Total execution time: 00:00:01.853
```

	City	StateProvince
1	Joliet	Illinois
2	Oxnard	California
3	Peoria	Illinois
4	Irving	Texas
5	Port O…	Washington
6	Austin	Texas
7	Federa…	Washington
8	San Di…	California
9	Idaho …	Idaho
1…	North …	Nevada
1…	Lynnwo…	Washington
1…	Detroit	Michigan
1…	Milpit…	California
1…	Everett	Washington
1…	North …	South Dakota

As query shown in image I retrieve the list of cities. and retrieves all values for City and StateProvince, removing duplicates.

5.b) Retrieve the heaviest products

Transportation costs are increasing, and you need to identify the heaviest products. Retrieve the names of the top ten percent of products by weight.



```sql
SELECT TOP(10) PERCENT
Name, Weight
FROM [SalesLT].[Product]
Order by Weight DESC
```

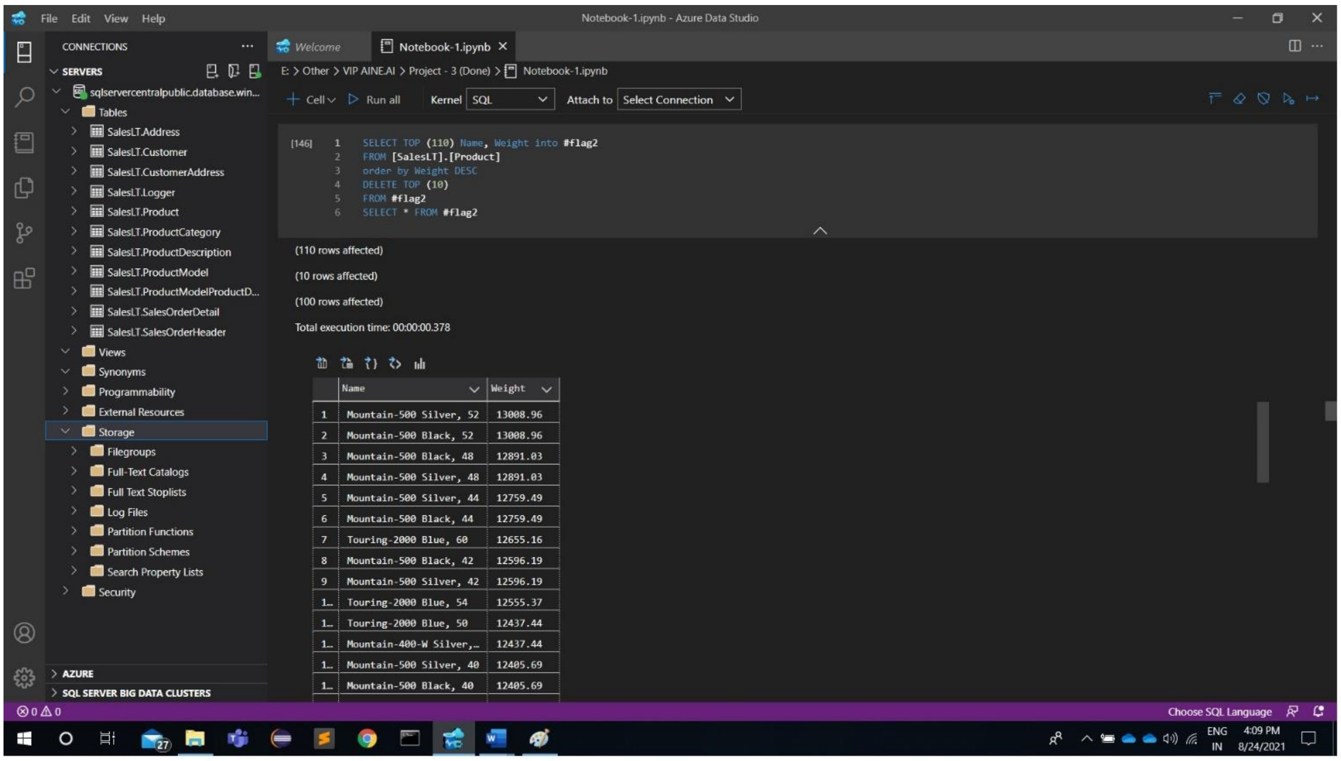
```
(30 rows affected)

Total execution time: 00:00:00.314
```

	Name	Weight
1	Touring-3000 Blue, 62	13607.70
2	Touring-3000 Yellow, 62	13607.70
3	Touring-3000 Blue, 58	13562.34
4	Touring-3000 Yellow, 58	13512.45
5	Touring-3000 Blue, 54	13462.55
6	Touring-3000 Yellow, 54	13344.62
7	Touring-3000 Yellow, 50	13213.08
8	Touring-3000 Blue, 50	13213.08
9	Touring-3000 Blue, 44	13049.78
1…	Touring-3000 Yellow, 44	13049.78
1…	Mountain-500 Silver, 52	13008.96
1…	Mountain-500 Black, 52	13008.96
1…	Mountain-500 Black, 48	12891.03
1…	Mountain-500 Silver, 48	12891.03
1…	Mountain-500 Silver, 44	12759.49
1…	Mountain-500 Black, 44	12759.49
1…	Touring-2000 Blue, 60	12655.16

As query shown in the image I retrieve the heaviest products from product table. Retrieves the names of the top ten percent of products by weight.

5.c) Retrieve the heaviest 100 products not including the heaviest ten

The heaviest ten products are transported by a specialist carrier; therefore, you need to modify the previous query to list the heaviest 100 products not including the heaviest ten.



```sql
SELECT TOP (110) Name, Weight into #flag2
FROM [SalesLT].[Product]
order by Weight DESC
DELETE TOP (10)
FROM #flag2
SELECT * FROM #flag2
```

```
(110 rows affected)

(10 rows affected)

(100 rows affected)

Total execution time: 00:00:00.378
```

	Name	Weight
1	Mountain-500 Silver, 52	13008.96
2	Mountain-500 Black, 52	13008.96
3	Mountain-500 Black, 48	12891.03
4	Mountain-500 Silver, 48	12891.03
5	Mountain-500 Silver, 44	12759.49
6	Mountain-500 Black, 44	12759.49
7	Touring-2000 Blue, 60	12655.16
8	Mountain-500 Black, 42	12596.19
9	Mountain-500 Silver, 42	12596.19
1...	Touring-2000 Blue, 54	12555.37
1...	Touring-2000 Blue, 50	12437.44
1...	Mountain-400-W Silver,...	12437.44
1...	Mountain-500 Silver, 40	12405.69
1...	Mountain-500 Black, 40	12405.69

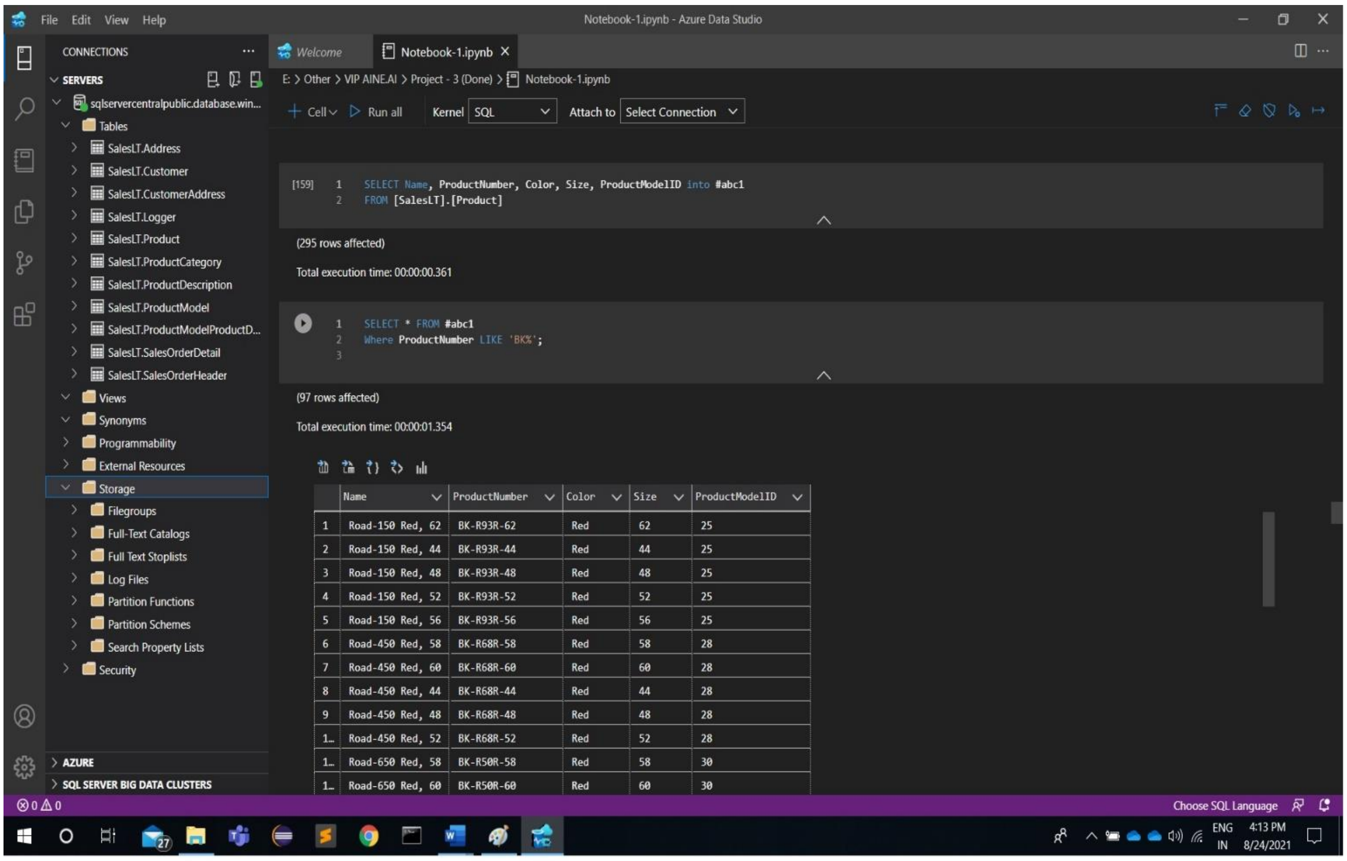
As query shown in image I retrieve the heaviest 100 products not including the heaviest ten.

5.d) Retrieve product details for product model 1

Initially, you need to find the names, colors, and sizes of the products with a product model ID 1.

Filter products by color and size

Retrieve the product number and name of the products that have a color of 'black', 'red', or 'white' and a size of 'S' or 'M'.



```
[159]
SELECT Name, ProductNumber, Color, Size, ProductModelID into #abc1
FROM [SalesLT].[Product]
```

```
(295 rows affected)

Total execution time: 00:00:00.361
```

```
SELECT * FROM #abc1
Where ProductNumber LIKE 'BK%';

```

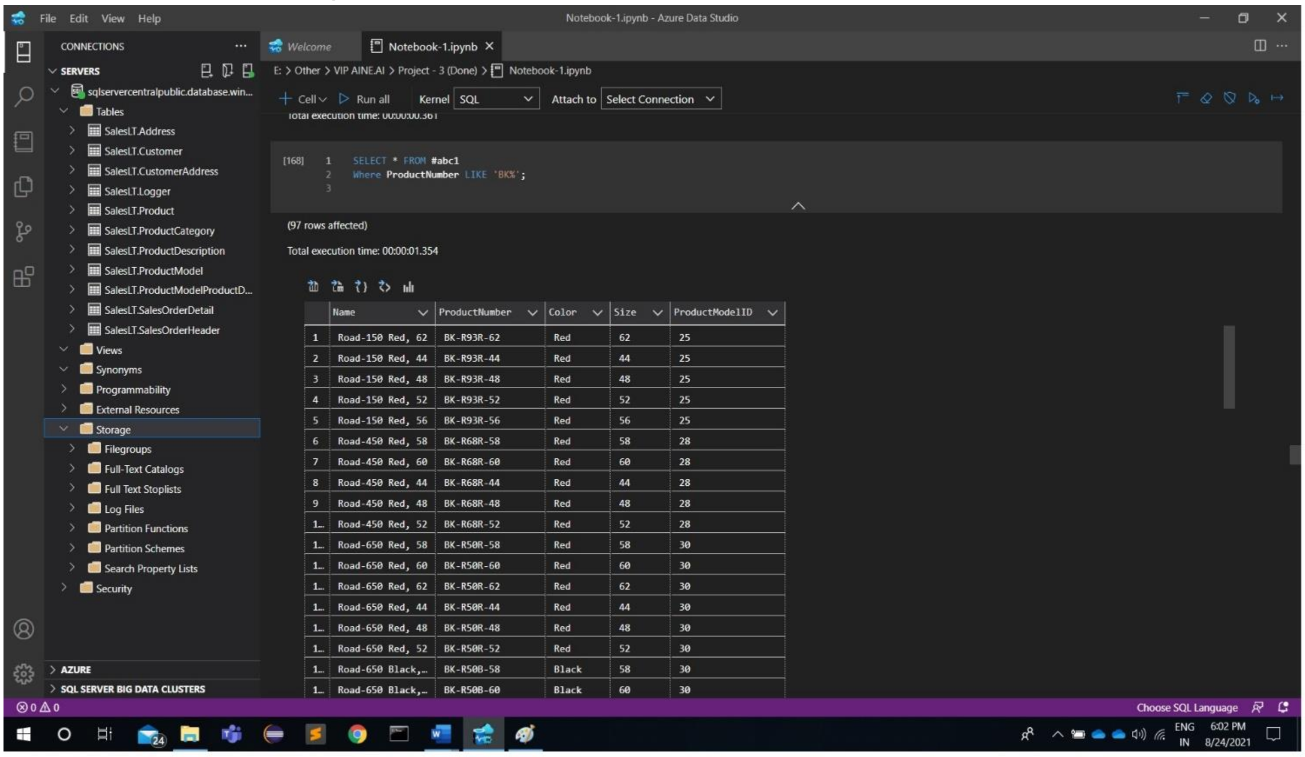
```
(97 rows affected)

Total execution time: 00:00:01.354
```

	Name	ProductNumber	Color	Size	ProductModelID
1	Road-150 Red, 62	BK-R93R-62	Red	62	25
2	Road-150 Red, 44	BK-R93R-44	Red	44	25
3	Road-150 Red, 48	BK-R93R-48	Red	48	25
4	Road-150 Red, 52	BK-R93R-52	Red	52	25
5	Road-150 Red, 56	BK-R93R-56	Red	56	25
6	Road-450 Red, 58	BK-R68R-58	Red	58	28
7	Road-450 Red, 60	BK-R68R-60	Red	60	28
8	Road-450 Red, 44	BK-R68R-44	Red	44	28
9	Road-450 Red, 48	BK-R68R-48	Red	48	28
1…	Road-450 Red, 52	BK-R68R-52	Red	52	28
1…	Road-650 Red, 58	BK-R50R-58	Red	58	30
1…	Road-650 Red, 60	BK-R50R-60	Red	60	30

As query shown in image I retrieve the product number and name of the products that have a colour of ‘black’, ‘red’, or ‘white’ and a size of ‘S’ or ‘M’.

5.e) Filter products by product number

Retrieve the product number, name, and list price of products whose product number begins 'BK-'.



```
[168]
SELECT * FROM #abc1
Where ProductNumber LIKE 'BK%';
```

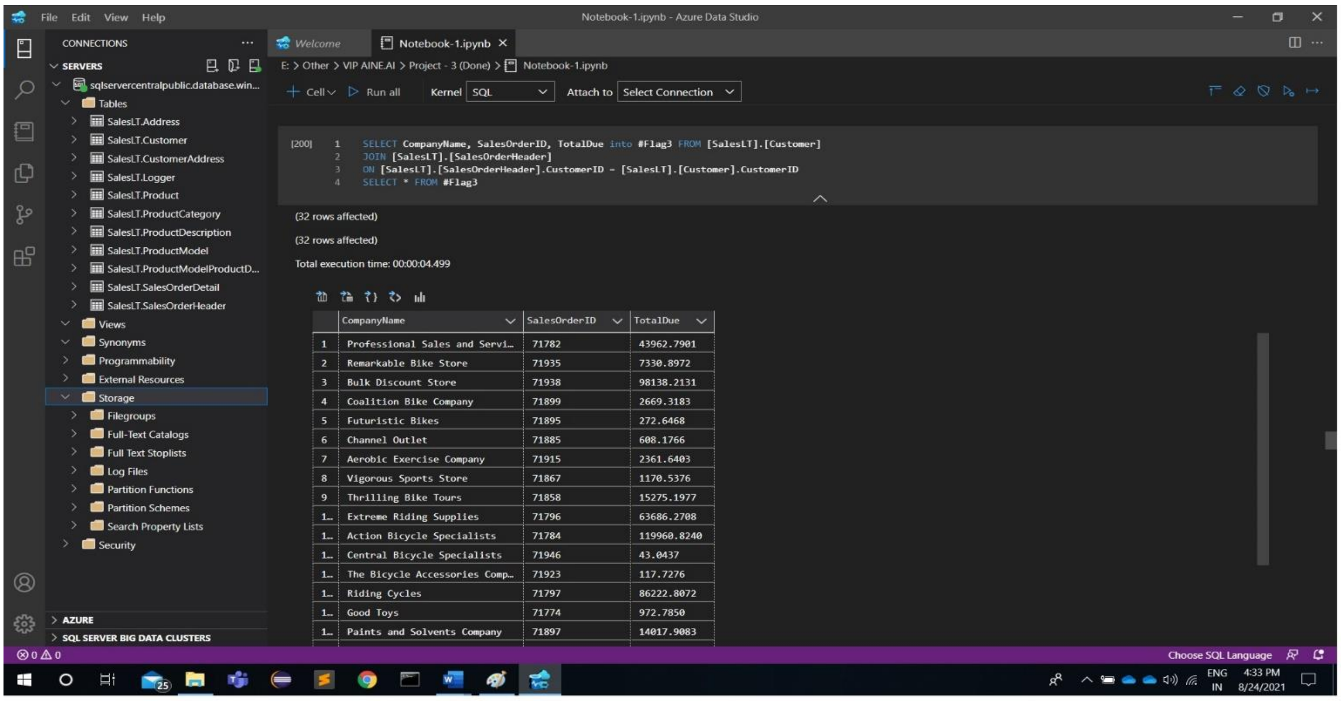
```
(97 rows affected)

Total execution time: 00:00:01.354
```

	Name	ProductNumber	Color	Size	ProductModelID
1	Road-150 Red, 62	BK-R93R-62	Red	62	25
2	Road-150 Red, 44	BK-R93R-44	Red	44	25
3	Road-150 Red, 48	BK-R93R-48	Red	48	25
4	Road-150 Red, 52	BK-R93R-52	Red	52	25
5	Road-150 Red, 56	BK-R93R-56	Red	56	25
6	Road-450 Red, 58	BK-R68R-58	Red	58	28
7	Road-450 Red, 60	BK-R68R-60	Red	60	28
8	Road-450 Red, 44	BK-R68R-44	Red	44	28
9	Road-450 Red, 48	BK-R68R-48	Red	48	28
1...	Road-450 Red, 52	BK-R68R-52	Red	52	28
1...	Road-650 Red, 58	BK-R50R-58	Red	58	30
1...	Road-650 Red, 60	BK-R50R-60	Red	60	30
1...	Road-650 Red, 62	BK-R50R-62	Red	62	30
1...	Road-650 Red, 44	BK-R50R-44	Red	44	30
1...	Road-650 Red, 48	BK-R50R-48	Red	48	30
1...	Road-650 Red, 52	BK-R50R-52	Red	52	30
1...	Road-650 Black,...	BK-R50B-58	Black	58	30
1...	Road-650 Black,...	BK-R50B-60	Black	60	30

From above query I filter products by product number. I retrieve the product number, name, and list price of product name begins 'BK-'

6.a) Retrieve customer orders to generate invoice reports

As an initial step towards generating the invoice report, write a query that returns the company name from the SalesLT.Customer table, and the sales order ID and total due from the SalesLT.SalesOrderHeader table.



```
[200]
SELECT CompanyName, SalesOrderID, TotalDue into #Flag3 FROM [SalesLT].[Customer]
JOIN [SalesLT].[SalesOrderHeader]
ON [SalesLT].[SalesOrderHeader].CustomerID = [SalesLT].[Customer].CustomerID
SELECT * FROM #Flag3
```

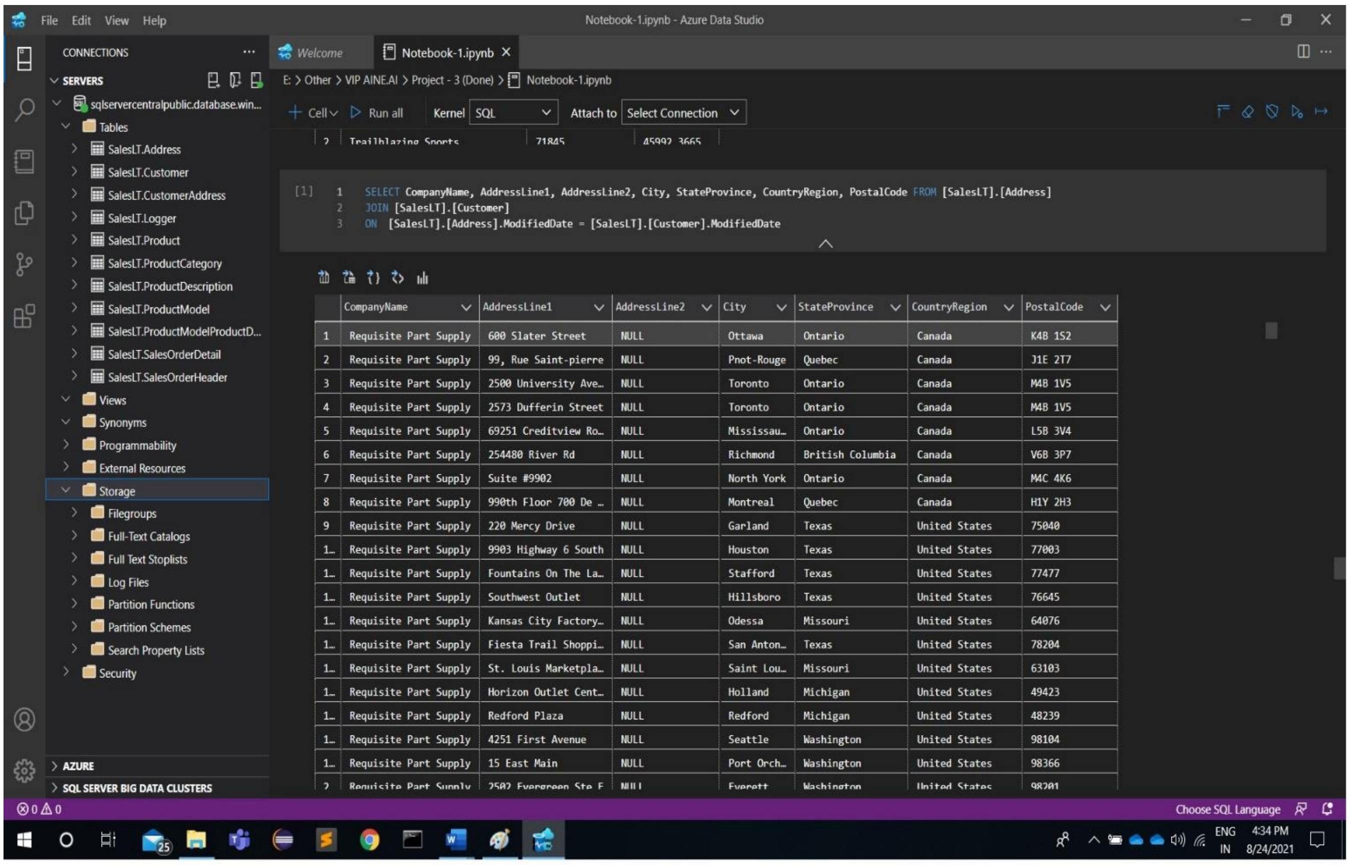
```
(32 rows affected)

(32 rows affected)

Total execution time: 00:00:04.499
```

	CompanyName	SalesOrderID	TotalDue
1	Professional Sales and Servi...	71782	43962.7901
2	Remarkable Bike Store	71935	7330.8972
3	Bulk Discount Store	71938	98138.2131
4	Coalition Bike Company	71899	2669.3183
5	Futuristic Bikes	71895	272.6468
6	Channel Outlet	71885	608.1766
7	Aerobic Exercise Company	71915	2361.6403
8	Vigorous Sports Store	71867	1170.5376
9	Thrilling Bike Tours	71858	15275.1977
1...	Extreme Riding Supplies	71796	63686.2708
1...	Action Bicycle Specialists	71784	119960.8240
1...	Central Bicycle Specialists	71946	43.0437
1...	The Bicycle Accessories Comp...	71923	117.7276
1...	Riding Cycles	71797	86222.8072
1...	Good Toys	71774	972.7850
1...	Paints and Solvents Company	71897	14017.9083

From query shown in image I retrieve the customer orders to generate the invoice report. As an initial step towards generating the invoice report, I write query that returns the company name from the SalesLT.Customer table, and the sales order ID and total due from the SalesLT.SalesOrderHeader table.

6.b) Retrieve customer orders with addresses

Extend your customer orders query to include the Main Office address for each customer, including the full street address, city, state or province, postal code, and country or region.



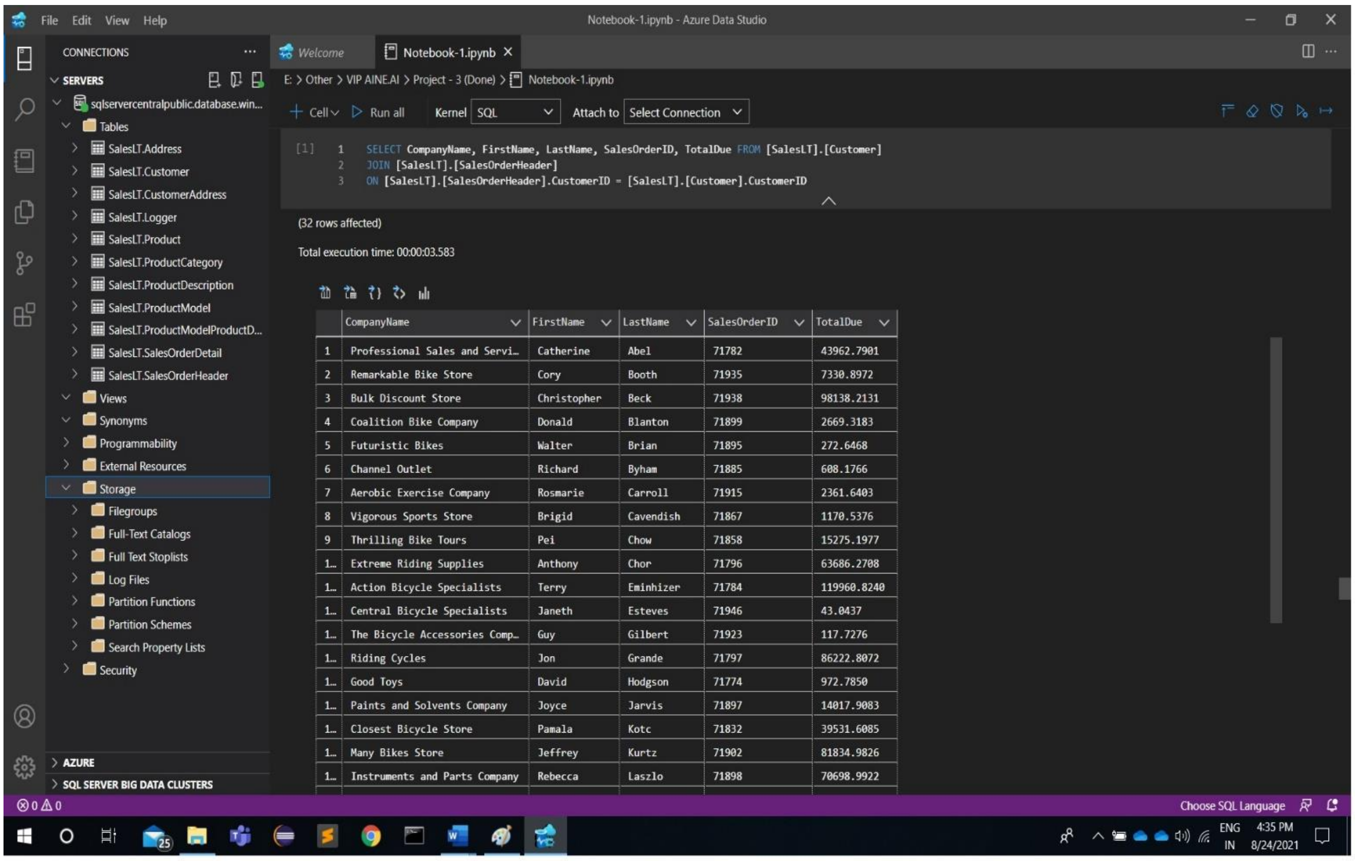
```
[1]
SELECT CompanyName, AddressLine1, AddressLine2, City, StateProvince, CountryRegion, PostalCode FROM [SalesLT].[Address]
JOIN [SalesLT].[Customer]
ON  [SalesLT].[Address].ModifiedDate = [SalesLT].[Customer].ModifiedDate
```

	CompanyName	AddressLine1	AddressLine2	City	StateProvince	CountryRegion	PostalCode
1	Requisite Part Supply	600 Slater Street	NULL	Ottawa	Ontario	Canada	K4B 1S2
2	Requisite Part Supply	99, Rue Saint-pierre	NULL	Pnot-Rouge	Quebec	Canada	J1E 2T7
3	Requisite Part Supply	2500 University Ave…	NULL	Toronto	Ontario	Canada	M4B 1V5
4	Requisite Part Supply	2573 Dufferin Street	NULL	Toronto	Ontario	Canada	M4B 1V5
5	Requisite Part Supply	69251 Creditview Ro…	NULL	Mississau…	Ontario	Canada	L5B 3V4
6	Requisite Part Supply	254480 River Rd	NULL	Richmond	British Columbia	Canada	V6B 3P7
7	Requisite Part Supply	Suite #9902	NULL	North York	Ontario	Canada	M4C 4K6
8	Requisite Part Supply	990th Floor 700 De …	NULL	Montreal	Quebec	Canada	H1Y 2H3
9	Requisite Part Supply	220 Mercy Drive	NULL	Garland	Texas	United States	75040
1…	Requisite Part Supply	9903 Highway 6 South	NULL	Houston	Texas	United States	77003
1…	Requisite Part Supply	Fountains On The La…	NULL	Stafford	Texas	United States	77477
1…	Requisite Part Supply	Southwest Outlet	NULL	Hillsboro	Texas	United States	76645
1…	Requisite Part Supply	Kansas City Factory…	NULL	Odessa	Missouri	United States	64076
1…	Requisite Part Supply	Fiesta Trail Shoppi…	NULL	San Anton…	Texas	United States	78204
1…	Requisite Part Supply	St. Louis Marketpla…	NULL	Saint Lou…	Missouri	United States	63103
1…	Requisite Part Supply	Horizon Outlet Cent…	NULL	Holland	Michigan	United States	49423
1…	Requisite Part Supply	Redford Plaza	NULL	Redford	Michigan	United States	48239
1…	Requisite Part Supply	4251 First Avenue	NULL	Seattle	Washington	United States	98104
1…	Requisite Part Supply	15 East Main	NULL	Port Orch…	Washington	United States	98366
2	Requisite Part Supply	2502 Evergreen Ste E	NULL	Everett	Washington	United States	98201

From above query show in image I retrieve customer orders with addresses including the full street address, city, state or province, postal code, and country or region.

6.c) Retrieve a list of all customers and their orders

The sales manager wants a list of all customer companies and their contacts (first name and last name), showing the sales order ID and total due for each order they have placed. Customers who have not placed any orders should be included at the bottom of the list with NULL values for the order ID and total due.



```sql
SELECT CompanyName, FirstName, LastName, SalesOrderID, TotalDue FROM [SalesLT].[Customer]
JOIN [SalesLT].[SalesOrderHeader]
ON [SalesLT].[SalesOrderHeader].CustomerID = [SalesLT].[Customer].CustomerID
```

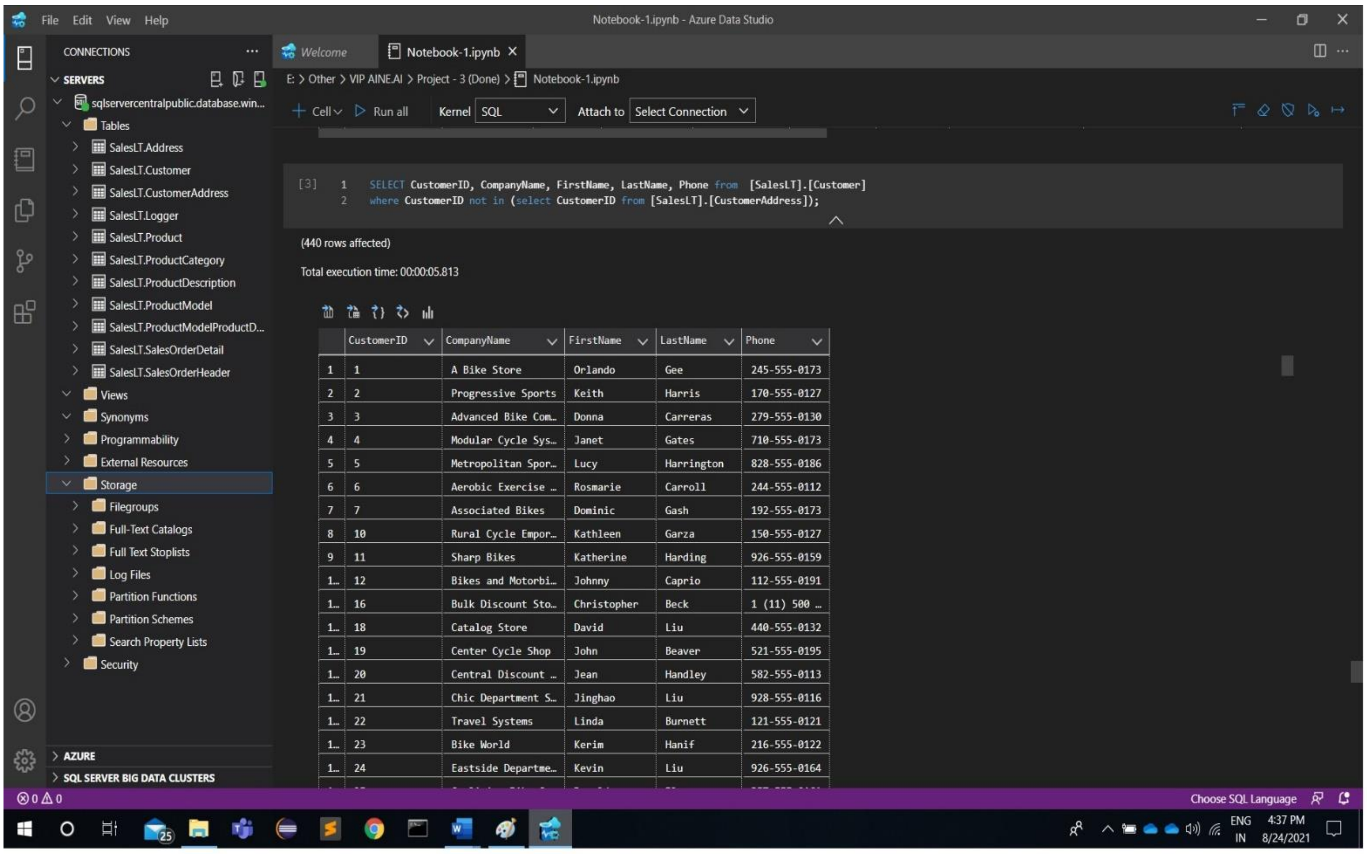
```
(32 rows affected)

Total execution time: 00:00:03.583
```

	CompanyName	FirstName	LastName	SalesOrderID	TotalDue
1	Professional Sales and Servi…	Catherine	Abel	71782	43962.7901
2	Remarkable Bike Store	Cory	Booth	71935	7330.8972
3	Bulk Discount Store	Christopher	Beck	71938	98138.2131
4	Coalition Bike Company	Donald	Blanton	71899	2669.3183
5	Futuristic Bikes	Walter	Brian	71895	272.6468
6	Channel Outlet	Richard	Byham	71885	608.1766
7	Aerobic Exercise Company	Rosmarie	Carroll	71915	2361.6403
8	Vigorous Sports Store	Brigid	Cavendish	71867	1170.5376
9	Thrilling Bike Tours	Pei	Chow	71858	15275.1977
1…	Extreme Riding Supplies	Anthony	Chor	71796	63686.2708
1…	Action Bicycle Specialists	Terry	Eminhizer	71784	119960.8240
1…	Central Bicycle Specialists	Janeth	Esteves	71946	43.0437
1…	The Bicycle Accessories Comp…	Guy	Gilbert	71923	117.7276
1…	Riding Cycles	Jon	Grande	71797	86222.8072
1…	Good Toys	David	Hodgson	71774	972.7850
1…	Paints and Solvents Company	Joyce	Jarvis	71897	14017.9083
1…	Closest Bicycle Store	Pamala	Kotc	71832	39531.6085
1…	Many Bikes Store	Jeffrey	Kurtz	71902	81834.9826
1…	Instruments and Parts Company	Rebecca	Laszlo	71898	70698.9922

As query shown in image I retrieve a list of all customers and their orders. I retrieve a list of all customer companies and their contacts (first name and last name), showing the sales order ID and total due for each order they have placed.

6.d) Retrieve a list of customers with no address

A sales employee has noticed that AdventureWorks does not have address information for all customers. You must write a query that returns a list of customer IDs, company names, contact names (first name and last name), and phone numbers for customers with no address stored in the database.



```sql
SELECT CustomerID, CompanyName, FirstName, LastName, Phone from  [SalesLT].[Customer]
where CustomerID not in (select CustomerID from [SalesLT].[CustomerAddress]);
```

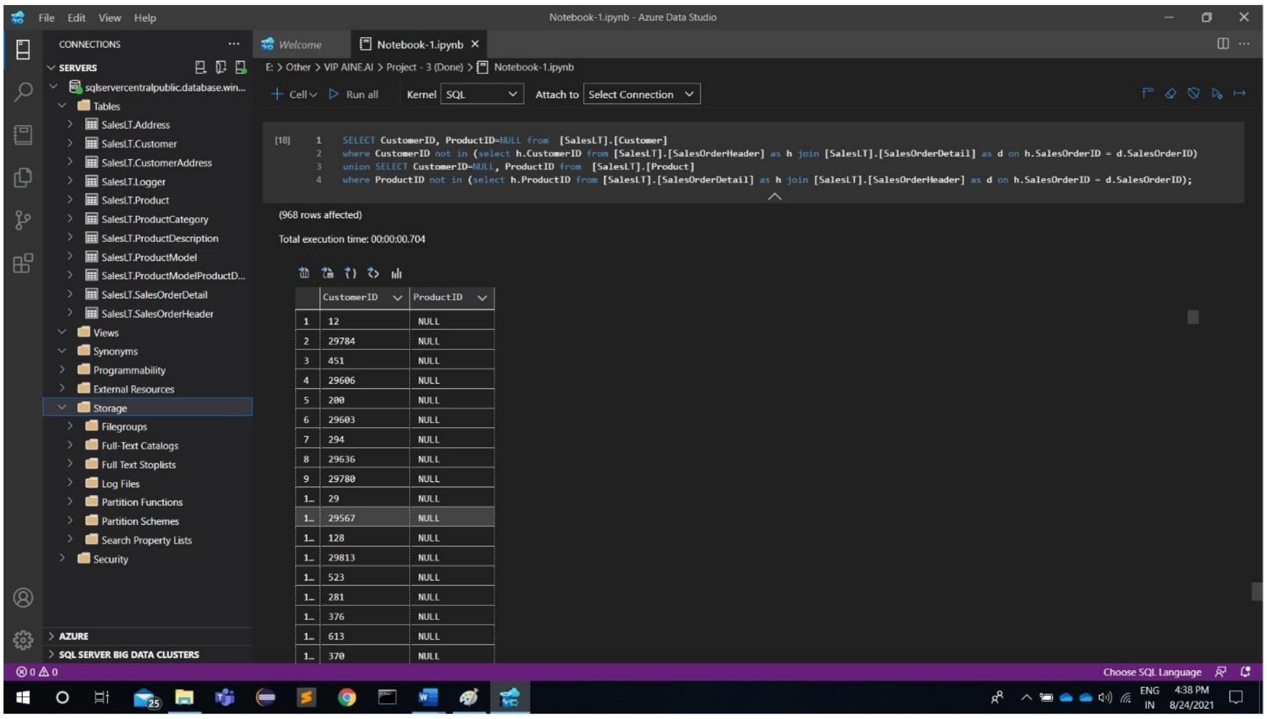
```
(440 rows affected)

Total execution time: 00:00:05.813
```

	CustomerID	CompanyName	FirstName	LastName	Phone
1	1	A Bike Store	Orlando	Gee	245-555-0173
2	2	Progressive Sports	Keith	Harris	170-555-0127
3	3	Advanced Bike Com…	Donna	Carreras	279-555-0130
4	4	Modular Cycle Sys…	Janet	Gates	710-555-0173
5	5	Metropolitan Spor…	Lucy	Harrington	828-555-0186
6	6	Aerobic Exercise …	Rosmarie	Carroll	244-555-0112
7	7	Associated Bikes	Dominic	Gash	192-555-0173
8	10	Rural Cycle Empor…	Kathleen	Garza	150-555-0127
9	11	Sharp Bikes	Katherine	Harding	926-555-0159
1…	12	Bikes and Motorbi…	Johnny	Caprio	112-555-0191
1…	16	Bulk Discount Sto…	Christopher	Beck	1 (11) 500 …
1…	18	Catalog Store	David	Liu	440-555-0132
1…	19	Center Cycle Shop	John	Beaver	521-555-0195
1…	20	Central Discount …	Jean	Handley	582-555-0113
1…	21	Chic Department S…	Jinghao	Liu	928-555-0116
1…	22	Travel Systems	Linda	Burnett	121-555-0121
1…	23	Bike World	Kerim	Hanif	216-555-0122
1…	24	Eastside Departme…	Kevin	Liu	926-555-0164

From above query shown in image I retrieve a list of customer with no address.

6.e) Retrieve a list of customers and products without orders

Some customers have never placed orders, and some products have never been ordered. Create a query that returns a column of customer IDs for customers who have never placed an order, and a column of product IDs for products that have never been ordered. Each row with a customer ID should have a NULL product ID (because the customer has never ordered a product) and each row with a product ID should have a NULL customer ID (because the product has never been ordered by a customer).

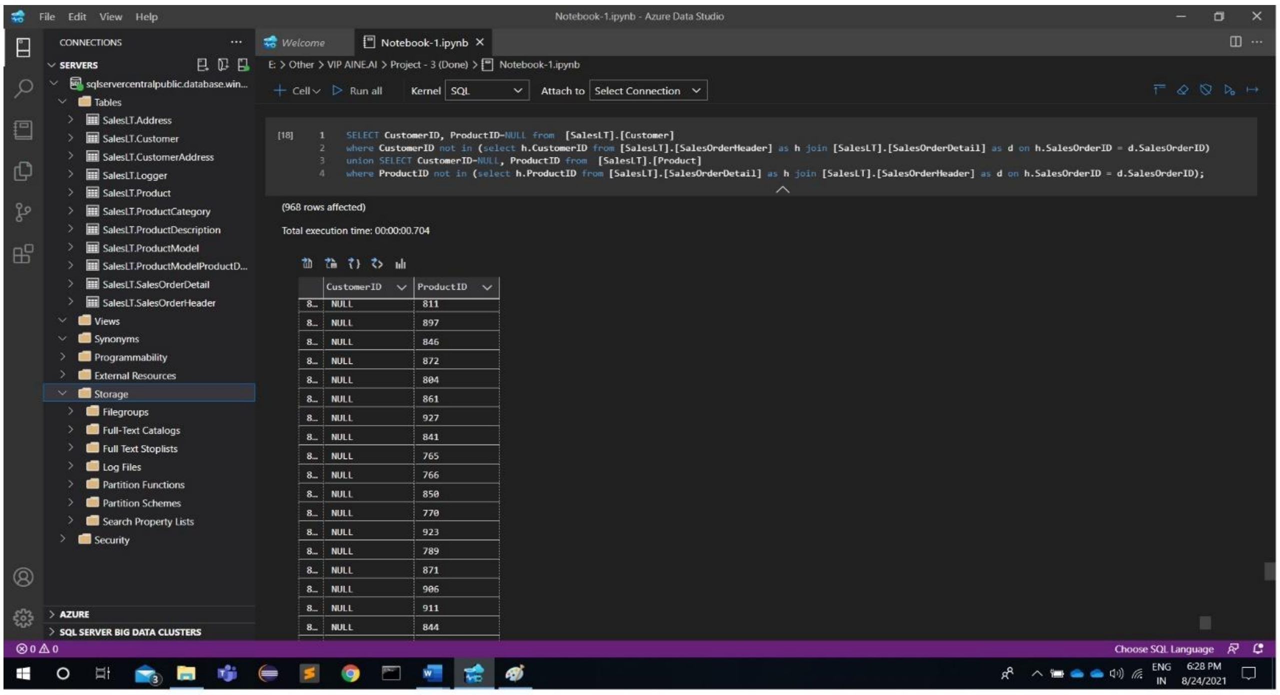


```sql
SELECT CustomerID, ProductID=NULL from  [SalesLT].[Customer]
where CustomerID not in (select h.CustomerID from [SalesLT].[SalesOrderHeader] as h join [SalesLT].[SalesOrderDetail] as d on h.SalesOrderID = d.SalesOrderID)
union SELECT CustomerID=NULL, ProductID from  [SalesLT].[Product]
where ProductID not in (select h.ProductID from [SalesLT].[SalesOrderDetail] as h join [SalesLT].[SalesOrderHeader] as d on h.SalesOrderID = d.SalesOrderID);
```

```
(968 rows affected)

Total execution time: 00:00:00.704
```

	CustomerID	ProductID
1	12	NULL
2	29784	NULL
3	451	NULL
4	29606	NULL
5	200	NULL
6	29603	NULL
7	294	NULL
8	29636	NULL
9	29780	NULL
1...	29	NULL
1...	29567	NULL
1...	128	NULL
1...	29813	NULL
1...	523	NULL
1...	281	NULL
1...	376	NULL
1...	613	NULL
1...	370	NULL

From above query shown in image I retrieve list of customers and products without orders. There is null values in productID column as shown in output.



```sql
SELECT CustomerID, ProductID=NULL from  [SalesLT].[Customer]
where CustomerID not in (select h.CustomerID from [SalesLT].[SalesOrderHeader] as h join [SalesLT].[SalesOrderDetail] as d on h.SalesOrderID = d.SalesOrderID)
union SELECT CustomerID=NULL, ProductID from  [SalesLT].[Product]
where ProductID not in (select h.ProductID from [SalesLT].[SalesOrderDetail] as h join [SalesLT].[SalesOrderHeader] as d on h.SalesOrderID = d.SalesOrderID);
```

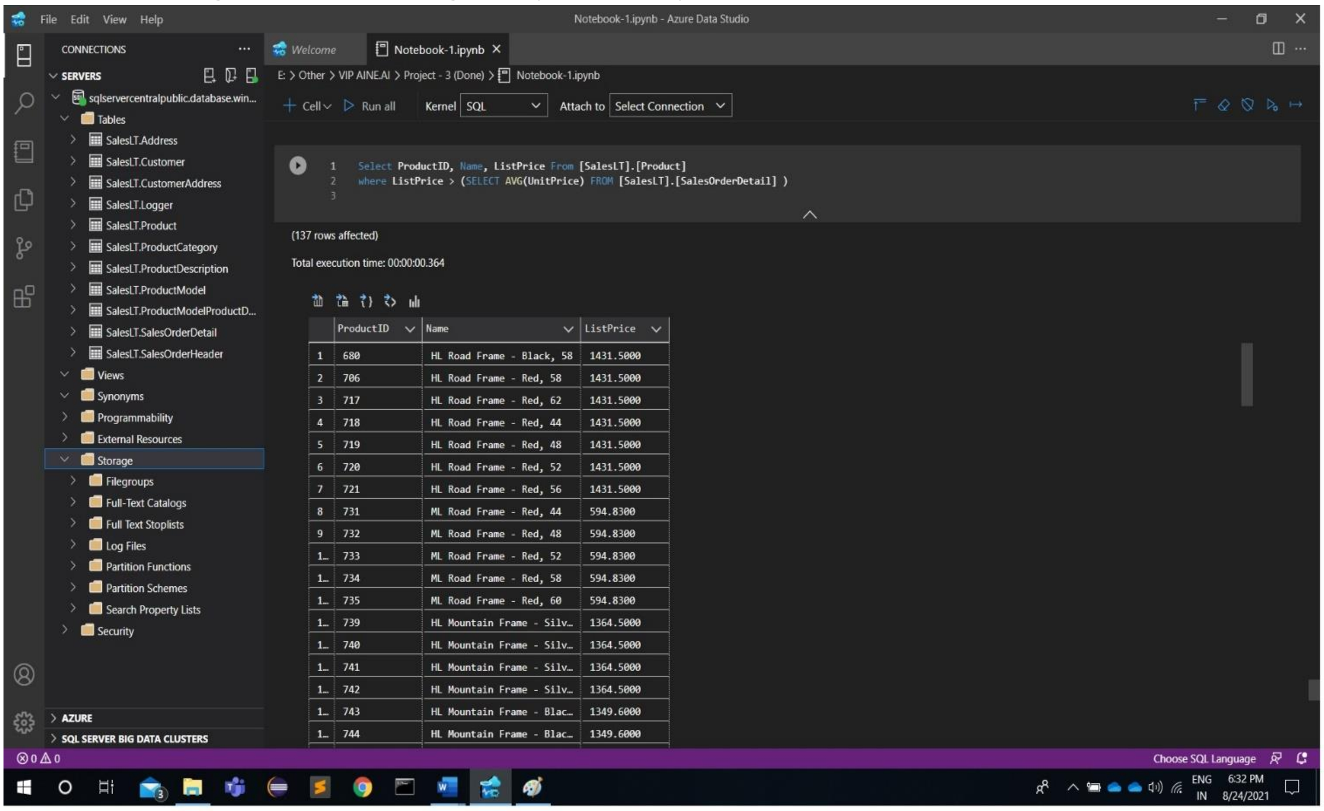
```
(968 rows affected)

Total execution time: 00:00:00.704
```

	CustomerID	ProductID
8...	NULL	811
8...	NULL	897
8...	NULL	846
8...	NULL	872
8...	NULL	804
8...	NULL	861
8...	NULL	927
8...	NULL	841
8...	NULL	765
8...	NULL	766
8...	NULL	850
8...	NULL	770
8...	NULL	923
8...	NULL	789
8...	NULL	871
8...	NULL	906
8...	NULL	911
8...	NULL	844

In this image there is null values in output of customerID column as shown in above image.

7.a) Retrieve products whose list price is higher than the average unit price

Retrieve the product ID, name, and list price for each product where the list price is higher than the average unit price for all products that have been sold.



```sql
Select ProductID, Name, ListPrice From [SalesLT].[Product]
where ListPrice > (SELECT AVG(UnitPrice) FROM [SalesLT].[SalesOrderDetail] )
```

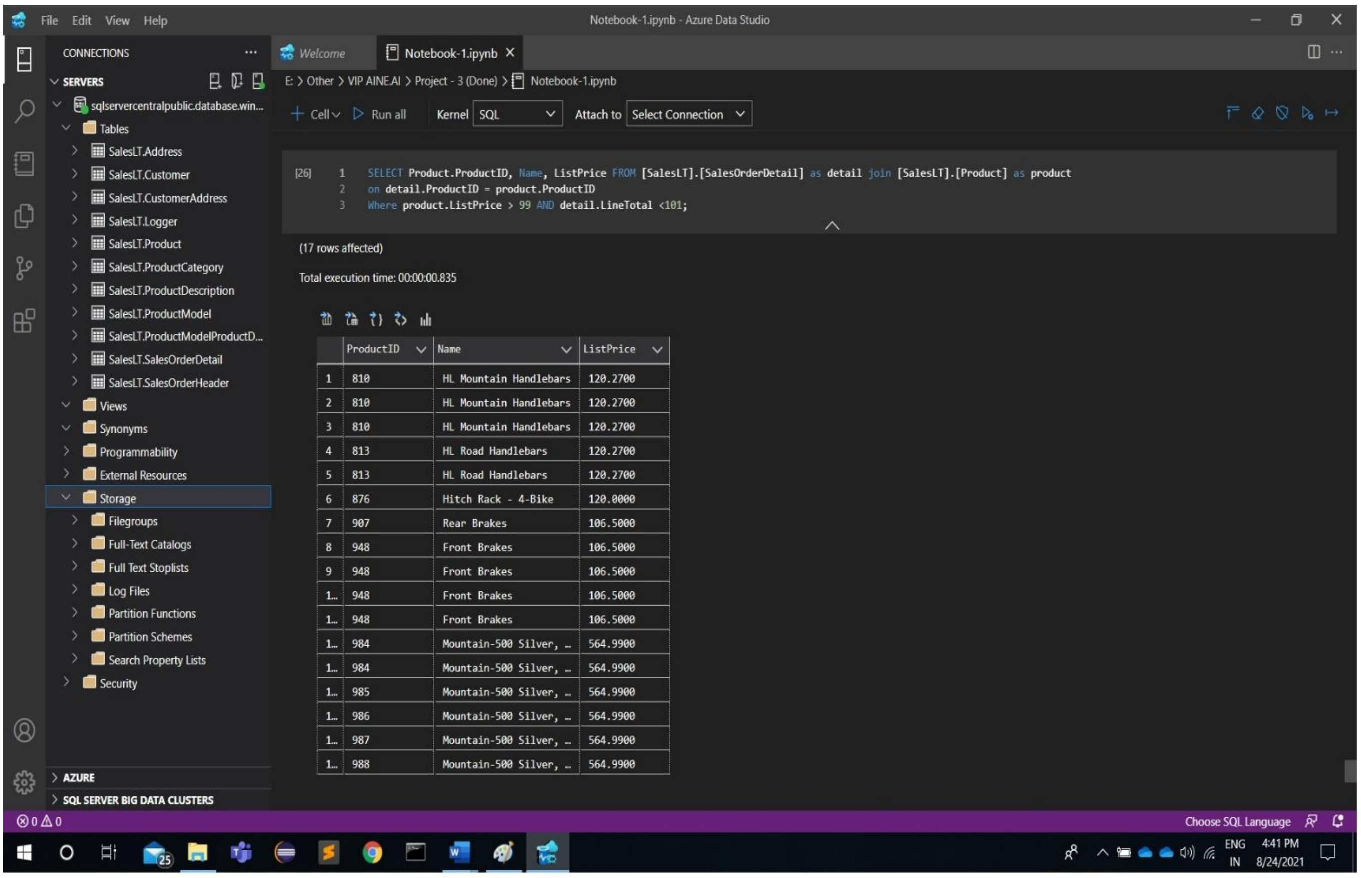
```
(137 rows affected)

Total execution time: 00:00:00.364
```

	ProductID	Name	ListPrice
1	680	HL Road Frame - Black, 58	1431.5000
2	706	HL Road Frame - Red, 58	1431.5000
3	717	HL Road Frame - Red, 62	1431.5000
4	718	HL Road Frame - Red, 44	1431.5000
5	719	HL Road Frame - Red, 48	1431.5000
6	720	HL Road Frame - Red, 52	1431.5000
7	721	HL Road Frame - Red, 56	1431.5000
8	731	ML Road Frame - Red, 44	594.8300
9	732	ML Road Frame - Red, 48	594.8300
1…	733	ML Road Frame - Red, 52	594.8300
1…	734	ML Road Frame - Red, 58	594.8300
1…	735	ML Road Frame - Red, 60	594.8300
1…	739	HL Mountain Frame - Silv…	1364.5000
1…	740	HL Mountain Frame - Silv…	1364.5000
1…	741	HL Mountain Frame - Silv…	1364.5000
1…	742	HL Mountain Frame - Silv…	1364.5000
1…	743	HL Mountain Frame - Blac…	1349.6000
1…	744	HL Mountain Frame - Blac…	1349.6000

As query shown in image I retrieve products whose list price is higher than the average unit price. Retrieve the product ID, name, and list price for each product where the list price is higher than the average unit price.

7.b) Retrieve Products with a list price of $100 or more that have been sold for less than $100

Retrieve the product ID, name, and list price for each product where the list price is $100 or more, and the product has been sold for less than $100.



```sql
SELECT Product.ProductID, Name, ListPrice FROM [SalesLT].[SalesOrderDetail] as detail join [SalesLT].[Product] as product
on detail.ProductID = product.ProductID
Where product.ListPrice > 99 AND detail.LineTotal <101;
```

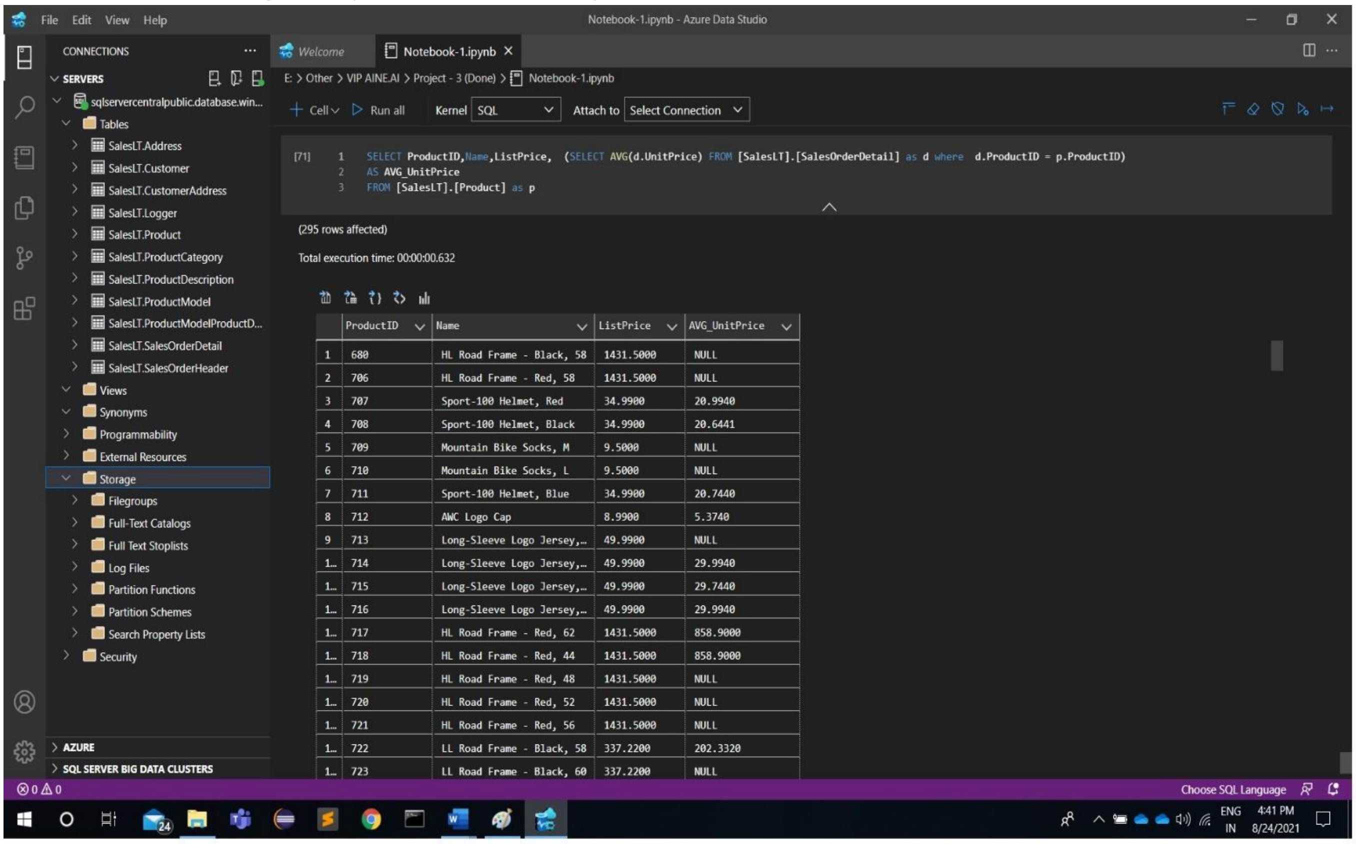
```
(17 rows affected)

Total execution time: 00:00:00.835
```

	ProductID	Name	ListPrice
1	810	HL Mountain Handlebars	120.2700
2	810	HL Mountain Handlebars	120.2700
3	810	HL Mountain Handlebars	120.2700
4	813	HL Road Handlebars	120.2700
5	813	HL Road Handlebars	120.2700
6	876	Hitch Rack - 4-Bike	120.0000
7	907	Rear Brakes	106.5000
8	948	Front Brakes	106.5000
9	948	Front Brakes	106.5000
1…	948	Front Brakes	106.5000
1…	948	Front Brakes	106.5000
1…	984	Mountain-500 Silver, …	564.9900
1…	984	Mountain-500 Silver, …	564.9900
1…	985	Mountain-500 Silver, …	564.9900
1…	986	Mountain-500 Silver, …	564.9900
1…	987	Mountain-500 Silver, …	564.9900
1…	988	Mountain-500 Silver, …	564.9900

As query shown in image I retrieve products with a list price of $100 or more that have been sold for less than $100.

7.c) Retrieve the cost, list price, and average selling price for each product

Retrieve the product ID, name, cost, and list price for each product along with the average unit price for which that product has been sold.



```
SELECT ProductID,Name,ListPrice,  (SELECT AVG(d.UnitPrice) FROM [SalesLT].[SalesOrderDetail] as d where  d.ProductID = p.ProductID)
AS AVG_UnitPrice
FROM [SalesLT].[Product] as p
```

```
(295 rows affected)

Total execution time: 00:00:00.632
```

	ProductID	Name	ListPrice	AVG_UnitPrice
1	680	HL Road Frame - Black, 58	1431.5000	NULL
2	706	HL Road Frame - Red, 58	1431.5000	NULL
3	707	Sport-100 Helmet, Red	34.9900	20.9940
4	708	Sport-100 Helmet, Black	34.9900	20.6441
5	709	Mountain Bike Socks, M	9.5000	NULL
6	710	Mountain Bike Socks, L	9.5000	NULL
7	711	Sport-100 Helmet, Blue	34.9900	20.7440
8	712	AWC Logo Cap	8.9900	5.3740
9	713	Long-Sleeve Logo Jersey,…	49.9900	NULL
1…	714	Long-Sleeve Logo Jersey,…	49.9900	29.9940
1…	715	Long-Sleeve Logo Jersey,…	49.9900	29.7440
1…	716	Long-Sleeve Logo Jersey,…	49.9900	29.9940
1…	717	HL Road Frame - Red, 62	1431.5000	858.9000
1…	718	HL Road Frame - Red, 44	1431.5000	858.9000
1…	719	HL Road Frame - Red, 48	1431.5000	NULL
1…	720	HL Road Frame - Red, 52	1431.5000	NULL
1…	721	HL Road Frame - Red, 56	1431.5000	NULL
1…	722	LL Road Frame - Black, 58	337.2200	202.3320
1…	723	LL Road Frame - Black, 60	337.2200	NULL

From above query shown in image I retrieve the cost, list price, and average selling price for each product. Retrieved the product ID, name, cost and list price for each product along with the average unit price for which that product has been sold.