Managing Relational and Non-Relational Data

GROUP MEMBERS:
André Oliveira nº. 20222156
Diogo Fernandes nº. 20220507
Gonçalo Matos nº. 20221194
Rafael Chamusca nº. 20222174

PROFESSORS:
João Loureiro
António Sérgio Azevedo
André Ferreira

2023

Index

Introduction

Adventure Works employs about 290 people across multiple departments, including sales, production, purchasing, engineering, finance, information services, marketing, shipping and receiving, and R&D. The company is known for its original designs and high-quality items, and it has over 700 locations and 19,000 customers worldwide. This year, the leadership team decided to try something new: an online auction for all products for which a new model was due to be released within the next two weeks.

Adventure Works provided our team with a database that proves to be a useful resource for analyzing a manufacturing company's operations, deploying an online auction system as well as acquiring insights into the company's performance and discovering opportunities for the firm.

In this project, we will be analyzing the Adventure Works database and creating an online auction process in SQL. For that purpose, we will start by building an auction schema and then creating stored procedures for adding a product to the auction, bidding on a product, removing a product from the auction, updating a product's auction status and creating lists of products' bidding history. This report will be divided into three main parts: the auction schema, the stored procedures and a comprehensive analysis of the dataset to develop data driven arguments for a final proposition of the two best cities for Adventure Works to "set up shop".

1º Business problem

Schema and Tables Configurations

An Auction schema was created to store all the information related to the auctions. Four tables were created within this schema:

1) BidThresholds: to store minimum bid increase and minimum and maximum price thresholds,
2) DateThresholds: to store auction campaign initial and final dates,
3) AuctionInfo: to store all information necessary about each particularly auction (product being auctioned, initial and maximum bid prices, expiration date, status, current bid price and winner),
4) Bids: to store historical information about each bid ever made (related auction, customer id, bid amount and date);

Primary and Foreign keys and table columns can be checked in Annex (Figure 1). Note that DateThresholdsID wasn't added as foreign key to the AuctionInfo table. To prevent multiplying information since in the present settings there is only one possible date configuration. However, to ensure consistency, it would be advisable to add it in the future if more date configurations are added, following the same logic as BidThresholds table.

Our setup allows auction dates and bid thresholds to be modified at any time through the respective configuration table, in order to extend auction durations or change bid prices at will. To ensure

futureproofing, we designed this approach to be able to handle multiple and simultaneous campaigns with different bid thresholds, if needed. In the future, one only need to:

1) Add a new table to the auction schema with CampaignID(PK, INT, IDENTETY) and Description/CampaignName (Varchar) as columns.
2) Add a new column to hold CampaignID as a foreign key to DateThresholds, BidThresholds and AuctionInfo.
3) Revise the store procedures to account not only the AuctionID, but also the CampaignID, as the configuration values may vary across different campaigns.

Stored Procedures

uspAddProcuctToAuction

In this procedure we used 5 temporary variables:

1) BidThresholdID: which retrieves the threshold id from the configuration table considering the flag value of the product,
2) MaximumBidPrice: a multiplication between the defined listed price and maximum threshold bearing in mind the BidThresholdID
3) The Start and Stop bid dates from the configuration table and
4) a Status variable that defaults to ‘Active’.

For auction creation validity purposes, we checked:

1) If ProductID is not null, if it’s listed as a valid product, if it’s not discontinued and if there isn't already an auction ongoing,
2) If the expiration date isn’t in the past or if it’s before 13th November 2023,
3) If the auction is created before the end of the campaign,
4) If the initial bid price is between the calculated price limits.

The only required parameter in this procedure is the product id. If the expiration date and the initial bid price are not provided the duration of the auction is set to a week after the beginning date and for the minimum bid price calculated.

When the procedure it’s called it initiates a transaction. If the transaction succeeds, a new row on the AuctionInfo table is created and a message it’s printed informing the user that the auction was successfully created. If there is any error or interruption during running time a rollback is initiated and an error is raised informing the user that it wasn’t possible to create an auction for that product.

2) uspTryBidProduct

In this procedure we used 6 temporary variables:

1) AuctionID: Retrieves the identification of the ongoing auction for the product selected.
2) Date: Current date or bid date
3) CurrentPrice: Retrieves the current auction price of the product in question
4) MaximumBidPrice: Retrieves the maximum bid price of the product.
5) StartBidDate: Retrieves the beginning date of the bidding window (according to configurations table).
6) StopBidDate: Retrieves the ending date of the bidding window (according to configurations table).

For bid creation validity purposes, we checked:

1) If all required parameters weren't null.
2) If the customer is registered.
3) If there's an ongoing auction for the product selected.
4) If the bid amount is within the minimum and maximum bid prices and if it's not equal to the current bidding price.
5) If the bid creation date was within the auction window.

The only required parameters in this procedure is the product and customer id. If the bid amount is not provided first we check if it's equal to the maximum bid amount, in case the user desires to buy the product straight away. If it's not, the minimum bid increase previously configured is added to the auction current price.

When the procedure it's called it initiates a transaction, locking any other user bid attempts. If the transaction succeedes a new row with all the bid information is added to the Bids table and the current price of the auction is updated. If the bid reaches the auction maximum price, the auction ends, locking any other user to further bidding and setting the current user as the auction winner. If the transaction fails a rollback is initiated, in order to restore to the previous state and an error is raised, warning the user that it wasn't possible to place the bid.

3) uspRemoveProductFromAuction

In this procedure we only use one temporary variable, AuctionID, to retrieve the identification number of the ongoing auction for the product. First, we checked if the product id parameter isn't null and if there's an active auction for that product at the moment. We then initiate a transaction to inhibit any user from being able to make a bid on this product during removal. If the transaction is successful, it

sets the auction state to “Canceled” and it doesn't allow any more bids. If it’s not successful a rollback is initiated, and an error is raised warning the user that it wasn’t possible to remove the product from auction.

4) uspListBidsOffersHistory Procedure

We start this procedure by checking if all the mandatory parameters are not null, if the customer is registered on our database, and if the time interval set by the user is valid (end date not earlier then start date). If all conditions are met a query is initiated. If the “Active” parameter is true (default value) it returns all information regarding the bids posted by that customer between the desired time frame for all ongoing auctions. If this parameter is false, it will return all bid history of the customer. If the query fails, an error is raised warning the customer that it wasn’t possible to retrieve the bidding history.

5) uspUpdateProductAuctionStatus

This procedure begins a transaction. First, the status of the auctions with the expiration date later than the current date are set to “Ended”. Then it checks if there was any row affected by the update. If there is, then check to which customer belongs the maximum bet for the given auction and stores that information in the “Winner” column of the AuctionInfo table. If the transaction fails, a rollback is initiated, and an error is raised warning the user that it wasn’t possible to update the auction status. As the winner and “Ended” status is handled by the *trybidproduct* procedure and the “Canceled” by the *removeProductFromAuction* this procedure only needs to be runned once a day to check if any auction has expired.

2º Business problem: The first 2 Brick and Mortar Adventure Work Stores!

In order to offer products directly to clients, Adventure Works (AW) intends to open its first two physical storefronts in the US. However, only in places where their top 30 B2B clients don't have operations.

Every data analysis begins with a set of queries. By ''asking questions'' to our dataset, we were able to extract important facts and insights that could be used to address the current business issue. Thus, the report responds to the inquiries raised in order to support the investigation into the best cities for AW to establish its first two brick and mortar retail locations.

On an initial analysis, Adventure Works has 753 stores as clients (see Figure 2 - How many customers of AW are stores?), 391 of which are in the United States (see Figure 3 - Of 753 stores clients how many are in the US?), the only ones relevant toward in the decision-making process relating to choosing

the optimal cities to open direct to consumer stores. For the purposes of this analysis, we will solely focus on the latter 391 stores.

As an initial step, to exclude the 30 best customer stores, a query was run on the Adventure Works database to extract the list of cities where the 361 stores are located as well to ascertain the number of resellers in each city. Before proposing the two best cities to open AdventureWorks' first stores, we need to proceed with the exclusion of the cities where the best 30 customers are located (see Figure 4 - best 30 stores customers according to the SubTotal col (Sales.SalesOrderHeader table)), thus we filtered out those cities from the list of potential cities for this analysis (see Figure 4 - Top 30 Cities table). We retrieved the information in the SubTotal column to not include Tax amount and Freight on our analysis.

The remaining cities can be ranked based on factors such as number of existing non-store customers, total recurring revenue and existing store customers per city. This will aid in the selection of the two cities with the greatest potential to optimize future revenue for Adventure Works while reducing the risk of cannibalizing and competing with current store clients that purchase Adventure Works' products for resale.

Regarding non-store or individual clients, our analysis shows Adventure Works has 18484 globally and 7843 in the US (see Figure 5 - How many customers of AW are individuals?; see Figure 6 - How many individual customers are in the US?). We will focus on the latter for the purpose of informing the decision of which cities to "open up shop". In terms of the number of clients per city, Burien, Concord, Beaverton and Chula Vista stand out as having more than 200 non-store customers, featuring good demand prospects at first glance (see Figure 7 - How many individual clients per city in the US (Including top 30 store clients' cities)?, Number of customers per city table).

As the main deciding factor for our proposition, we will utilize the total registered revenue per city as the most viable proxy for demand in each city. The cities that reported most revenue usually show a high number of clients, the top ten cities in revenue are respectively Bellflower, Burbank, Berkeley, Colma, Burien, Bremerton, Burlingame, Concord and Beverly Hills, all showing similar levels of revenue and number of clients, namely every city shows more than 200 clients and roughly within a 10 thousand range in terms of revenue (see Figure 8 - Merge between individual clients and total revenue of customers grouped by city, Revenue Per city Table). As a criterion for choosing the two optimal cities, we will limit our proposal to the top 10 cities in terms of revenue, since the drop off in the latter and in clients is noticeable thereafter, indicating greater variance in demand.

Upon data and contextual analysis, the two cities we propose Adventure Works to "open up shop" in are Bellflower and Chula Vista. Bellflower has the highest revenue for a city at 334018 and shows a large potential client base of 243 individual clients. Chula Vista shows one of the highest potential client bases and sits in the fifth city in terms of total revenue at 257925.

Both these cities are located in California, the birthplace of mountain biking, where a group of cyclists began to explore Marin County's rugged terrain on modified single-speed cruisers in the late 1970s. California is also known for its active lifestyle and outdoor culture, making it an ideal location for bicycle enthusiasts. Chula Vista and Bellflower, specifically, are both areas with growing populations and increasing demand for recreational activities, showing. Also, neither of these cities have any store clients, nullifying the risk of cannibalization and competition with AdventureWorks' own clients (see Figure 9 - Which cities are we working with?).

Under the criteria of maximizing potential demand in terms of the number of clients and market size, as well as minimizing cannibalization, our two proposed cities prove to be optimal places to open stores. Operational efficiencies are also a possibility since both cities are within two hours' distance.

Conclusion

In the initial phase of the project we were able to create an auction schema within AW database with SQL. As a second step, we developed stored procedures for adding products for auctions, attempting to bid on products, removing products from auctions lists, viewing the history of bids and offers, and changing the status of product auctions. Lastly, we've suggested Adventure Works to open two new direct-to-consumer stores in the US, specifically in Chula Vista and Bellflower. According to our analysis on the amount of current non-store clients, total recurring revenue, and current store customers per city, we think that Adventure Works should expand its operations to these two cities**.**

Annex

Figure 1 - Auction Schema and Foreign Key Tables

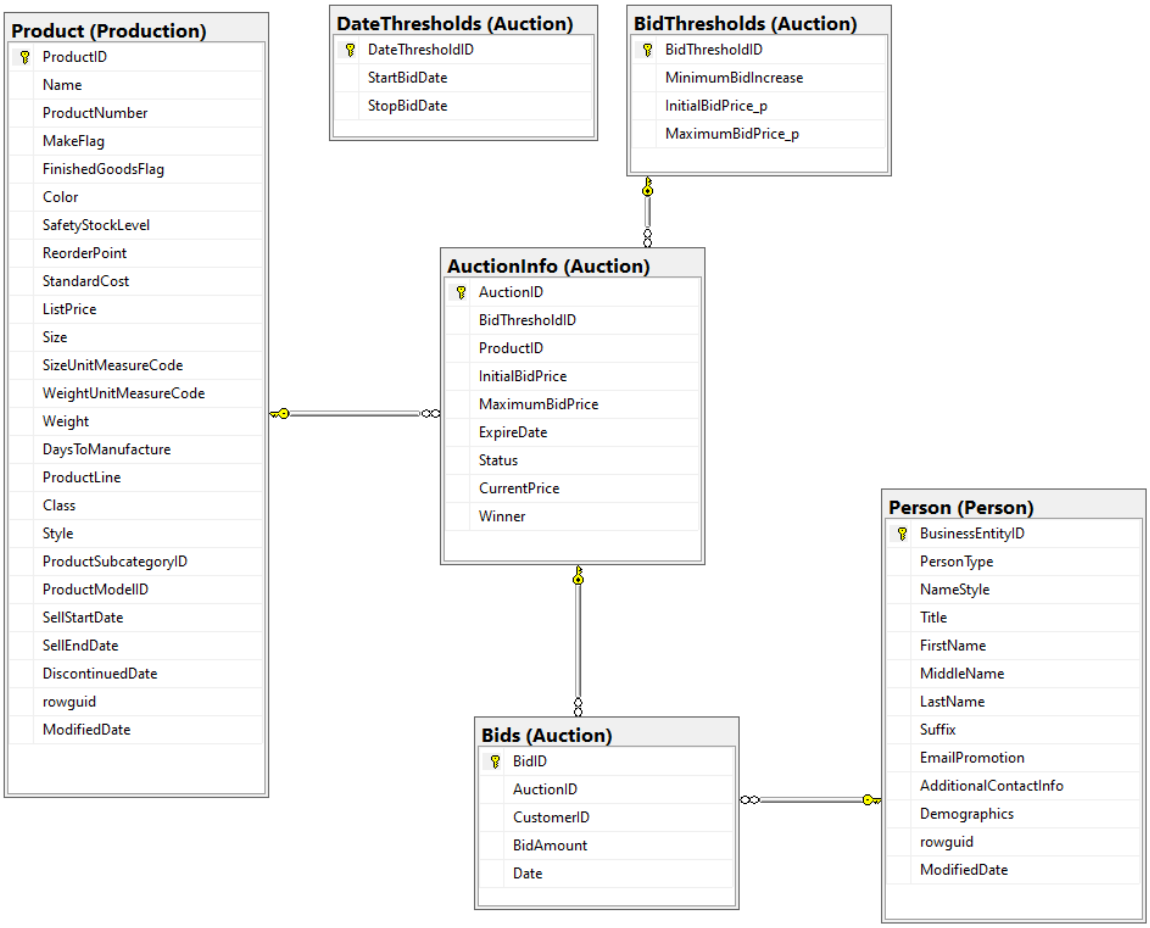

Figure 2 - How many customers of AW are stores?

```
SELECT
  COUNT(*) AS Resellers
FROM
  Person.Person
WHERE
  PersonType = 'SC';
```

Figure 3 – Of the 753 store clients, how many are in the US?

```
SELECT COUNT(*) AS StoreCustomersInUS
FROM Person.Person AS p
JOIN Sales.Customer AS c
ON p.BusinessEntityID = c.PersonID
JOIN Sales.SalesTerritory AS t
ON c.TerritoryID = t.TerritoryID
WHERE p.PersonType = 'SC' AND t.CountryRegionCode = 'US'
```

Figure 4 - best 30 stores customers according to the SubTotal column (Sales.SalesOrderHeader table)

```
SELECT TOP 30
  c.CustomerID,
  p.FirstName + ' ' + p.LastName AS Name,
  a.City,
  SUM(s.SubTotal) AS TotalRevenue
FROM
  Person.Person AS p
JOIN Sales.Customer AS c
  ON p.BusinessEntityID = c.PersonID
JOIN Sales.SalesOrderHeader AS s
  ON c.CustomerID = s.CustomerID
JOIN Sales.SalesTerritory AS t
  ON c.TerritoryID = t.TerritoryID
JOIN Person.Address AS a
  ON s.ShipToAddressID = a.AddressID
```

```
WHERE
  p.PersonType = 'SC' AND t.CountryRegionCode = 'US'
GROUP BY
  c.CustomerID, p.FirstName, p.LastName, a.City
ORDER BY
  TotalRevenue DESC;
```

Top 30 Store Customers' cities
Tooele
Memphis
San Antonio
Nashua
Lacey
Garland
Seattle
Odessa
Casper
Carson
Loveland
Union City

Houston
College Station
Austin
La Mesa
Reno
Greeley
Melville
Seattle
Sand City
Bellingham
Branch
Culver City
Ellensburg
Newark
Gulfport
Bellevue
Orlando
Las Cruces

Figure 5 - How many customers of AW are individuals?

```
SELECT
  COUNT(*) AS Individuals
FROM
  Person.Person
WHERE
  PersonType = 'IN';
```

Figure 6 - How many individual customers are in the US?

```
SELECT COUNT(Person.PersonType) AS Individuals, sp.CountryRegionCode
FROM Person.Person
JOIN Person.BusinessEntity AS be
ON Person.BusinessEntityID = be.BusinessEntityID
JOIN Person.BusinessEntityAddress as bea
ON be.BusinessEntityID = bea.BusinessEntityID
JOIN Person.Address as pa
ON pa.AddressID = bea.AddressID
JOIN Person.StateProvince as sp
ON sp.StateProvinceID = pa.StateProvinceID
WHERE Person.PersonType = 'IN' AND sp.CountryRegionCode = 'US'
GROUP BY sp.CountryRegionCode;
```

Figure 7 - How many individual clients per city in the US(Including top 30 store clients' cities)?

```
SELECT COUNT(Person.PersonType) AS Individuals, pa.City
```

```
FROM Person.Person
JOIN Person.BusinessEntity AS be
ON Person.BusinessEntityID = be.BusinessEntityID
JOIN Person.BusinessEntityAddress as bea
ON be.BusinessEntityID = bea.BusinessEntityID
JOIN Person.Address as pa
ON pa.AddressID = bea.AddressID
JOIN Person.StateProvince as sp
ON sp.StateProvinceID = pa.StateProvinceID
WHERE Person.PersonType = 'IN' AND sp.CountryRegionCode = 'US'
GROUP BY pa.City;
```

Individuals	City Of Commerce
212	Burien
212	Concord
210	Beaverton
210	Bellingham
206	Chula Vista
200	Berkeley
198	Burlingame
194	Bellflower
193	Burbank

188	Beverly Hills
182	Bremerton
168	Colma
132	Coronado
110	Downey
108	Lebanon
107	Lemon Grove
104	Milwaukie
104	El Cajon
104	Lake Oswego

Figure 8 - Merge between individual clients and total revenue of customers grouped by city.

```
SELECT
  a.City, a.PostalCode,
  COUNT(pe.BusinessEntityID) AS IndividualClients,
  sum(Sales.SalesOrderHeader.TotalDue) as Money_
FROM
  Person.Person AS pe
JOIN Person.BusinessEntity AS be ON pe.BusinessEntityID = be.BusinessEntityID
JOIN Person.BusinessEntityAddress AS bea ON be.BusinessEntityID = bea.BusinessEntityID
```

```
JOIN Person.Address AS a ON bea.AddressID = a.AddressID
Join Sales.SalesOrderHeader
ON Sales.SalesOrderHeader.BillToAddressID = a.AddressID
JOIN Sales.SalesTerritory
ON Sales.SalesTerritory.TerritoryID = Sales.SalesOrderHeader.TerritoryID
JOIN Person.StateProvince AS sp ON a.StateProvinceID = sp.StateProvinceID
WHERE
  pe.PersonType = 'IN' AND sp.CountryRegionCode = 'US'
  AND a.City NOT IN(
  SELECT TOP 30 City
  FROM Sales.SalesOrderHeader
  JOIN Person.Address
  ON Address.AddressID = SalesOrderHeader.BillToAddressID
  GROUP BY City
  ORDER BY SUM(SubTotal) DESC
)
GROUP BY
  a.City, sp.StateProvinceCode, a.PostalCode
ORDER BY
Money_ DESC;
```

City Of Commerce	PostalCode	IndividualClients	Total Revenue
Bellflower	90706	243	334018.0869
Burbank	91502	238	305487.1492
Berkeley	94704	242	285243.0051

Colma	94014	205	258528.0009
Chula Vista	91910	252	257925.0512
Burien	98168	259	254547.2501
Bremerton	98312	226	250925.2273
Burlingame	94010	239	249358.325
Concord	94519	253	232542.6845
Beverly Hills	90210	226	223980.1525
Lemon Grove	91945	138	203946.3343
Woodland Hills	91364	129	188157.111
Beaverton	97005	246	178965.1733
Santa Monica	90401	123	177272.155
Torrance	90505	122	164014.9487
Coronado	92118	158	162254.8397
Long Beach	90802	119	162215.857
El Cajon	92020	125	153420.1926
Walla Walla	99362	129	153301.3853
Santa Cruz	95062	118	152296.3796
Spring Valley	91977	113	148473.8405

Lebanon	97355	138	147582.6015
Milwaukie	97222	130	145879.8588
Issaquah	98027	126	142206.7925
Lincoln Acres	91950	125	140072.2459
Downey	90241	130	139227.4228
Oakland	94611	113	139117.5725
Corvallis	97330	120	137858.1912
Lake Oswego	97034	123	136854.3387
San Gabriel	91776	117	132384.1191
Lakewood	90712	106	132314.7533
Marysville	98270	111	131199.8369
Glendale	91203	126	130784.9165
Olympia	98501	114	128499.7971
Sedro Woolley	98284	107	128342.5717
West Covina	91791	103	128103.9682
Newport Beach	92625	110	127922.7919
Novato	94947	109	127782.9606
National City	91950	124	127264.8689

San Carlos	94070	108	126497.0266
Palo Alto	94303	115	125810.9998
Los Angeles	90012	114	124846.9987
Puyallup	98371	120	124301.6478
Grossmont	91941	118	123513.3923
Portland	97205	112	122267.7406
Port Orchard	98366	108	118192.6332
La Jolla	92806	115	116443.2454
Edmonds	98020	117	115814.7262
Lynnwood	98036	121	112590.8517
Tacoma	98403	113	112557.8115
Redwood City	94063	104	111764.9371
Kirkland	98033	109	110686.0335
San Diego	92102	111	110047.3878
W. Linn	97068	103	109652.6716
Salem	97301	114	109030.8366
Imperial Beach	91932	108	107829.7786
Renton	98055	107	107577.5659

Yakima	98901	116	104378.3802
Mill Valley	94941	103	101867.3792
Oregon City	97045	104	101096.9698
Everett	98201	115	101075.6509
Woodburn	97071	105	99508.5433
Daly City	94015	104	95306.0619
Redmond	98052	113	87100.9301
Fremont	94536	92	86364.387
Spokane	99202	99	76361.8328
San Francisco	94109	57	75868.3346
Ballard	98107	82	50066.4425
Clearwater	33755	4	8532.7324
Bell Gardens	90201	2	6541.8653
Cheyenne	82001	2	6520.1078
Canoga Park	91303	2	5307.6687
Bountiful	84010	2	4812.9491
Barstow	92311	1	3953.9884
Chicago	60610	5	3116.9947

Citrus Heights	95610	1	2709.5042
Bluffton	29910	1	2690.5866
City Of Commerce	90040	1	2652.3757
Clackamas	97015	1	2649.8453
Clackamas	97015-6403	1	2598.9158
Cheektowaga	14227	1	2582.6834
Chandler	85225	2	2324.9417
Bellevue	98004	1	2264.2536
Braintree	2184	1	2264.2536
Clay	13041	1	1918.2579
Bothell	98011	3	1842.5323
Columbus	31901	2	1795.0173
Camarillo	93010	2	1421.5052
Cerritos	90703	1	1265.7554
Cincinnati	45202	4	311.5328
Campbellsville	42718	2	239.7408
Cedar Park	78613	1	103.8369
Billings	59101	1	101.7484

Branch	55056	2	100.8645
Biloxi	39530	1	91.262
Branson	65616	1	90.0133
Carrollton	75006	1	79.5048
Columbus	43215	1	77.3169
Cedar City	84720	1	70.6869
Central Valley	10917	1	56.2887
Chantilly	20151	1	44.1779
Bradenton	34205	1	43.0729
Birmingham	35203	2	41.2055
Clarkston	30021	1	32.5754
Baytown	77520	1	28.1554
Burbank	44214	1	8.0444
Charlotte	28202	1	8.0444
Carol Stream	60188	1	8.0444
Chehalis	98532	1	5.514
Byron	31008	1	5.514

Figure 9 - Which cities are we working with?

```
WITH TopCities AS (
  SELECT TOP 30
        a.City,
        SUM(o.SubTotal) AS TotalRevenue
  FROM
    Person.Address AS a
  JOIN Sales.SalesOrderHeader AS o ON a.AddressID = o.ShipToAddressID
  GROUP BY a.City
  ORDER BY TotalRevenue DESC
)
SELECT
  c.CustomerID,
  p.FirstName + ' ' + p.LastName AS Name,
  a.City,
  SUM(s.SubTotal) AS TotalRevenue
FROM
  Person.Person AS p
JOIN Sales.Customer AS c ON p.BusinessEntityID = c.PersonID
JOIN Sales.SalesOrderHeader AS s ON c.CustomerID = s.CustomerID
JOIN Sales.SalesTerritory AS t ON c.TerritoryID = t.TerritoryID
JOIN Person.Address AS a ON s.ShipToAddressID = a.AddressID
WHERE
  p.PersonType = 'SC' AND t.CountryRegionCode = 'US' and a.City NOT IN (
        SELECT City FROM TopCities
  )
```

```
GROUP BY
  c.CustomerID, p.FirstName, p.LastName, a.City
ORDER BY
  TotalRevenue DESC;
```

CustomerID	Name	City	TotalRevenue
29772	Janet Gates	Austin	586524.9474
29562	Steven Brown	La Mesa	585516.4328
30107	Margaret Vanderkamp	Reno	577089.5723
29924	Mitch Kennedy	Greeley	572035.1121
29966	Richard Lum	Melville	537528.0958
29641	Raul Casts	Sand City	534956.2785
29486	Kim Abercrombie	Branch	519411.3564
29522	Thomas Armstrong	Culver City	492362.7634
30112	Patricia Vasquez	Ellensburg	477957.2715
29919	Eugene Kogan	Newark	456739.4746
29507	Phyllis Allen	Gulfport	446013.5351
29559	Robert Bernacchi	Bellevue	443316.1126
29992	Sandra Maynard	Las Cruces	429050.8729
29901	John Kelly	Gilroy	427890.8367

29834	Cheryl Herring	El Segundo	424512.6949
29825	James Hendergart	Austell	422706.6023
30046	Elizabeth Sullivan	Mesquite	418436.5299
29707	Helen Dennis	Chantilly	416702.7289
29690	Conor Cunningham	Park City	415310.5504
29703	Stefan Delmarco	Ontario	413054.8167
29861	Phyllis Huntsman	Minneapolis	408483.4465
29844	Nancy Hirota	Plaistow	402108.4718
29888	Brannon Jones	Winston-Salem	395413.1309
29724	Bernard Duerr	Milford	394260.688
29651	Lee Chapla	Racine	392784.6026
29885	David Johnson	Longview	387118.3508
29955	David Liu	Zeeland	373658.4657
29938	Frank Campbell	Cerritos	372146.7266
29594	Dave Browning	Indianapolis	357708.7747
29987	Frank Martinez	Chandler	355141.9807
29998	Jane McCarty	Santa Monica	354805.2477
29929	Jeffrey Kurtz	Fullerton	349342.1078

30094	Sairaj Uddin	Sacramento	338939.2001
30109	Nieves Vargas	Howell	336401.6607
29560	Matthias Berndt	Escondido	316323.8118
29853	Juanita Holman	Lake Elsinore	311446.431
29783	Mary Gimmi	Longmont	309077.4104
29509	Michael Allen	Tilton	299530.4863
29792	Abigail Gonzalez	Humble	294389.8076
29896	Judith Krane	Columbus	293610.7156
29658	Mike Choi	Daleville	292229.5405
30065	Karen Theisen	Stamford	288827.1098
29889	Jean Jordan	Monrovia	286972.0865
29996	Katie McAskill-White	New Hartford	286645.1385
29543	Karel Bates	Columbus	284876.0711
29579	Cornelius Brandon	Westminster	284305.6301
29732	Carol Elliott	Corpus Christi	278412.801
30111	Ranjit Varkey Chudukatil	Moline	276526.4031
29935	Elsa Leavitt	Salt Lake City	268660.7965
30076	Diane Tibbott	Parker	268166.0546

29525	Teresa Atkinson	City Of Commerce	261522.3269
29836	Sidney Higa	Kittery	258976.6809
29891	Diane Krane	Novi	248368.8168
29710	Brenda Diaz	Fernley	243014.5498
29916	Reed Koch	Lake George	239853.119
29622	Henry Campen	Tacoma	231593.6394
29963	Spencer Low	La Vergne	223285.3949
29947	Judy Lew	Kittery	220888.8487
29550	Stanley Alan	Milwaukie	216653.6231
30070	Kendra Thompson	Idaho Falls	213751.9967
29549	Shane Belli	Merritt Island	212208.8366
29695	Megan Davis	Everett	208802.8732
29643	Matthew Cavallari	Spokane	204203.3695
29513	Selena Alvarado	Nashville	194423.137
29880	Samuel Johnson	Gaffney	193708.595
29921	Joy Koski	Beaverton	193258.6708
30116	Wanda Vernon	Ontario	187114.2011
29738	Gail Erickson	Denby	185859.5956

29521	Tom Johnston	Las Vegas	184846.2361
29563	Chris Bidelman	Bradenton	184837.3299
29746	Fadi Fakhouri	Cincinnati	182730.2472
29660	Anthony Chor	Sherman Oaks	182025.0622
29524	Chris Ashton	Saint Ann	181062.7128
29954	Paulo Lisboa	Dallas	179831.3508
29727	Bart Duncan	Hillsboro	178066.6501
29774	Darren Gehring	Laredo	175502.1574
30105	Gregory Vanderbout	Springfield	173228.2944
29915	Andrew Kobylinski	Kirkland	172227.7313
29945	Gloria Lesko	Sarasota	165815.3344
30051	Brad Sutton	Kelso	163851.8865
29925	Margaret Krupka	North Bend	162313.2916
30071	Daniel Thompson	Issaquah	161109.3682
29504	J. Phillip Alexander	Pigeon Forge	158899.1488
29828	Jay Henningsen	Valley Stream	158171.4804
29958	Run Liu	Millington	154348.9886
29582	Ted Bremer	Redmond	153310.5787

29811	Mark Hanson	Newport News	151331.0556
29968	Helen Lutes	Greensboro	149121.222
29787	Scott Gode	Kent	148659.2722
29942	Roger Lengel	Renton	148588.0527
29737	John Emory	Medford	148098.2923
29720	Gerald Drury	Ogden	147887.6819
30045	Ruth Suffin	Salem	147784.2145
29645	Andrew Cencini	Klamath Falls	147421.7513
30060	Jeff Teper	New Castle	146256.5741
29967	Judy Lundahl	West Chicago	146233.5598
29528	Stephen Ayers	Southgate	144109.8667
30056	Clarence Tatman	San Ysidro	140874.514
29492	Jay Adams	Kansas City	140130.8409
29842	Mike Hines	Scottsdale	138868.2962