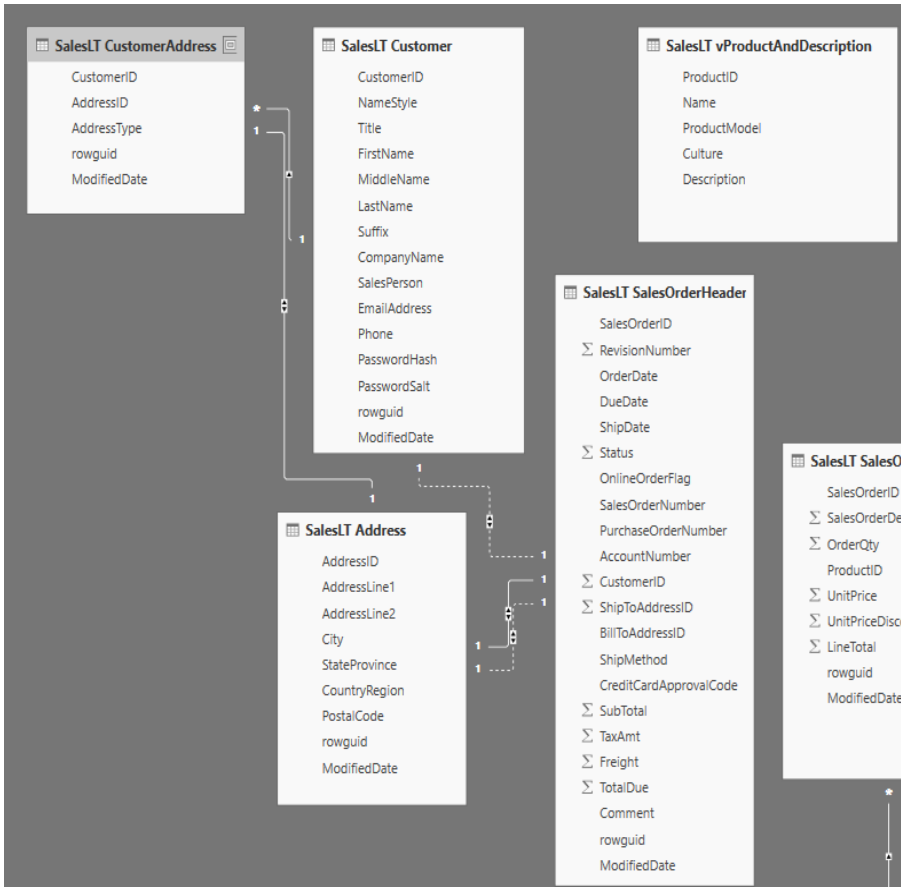
Data Model



SQL Queries

These are some critical Sql queries which impact business rules and processes:

- BI Queries & results (SQL queries is fine, can use BI tools but NOT graded on BI)
    - Ranked order of Vendors by purchase amount $

```
select RANK() over
(order by
SUM(FP.TotalDue)
desc) as Ranking,
DV.VendorName as
'Vendor Name',
CONCAT('$',SUM(FP.Tot
alDue)) as 'Purchase
Amount'
from FactPurchases as
FP
Inner join DimVendors
as DV
```

```
on FP.VendorID = DV.BusinessEntityID
group by DV.VendorName;
```

- Ranked order of products by amount $
  - By category

```
select RANK() over (order by SUM(FP.LineTotal)
desc) as Ranking, DP.ProductCategoryName as
'Product Category', CONCAT('$',SUM(FP.LineTotal))
as 'Purchase Amount'
from FactPurchases as FP
inner join DimProducts_Purchased as DP
on FP.ProductPurchasedSK = DP.ProductPurchasedSK
group by DP.ProductCategoryName;
```

  - By subcategory

```
select RANK() over (order by SUM(FP.LineTotal)
desc) as Ranking, DP.ProductSubcategoryName as
'Product Subcategory',
CONCAT('$',SUM(FP.LineTotal)) as 'Purchase Amount'
from FactPurchases as FP
inner join DimProducts_Purchased as DP
on FP.ProductPurchasedSK = DP.ProductPurchasedSK
group by DP.ProductSubcategoryName;
```

  - By product model (top 20)

```
select Top 20 RANK() over (order by
SUM(FP.LineTotal) desc) as Ranking, DP.ModelName as
'Product Model', CONCAT('$',SUM(FP.LineTotal)) as
'Purchase Amount'
from FactPurchases as FP
inner join DimProducts_Purchased as DP
on FP.ProductPurchasedSK = DP.ProductPurchasedSK
group by DP.ModelName;
```

  - By product (top 20)

```
select Top 20 RANK() over (order by
SUM(FP.LineTotal) desc) as Ranking, DP.ProductName
as 'Product Name', CONCAT('$',SUM(FP.LineTotal)) as
'Purchase Amount'
from FactPurchases as FP
inner join DimProducts_Purchased as DP
on FP.ProductPurchasedSK = DP.ProductPurchasedSK
group by DP.ProductName;
```

- Ranked list of employees purchasing products amount $

```
select RANK() over (order by SUM(FP.TotalDue) desc)
as Ranking, (DE.FirstName +' '+ DE.MiddleName+' '+
DE.LastName)as
'Employee Full
Name',
CONCAT('$',SUM(FP.
TotalDue)) as
'Purchase Amount'
from FactPurchases
as FP
Inner join
DimEmployee as DE
on FP. EmployeeID
=
DE.BusinessEntityI
D
group by
DE.FirstName,
DE.MiddleName,
DE.LastName;
```

- List of employees who purchased products with phone, email & address

```
select
(DE.FirstName +'
'+ DE.MiddleName+'
'+ DE.LastName)as
'Employee Full
Name',
DE.Phone,
DE.EmailAddress,
DE.AddressLine1,
DE.AddressLine2,
DG.City,
DG.StateProvinceNa
me as 'State',
DG.PostalCode
from FactPurchases
as FP
Inner join
DimEmployee as DE
on FP. EmployeeID
=
DE.BusinessEntityI
D
Inner join
DimGeography as DG
on DE.GeoSK =
DG.GeoSK;
```

- List of purchasing contacts with vendor name, phone, email & address

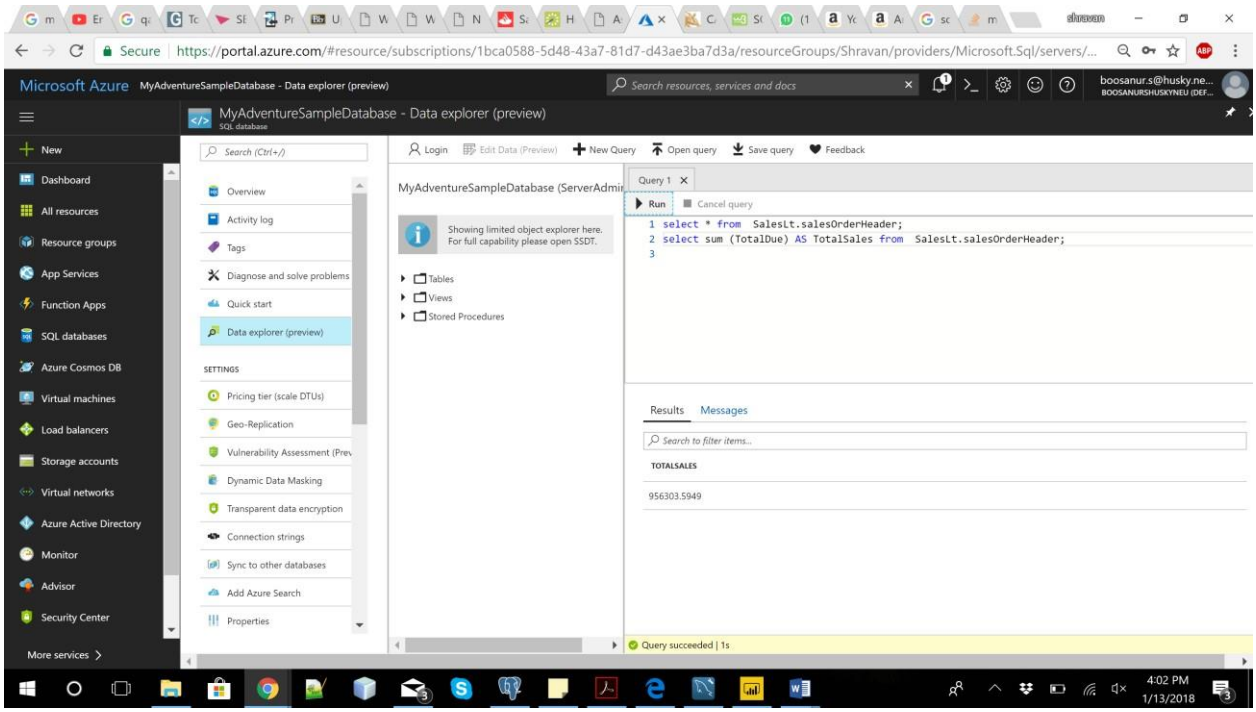
```
Inner join
DimGeography as DG
on DV.GeoSK =
DG.GeoSK;
```

```
select (DVC.FirstName +' '+ DVC.MiddleName+' '+
DVC.LastName)as 'Vendor Full Name', DVC.PhoneNumber,
DVC.EmailAddress, DV.AddressLine1, DV.AddressLine2,
DG.City, DG.StateProvinceName as 'State', DG.PostalCode
from FactPurchases as FP
Inner join DimVendorContacts as DVC
on FP.VendorID = DVC.Vendor_BusinessEntityID
Inner join DimVendors as DV
on DVC.VendorSK = DV.VendorSK
```

Microsoft Azure

```
select sum (TotalDue) AS TotalSales from SalesLt.salesOrderHeader;
```

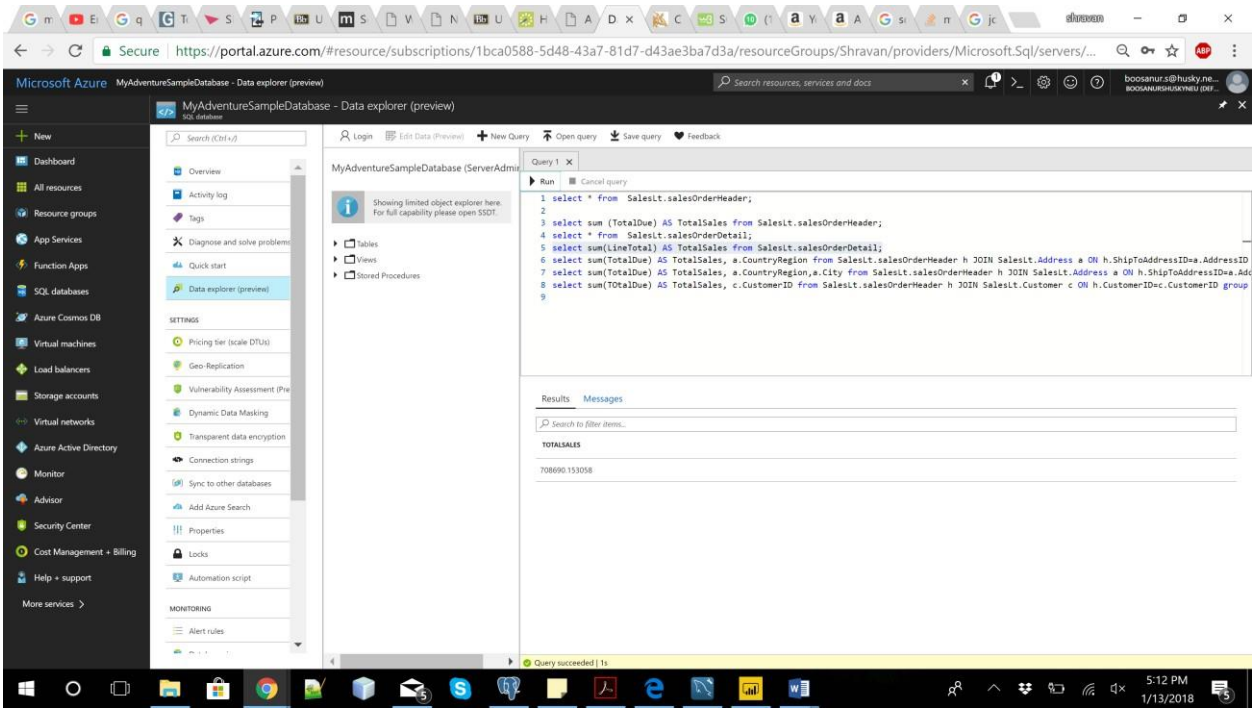
- Input tables: SalesLT.SalesOrderHeader
- Input columns: TotalDue
- Output columns: Sum(TotalDue) AS 'Total Sales'
- Result rowcount: 1



a2. Using SalesOrderDetail

```
select sum(LineTotal) AS TotalSales from SalesLt.salesOrderDetail;
```

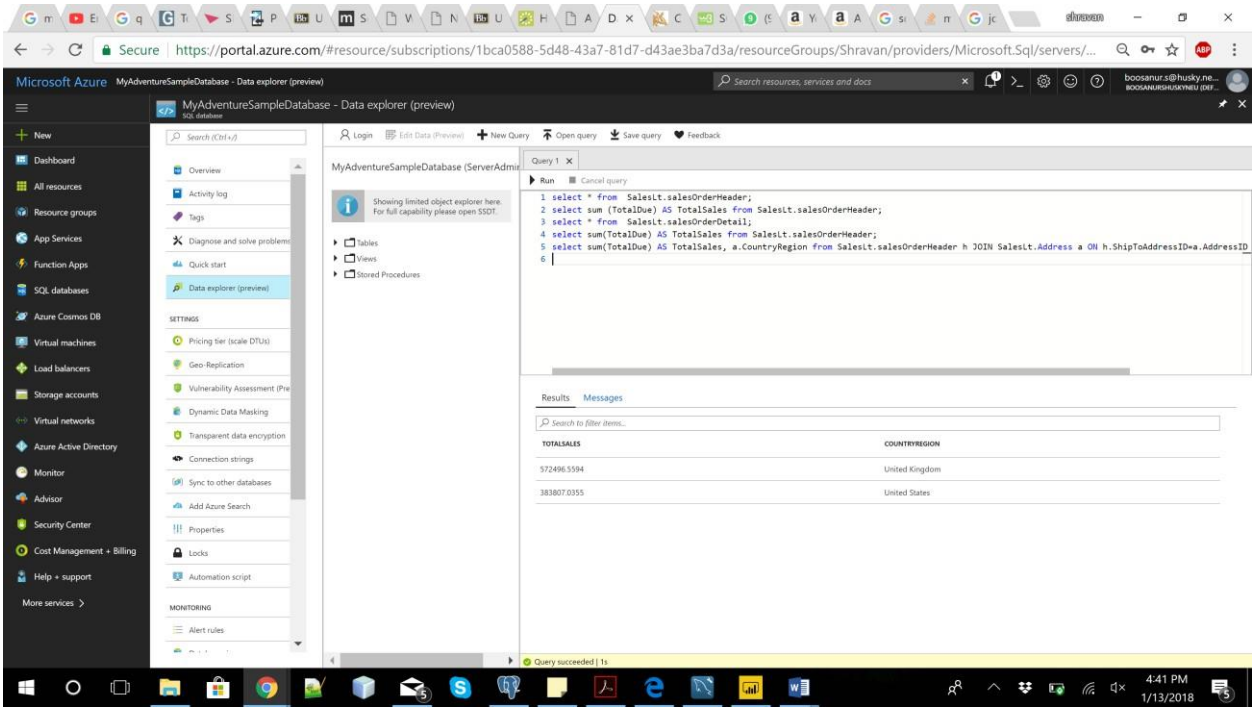
- Input tables: SalesLT.SalesOrderDetail
- Input columns: LineTotal
- Output columns: Sum(LineTotal) AS ‘Total Sales’
- Result rowcount: 1



b) Total sales by country – ranked/sorted (highest to lowest)

```
select sum(TotalDue) AS TotalSales, a.CountryRegion from
SalesLt.salesOrderHeader h JOIN SalesLt.Address a ON
h.ShipToAddressID=a.AddressID group by CountryRegion order by TotalSales
DESC;
```

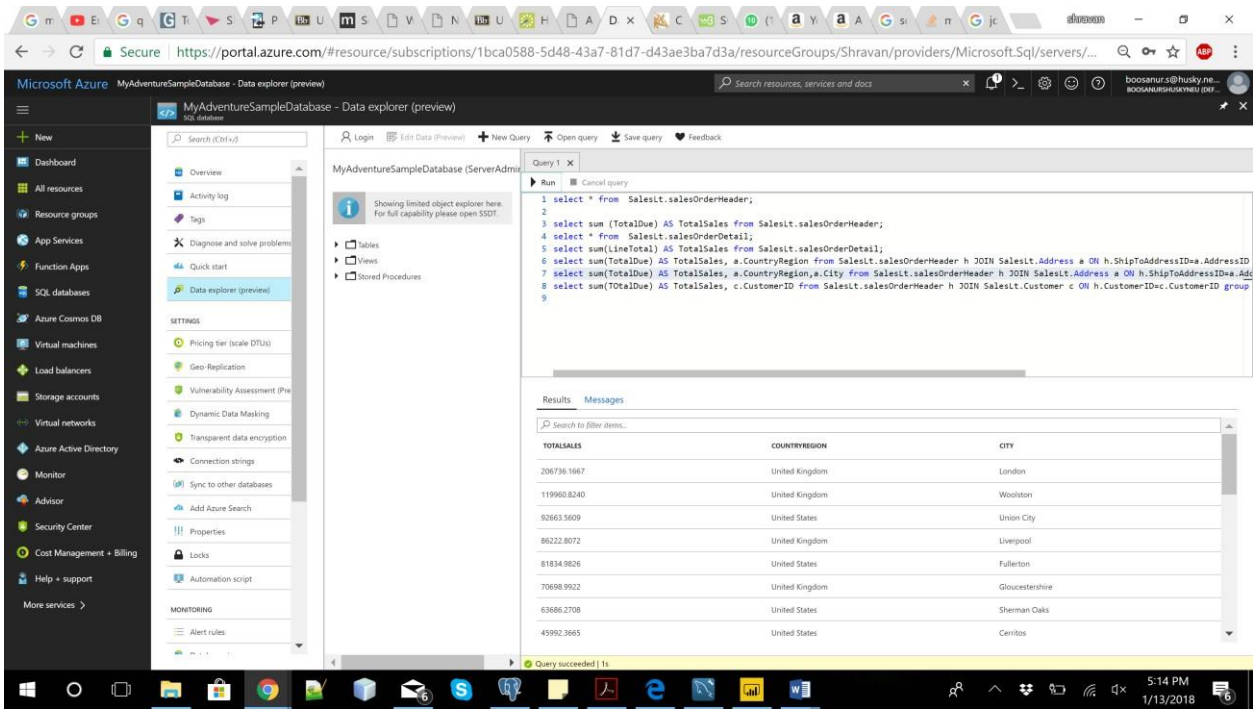
- Input tables: SalesLT.SalesOrderHeader, SalesLT.Address
- Input columns: From table SalesOrderHeader - Total due,
- From table SalesOrderDetail - CountryRegion
- Output columns: CountryRegion, Sum(TotalDue) AS 'Total Sales'
- Result rowcount: 2



c) Total sales by city & country – ranked/sorted (highest to lowest)

```
select sum(TotalDue) AS TotalSales, a.CountryRegion,a.City from
SalesLt.salesOrderHeader h JOIN SalesLt.Address a ON
h.ShipToAddressID=a.AddressID group by CountryRegion,City order by
TotalSales DESC;
```

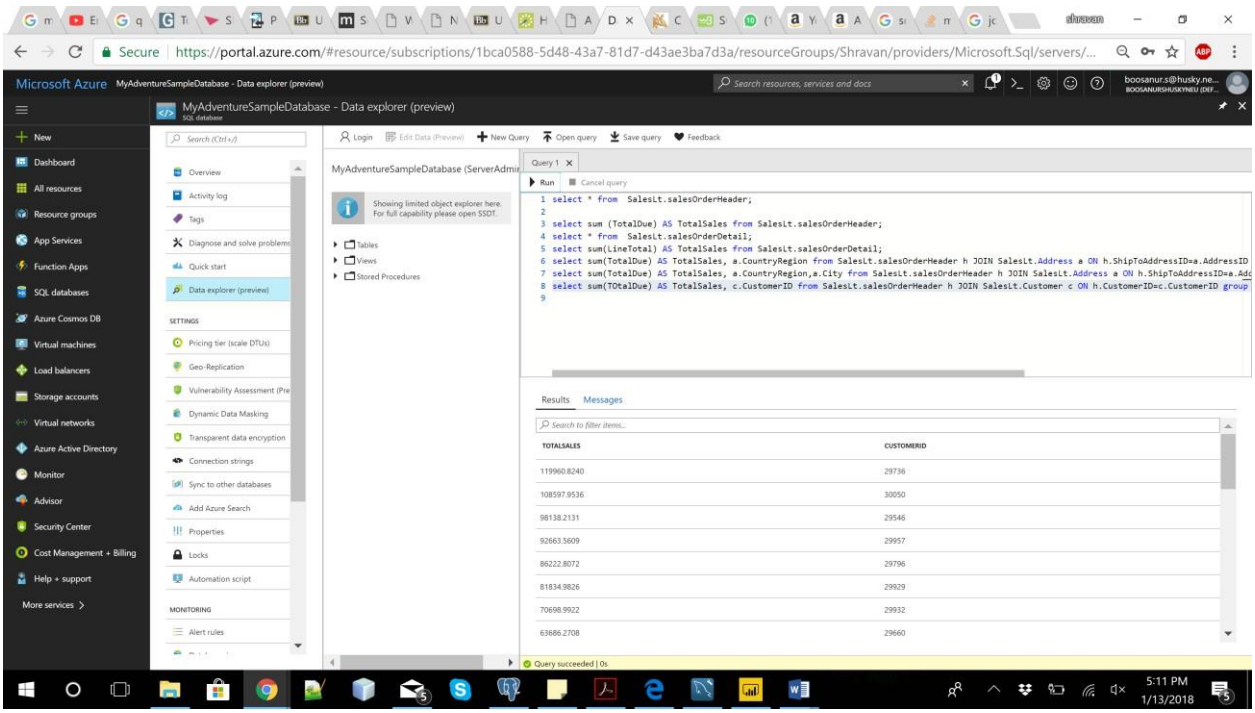
- Input tables: SalesLT.SalesOrderHeader, SalesLT.Address
- Input columns: From table SalesOrderHeader - Total due,
- From table Address – CountryRegion, City
- Output columns: Sum(TotalDue) AS 'Total Sales',CountryRegion, City,
- Result rowcount: 29



d) Total sales by customer (person) – ranked/sorted (highest to lowest)

```
select sum(TOtalDue) AS TotalSales, c.CustomerID from
SalesLt.salesOrderHeader h JOIN SalesLt.Customer c ON
h.CustomerID=c.CustomerID group by c.CustomerID order by TotalSales DESC;
```

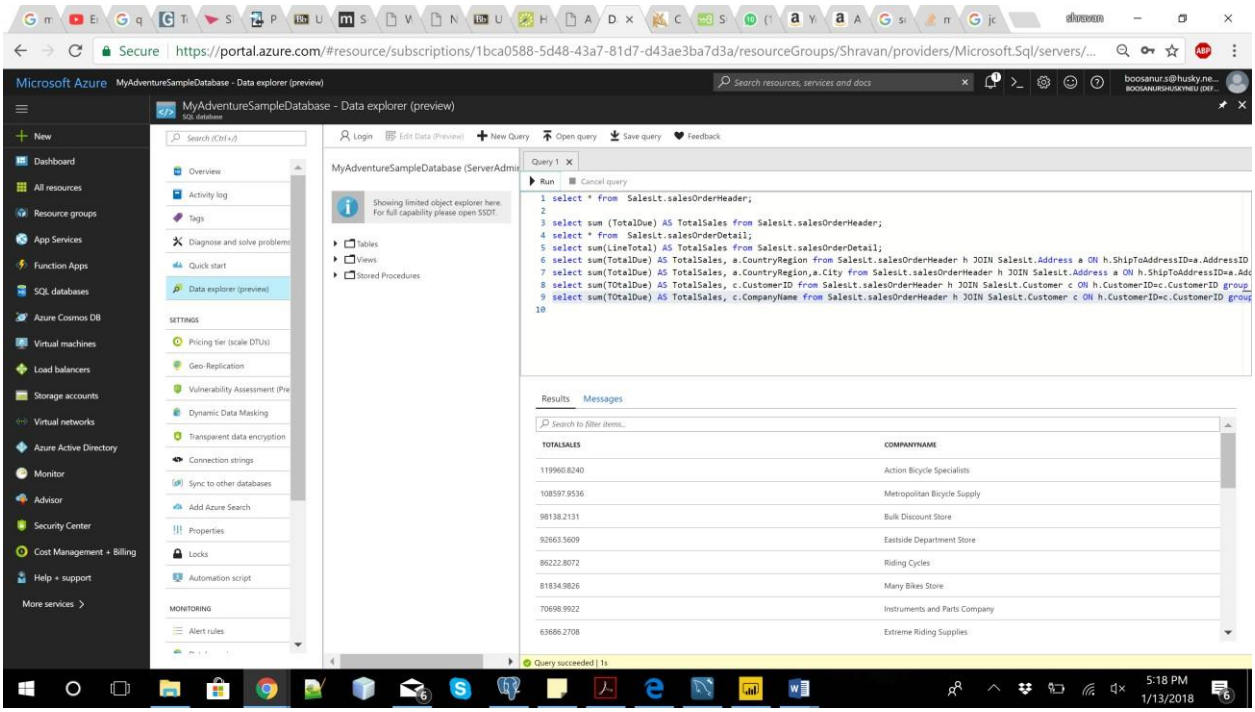
- Input tables: SalesLT.SalesOrderHeader, SalesLT.Customer
- Input columns: From table SalesOrderHeader - Total due,
- From table Customer – CustomerID,
- Output columns: 'Total Sales', CustomerID,
- Result rowcount: 32



e) Total sales by customer (company) – ranked/sorted (highest to lowest)

```
select sum(TOtalDue) AS TotalSales, c.CompanyName from
SalesLt.salesOrderHeader h JOIN SalesLt.Customer c ON
h.CustomerID=c.CustomerID group by c.CompanyName order by TotalSales DESC;
```

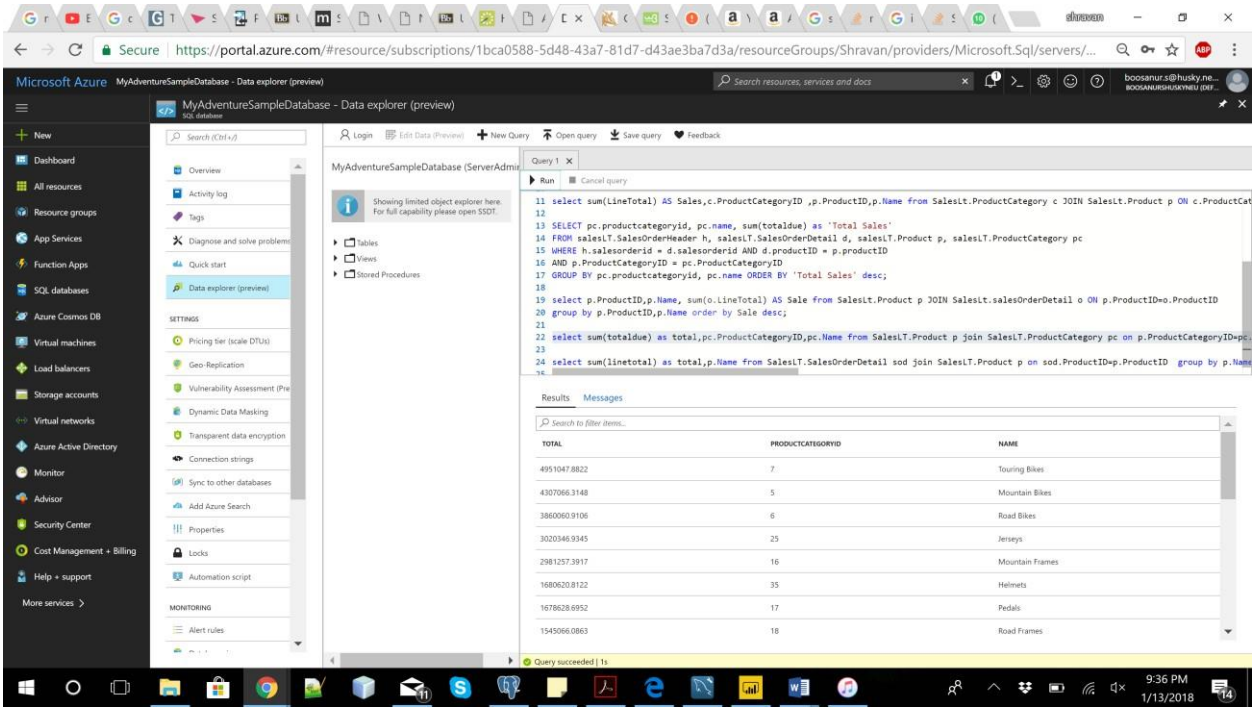
- Input tables: SalesLT.SalesOrderHeader, SalesLT.Customer
- Input columns: From table SalesOrderHeader - Total due,
- From table Customer – CompanyName
- Output columns: CompanyName, 'Total Sales'
- Result rowcount: 32



f) Sales by product category – ranked/sorted (highest to lowest)

```
select sum(totaldue) as total,pc.ProductCategoryID,pc.Name from
SalesLT.Product p join SalesLT.ProductCategory pc on
p.ProductCategoryID=pc.ProductCategoryID join SalesLT.SalesOrderDetail sod
on sod.ProductID=p.ProductID join SalesLT.SalesOrderHeader soh on
soh.SalesOrderID=sod.SalesOrderID group by pc.ProductCategoryID,pc.Name
order by total desc;
```

- Input tables: SalesOrderDetail, salesLT.Product, salesLT.ProductCategory.
- Input columns: From table SalesOrderDetail - LineTotal,
- From table ProductCategory – ProductCategoryID, Name
- Output columns: 'Total', ProductCategoryID, Name
- Result rowcount: 26

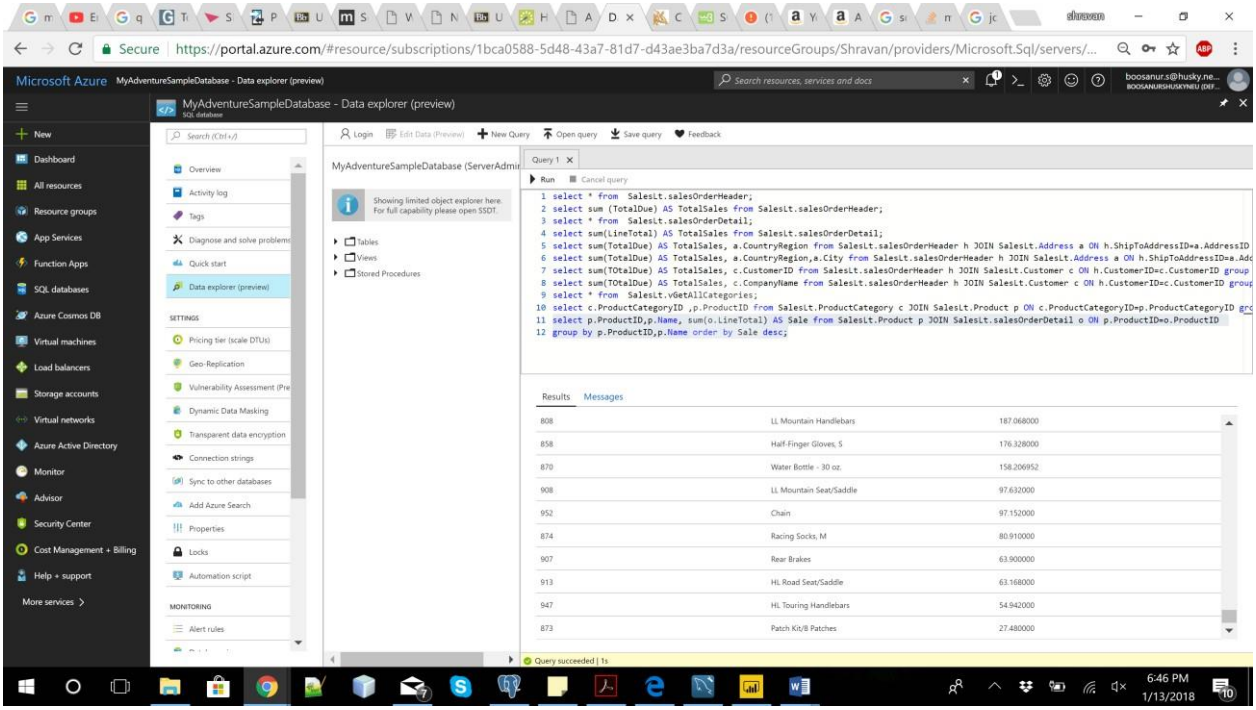


g) Sales by product name – ranked/sorted (highest to lowest)

```
select p.ProductID,p.Name, sum(o.LineTotal) AS Sale from SalesLt.Product p
JOIN SalesLt.salesOrderDetail o ON p.ProductID=o.ProductID

group by p.ProductID,p.Name order by Sale desc;
```

- Input tables: SalesOrderDetail, salesLT.Product
- Input columns: From table SalesOrderDetail - LineTotal,
- From table Product– ProductID, Name
- Output columns: ProductID, Name, ‘Total’
- Result rowcount: 142

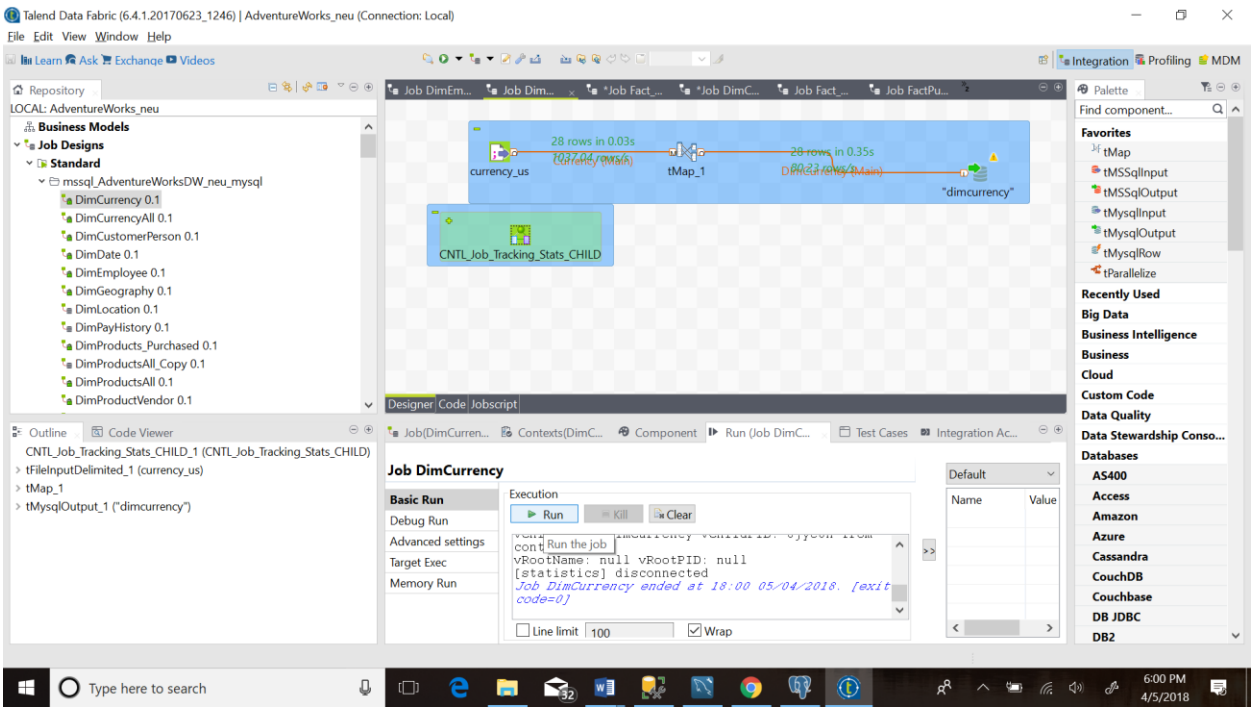


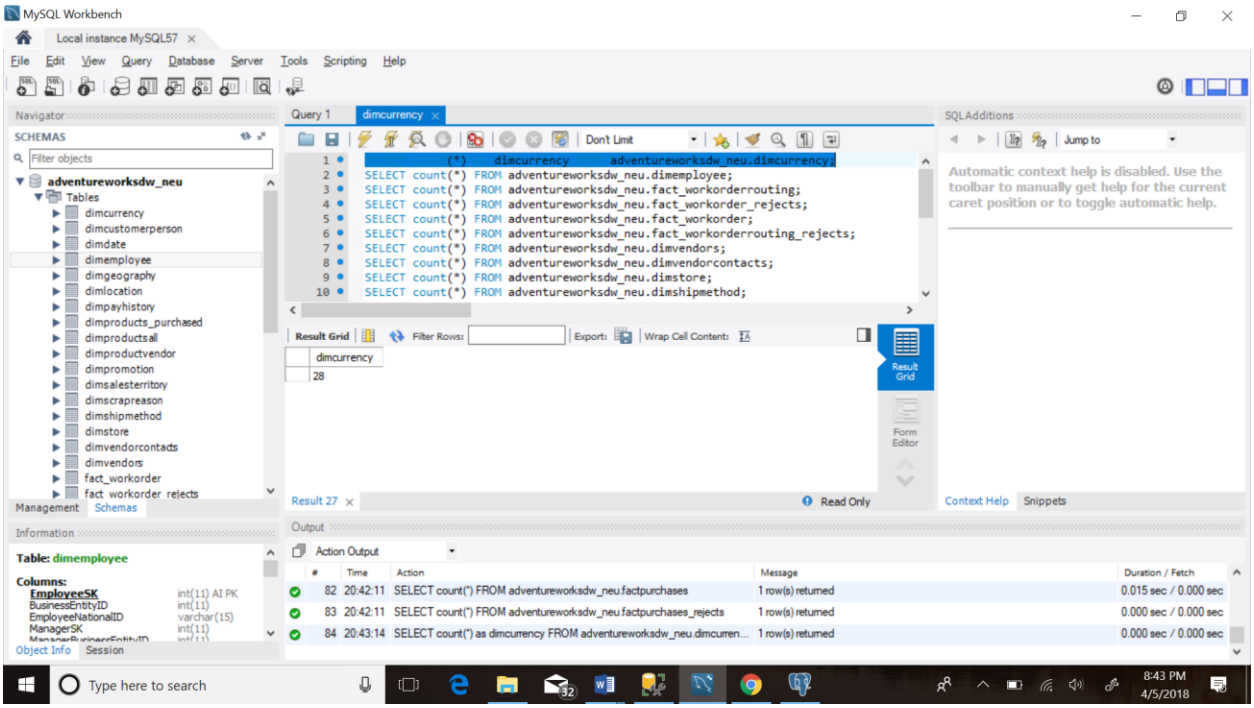
****
****
*

Data Integration

These are talend jobs for Extract, Transform and Load in Data Warehouse:

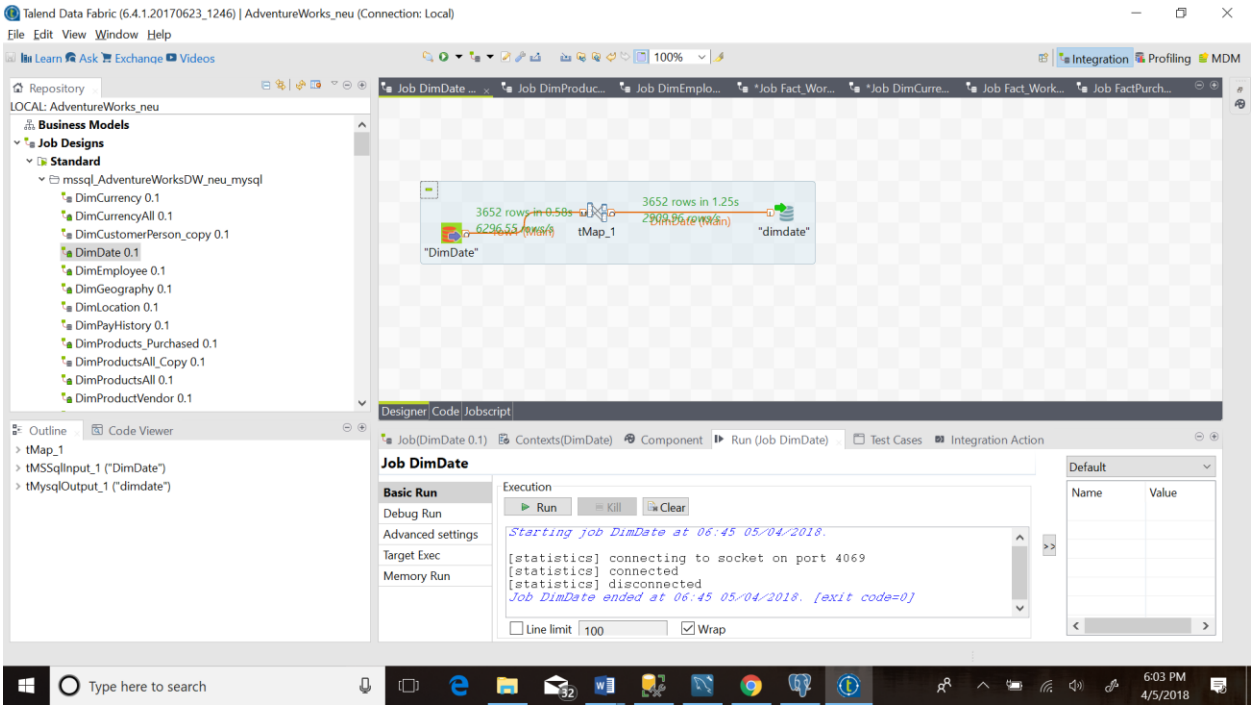
DimCurrency

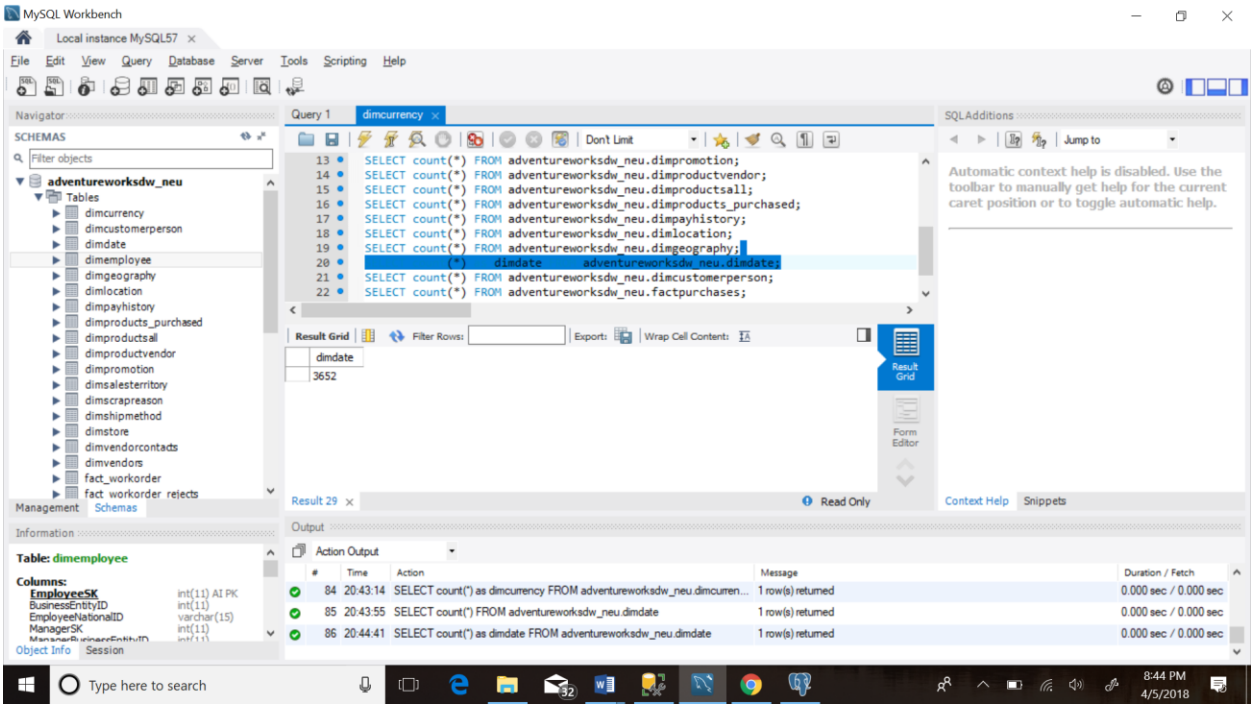




Row count: 28

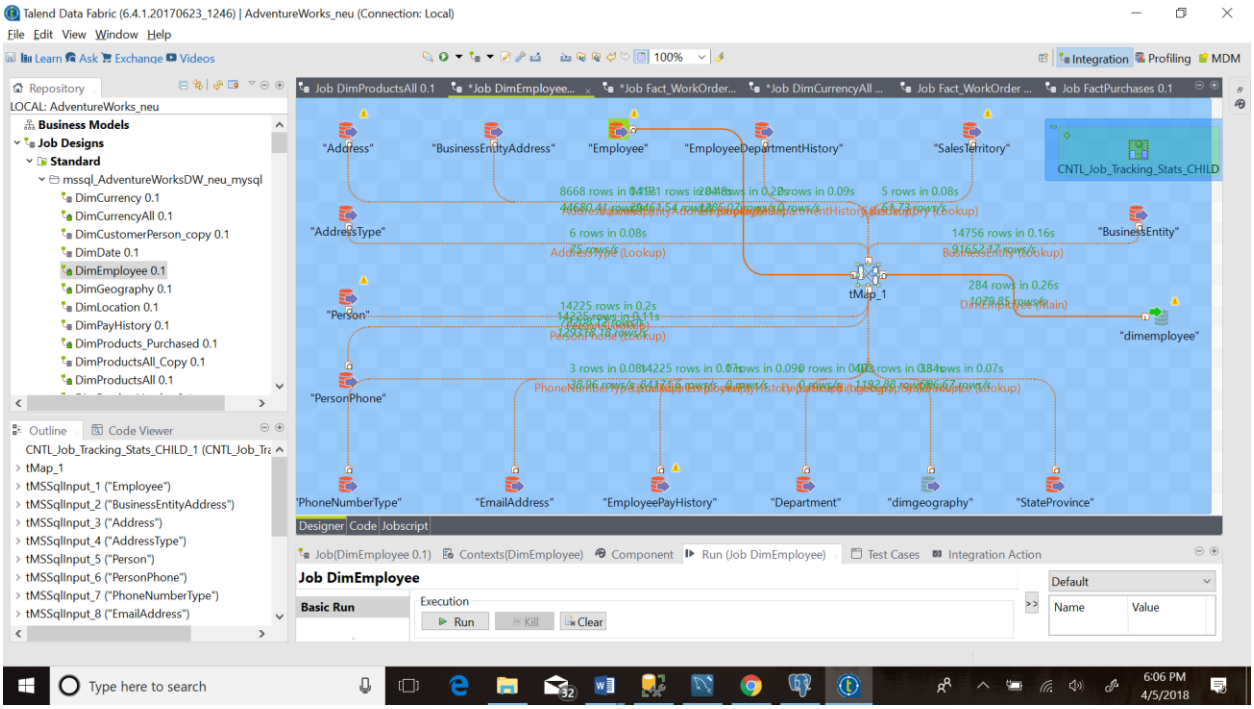
DimDate

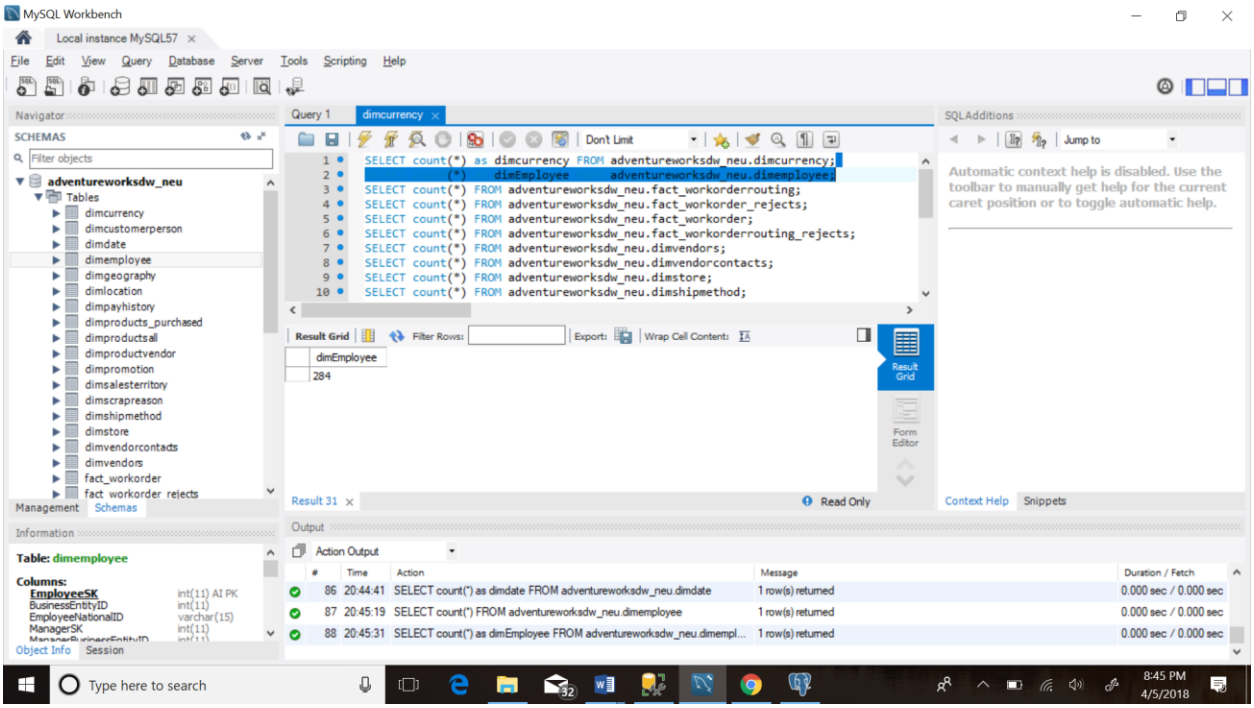




Row Count 3652

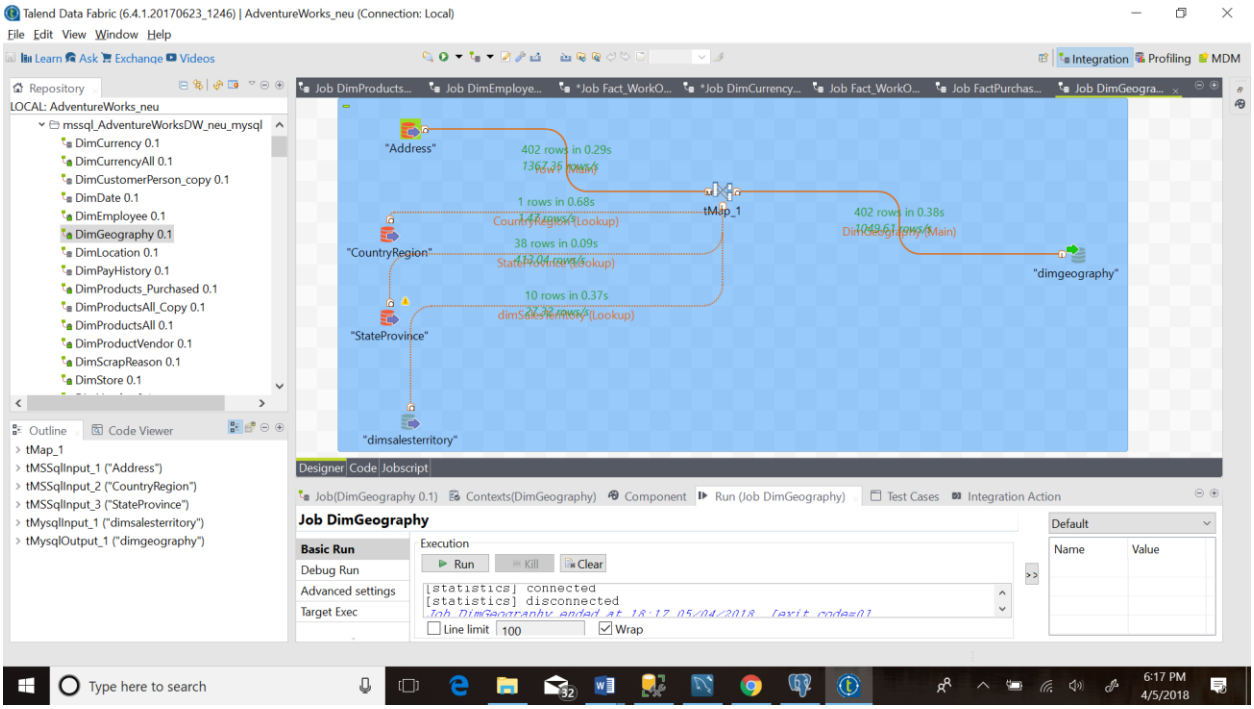
DimEmployee

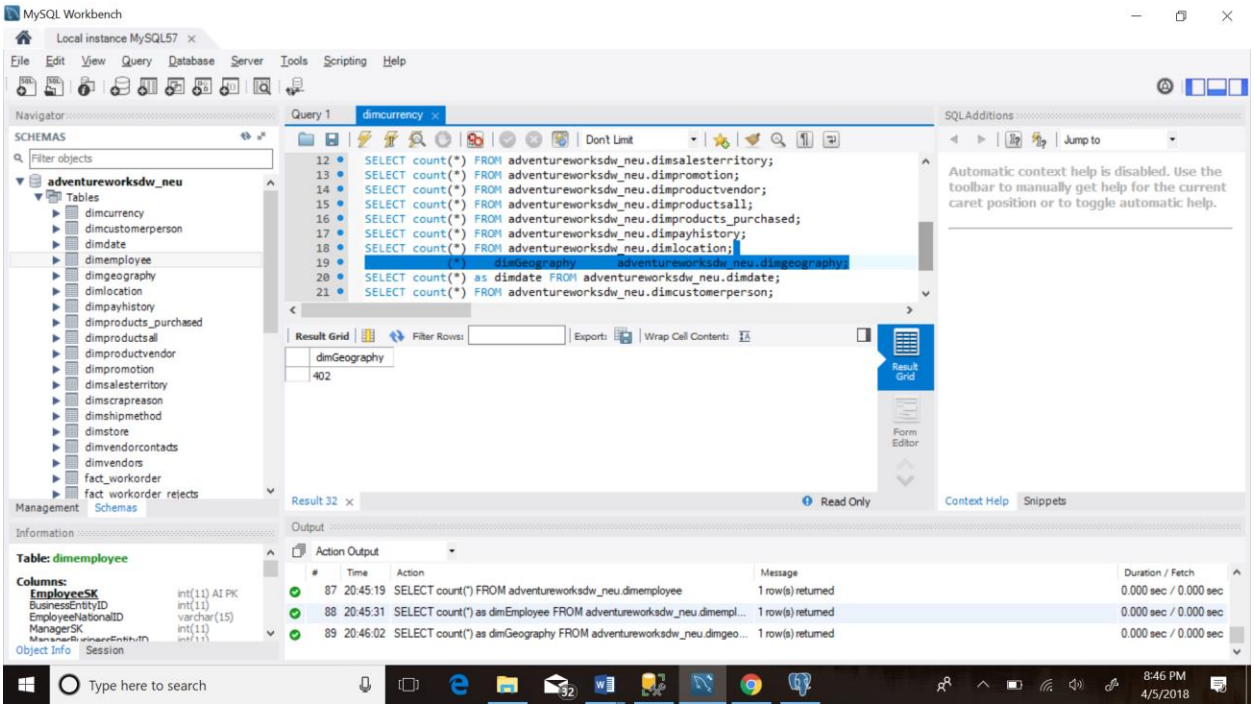




Row Count 284

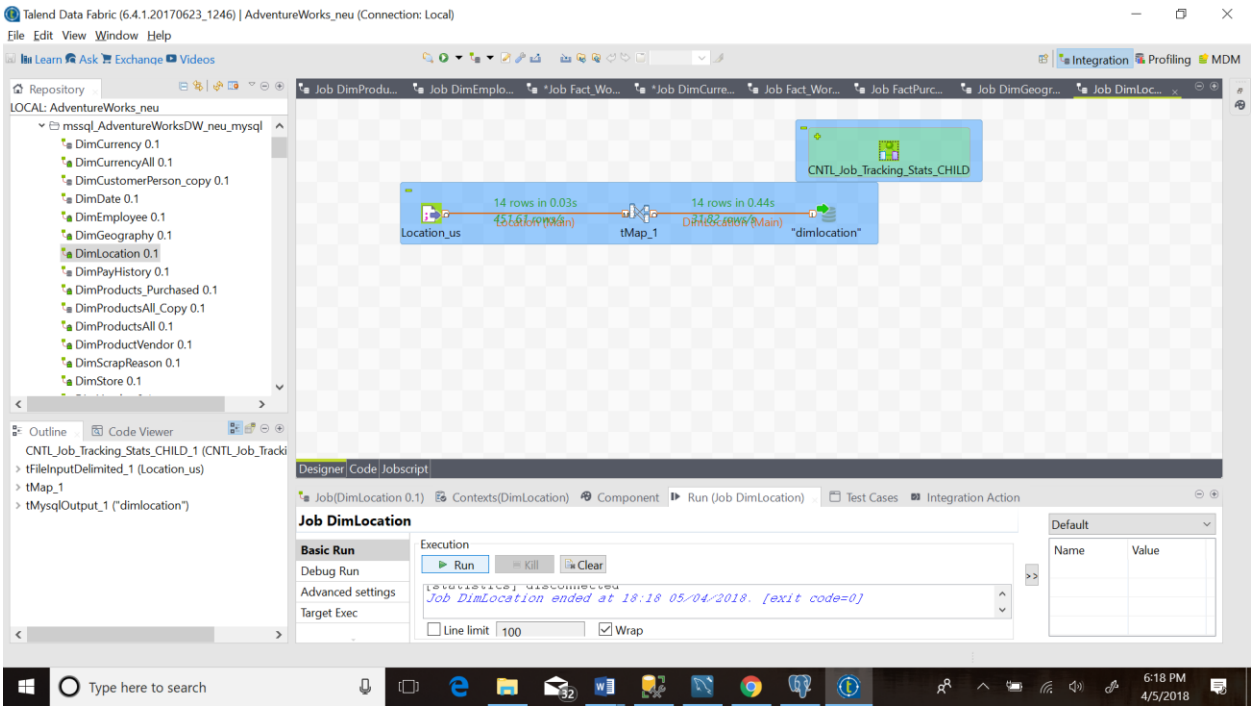
DimGeography

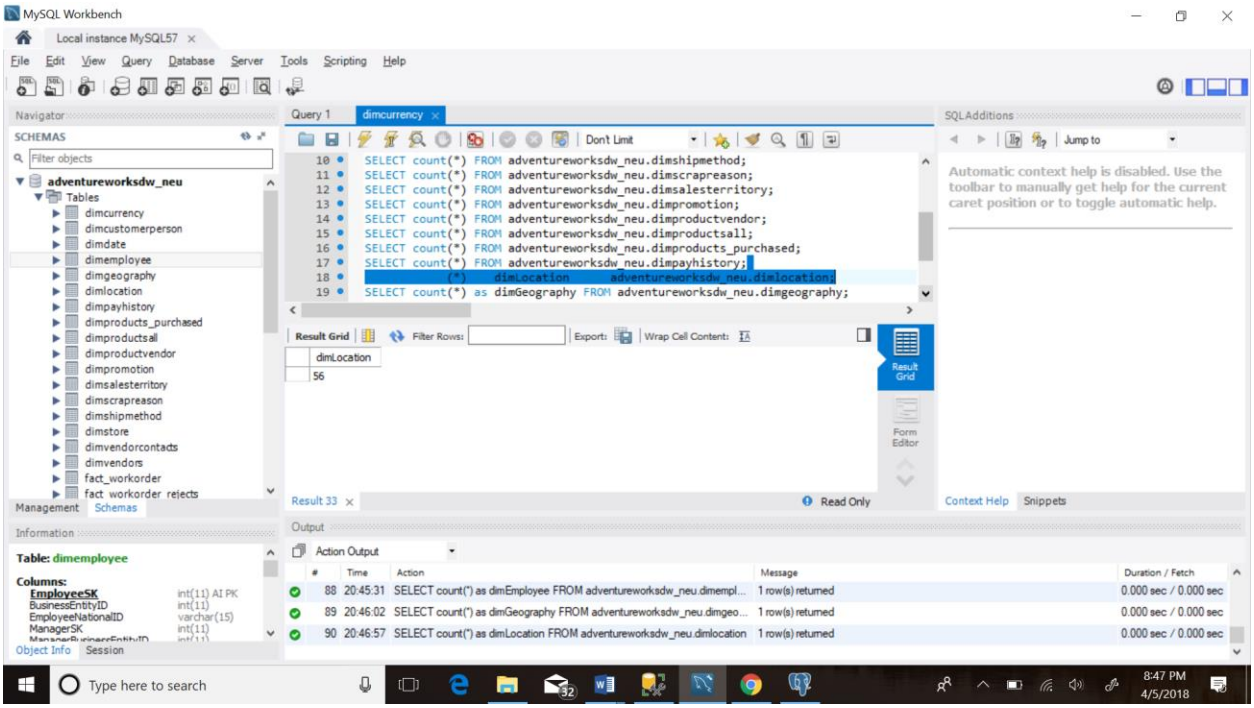




Row Count 402

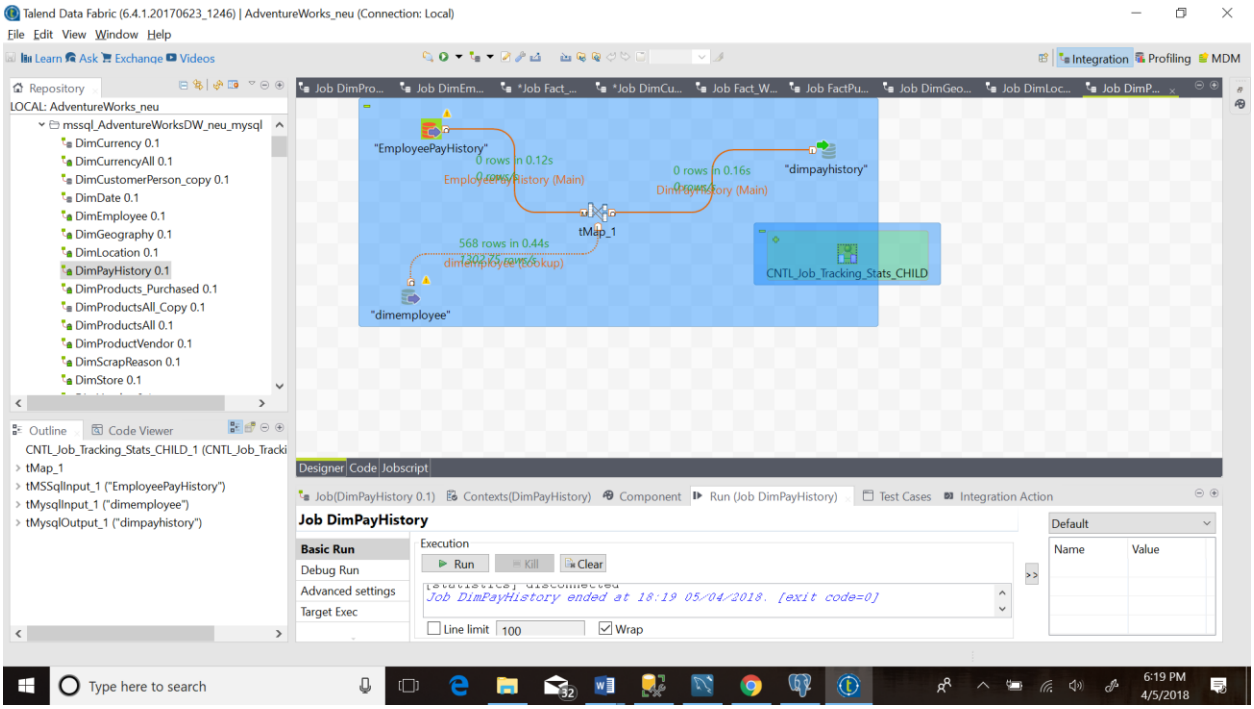
DimLocation

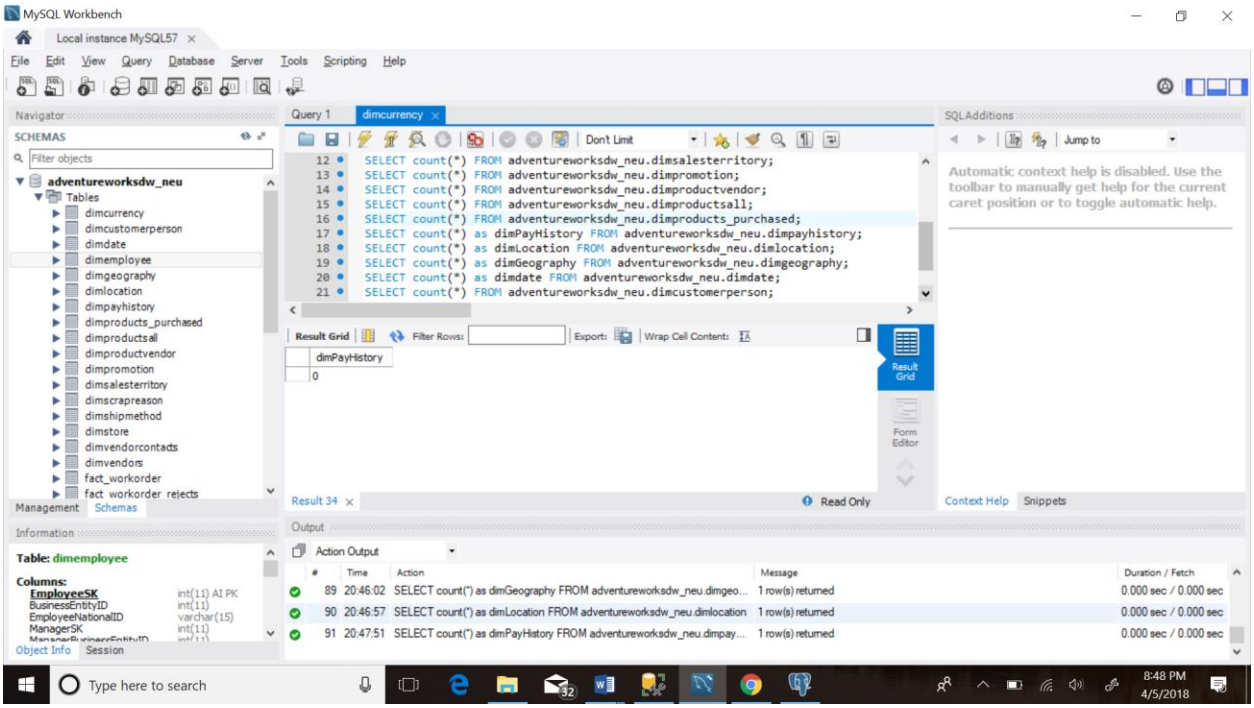




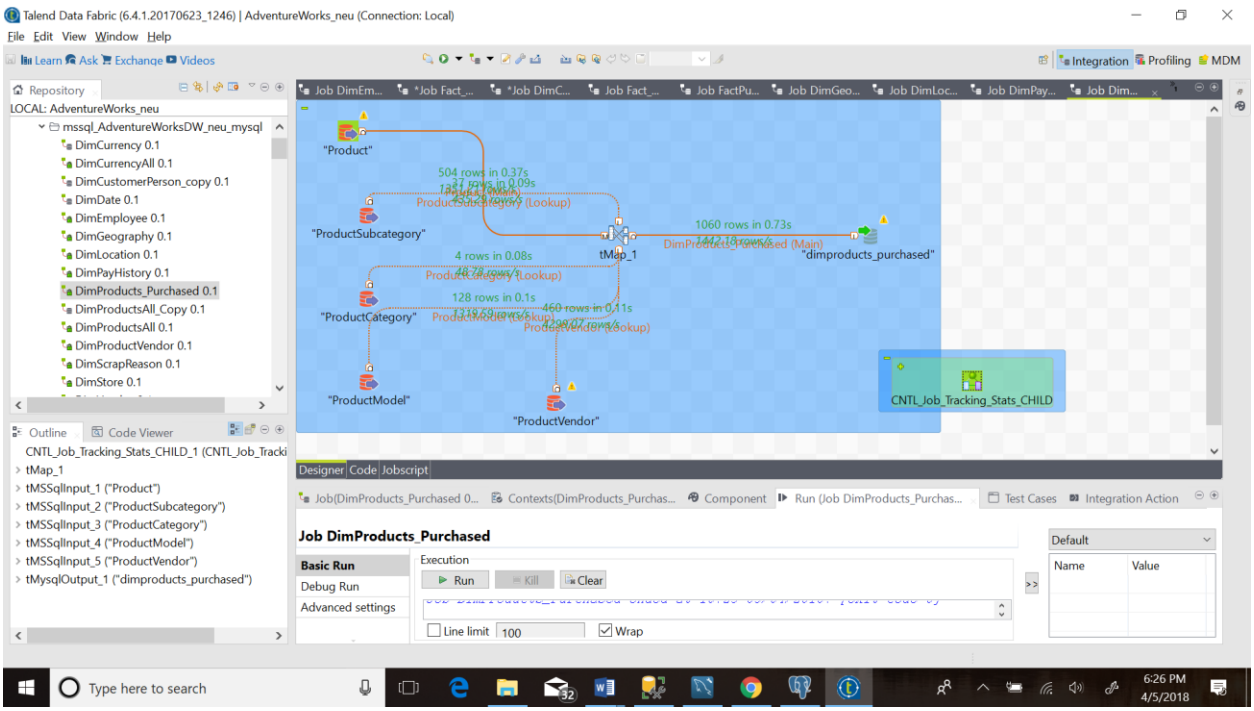
Row Count 14

DimPayHistory



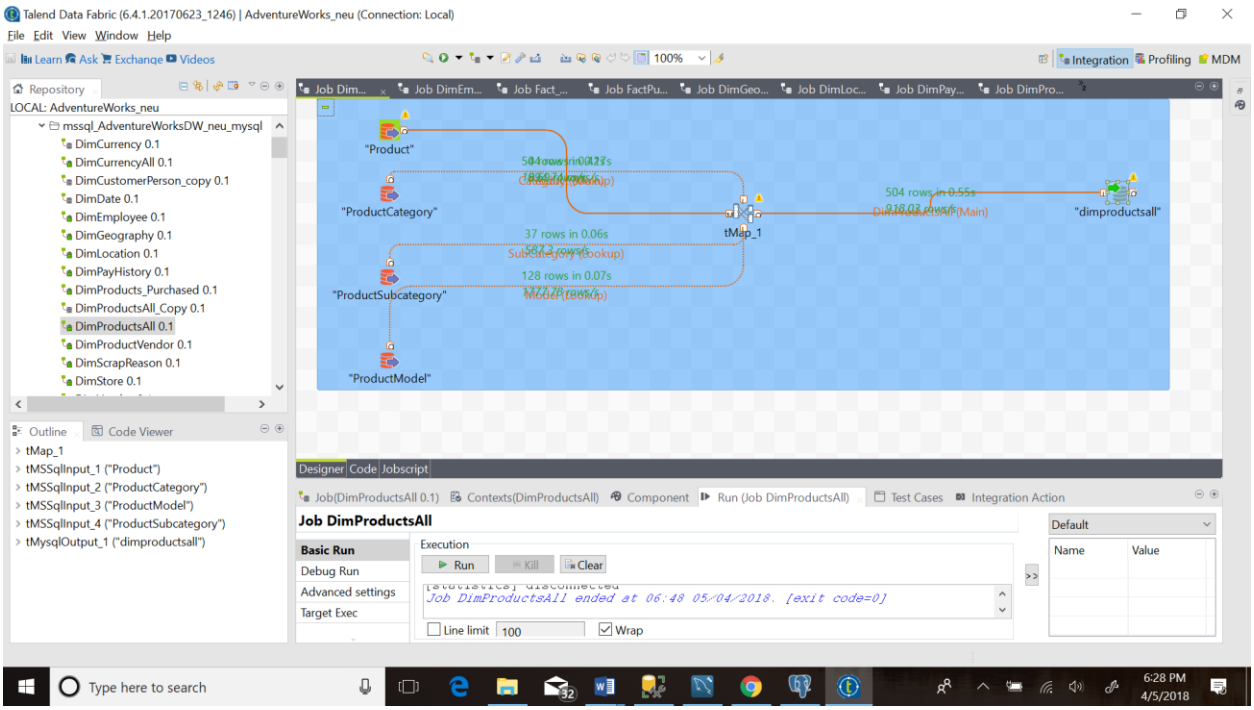


DimProduct_Purchased



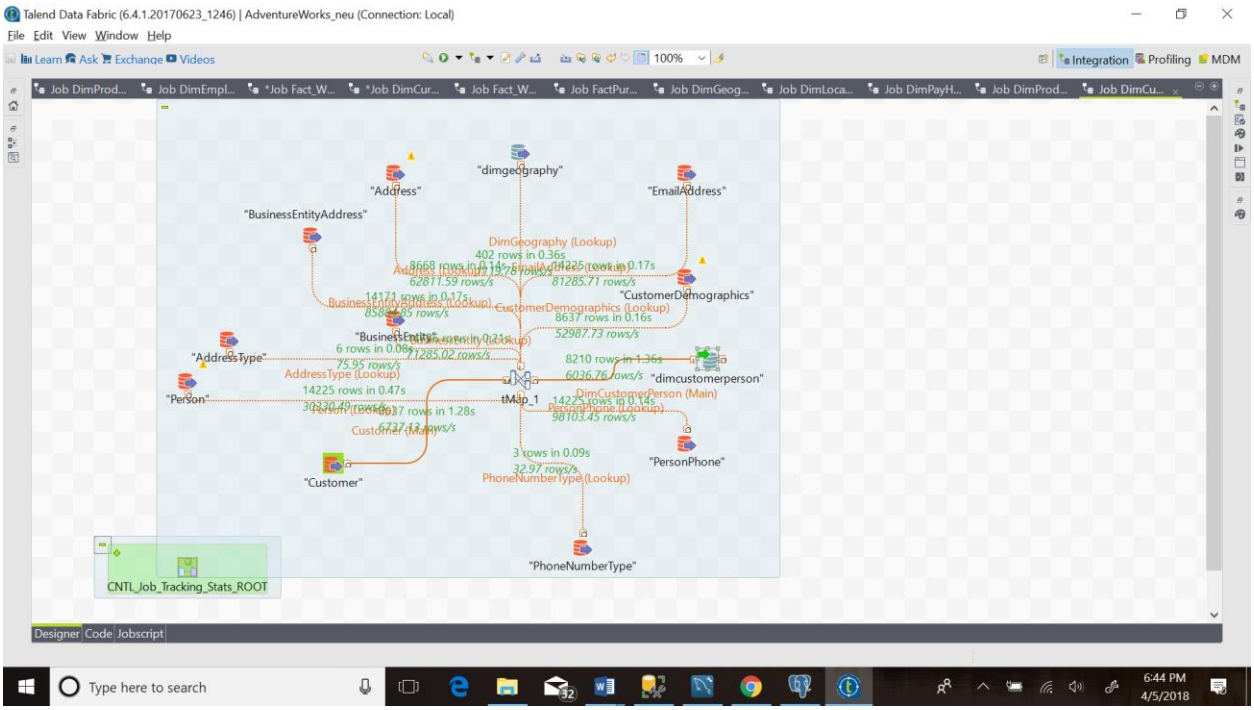
Row Count 1060

DimProductsAll



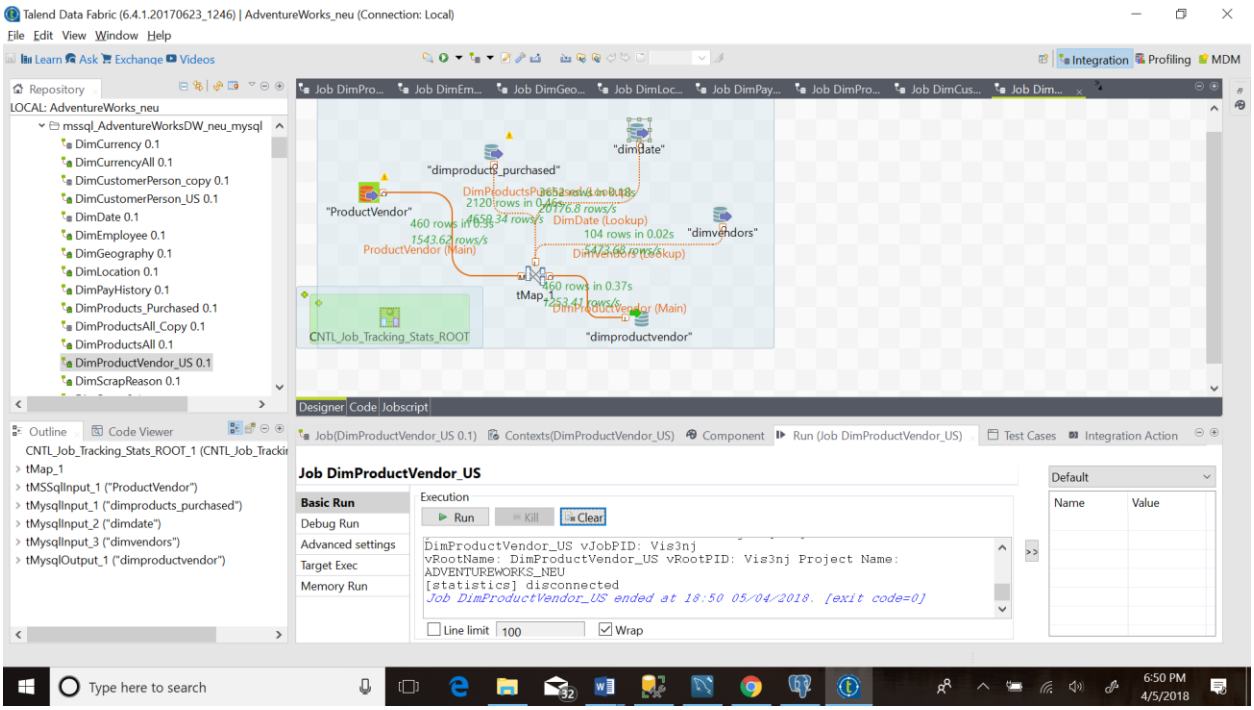
Row Count 504

DimCustomerPerson



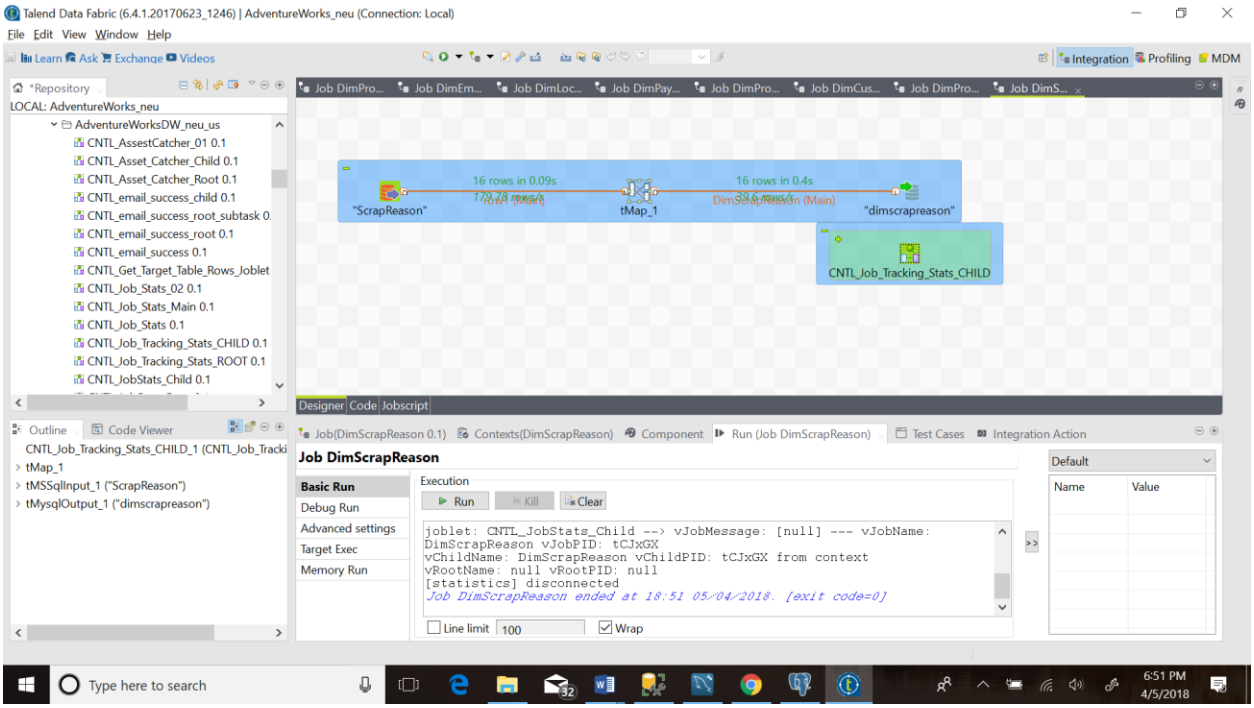
Row Count 8210

DimProductVedors



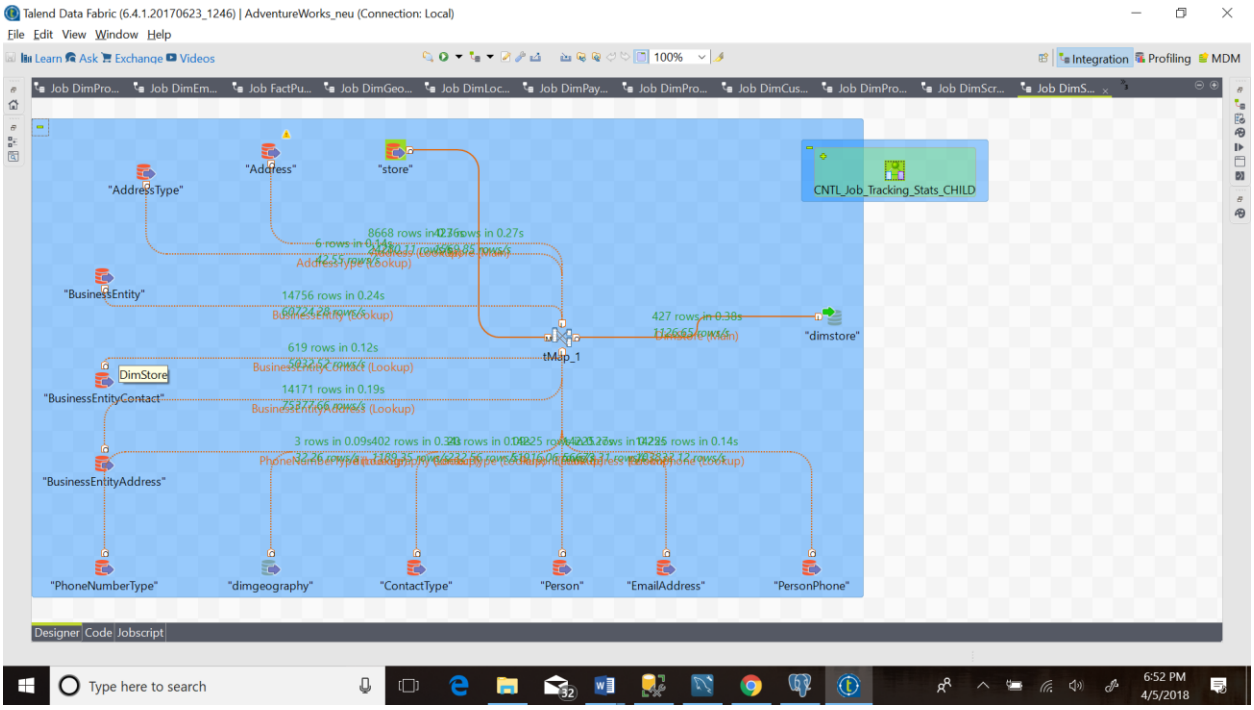
Row Count 104

DimScrapReason



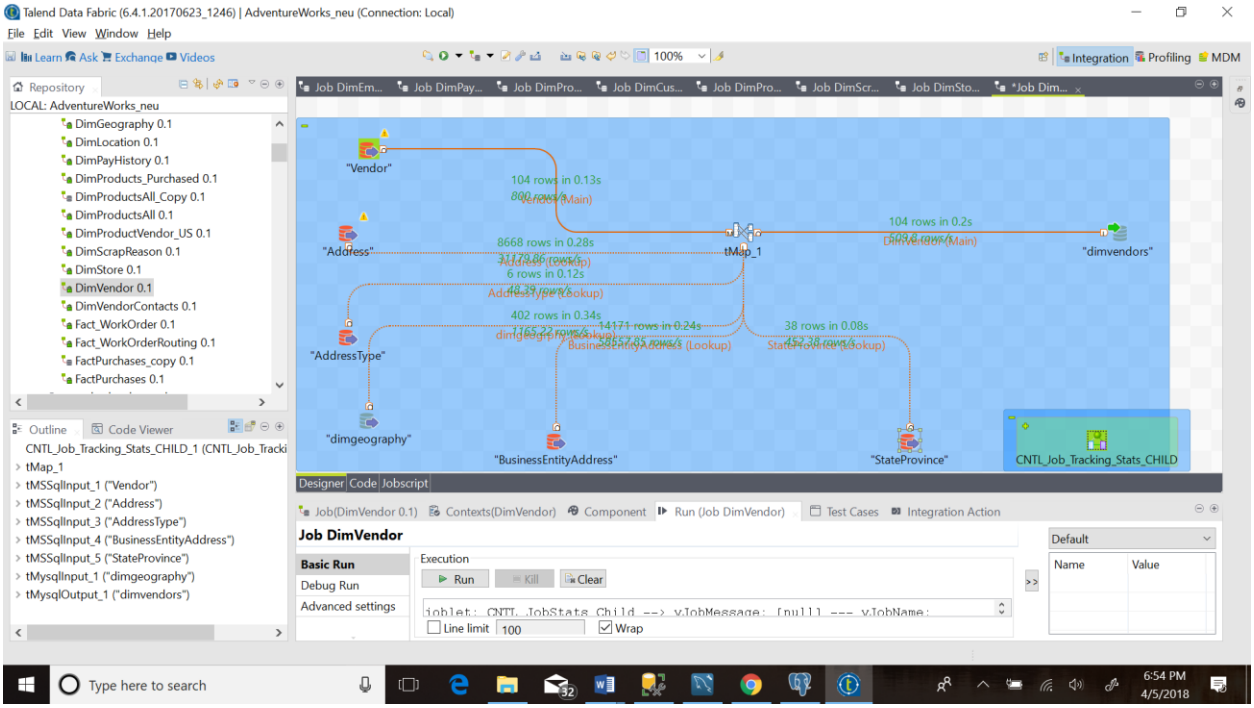
Row Count 16

DimStore



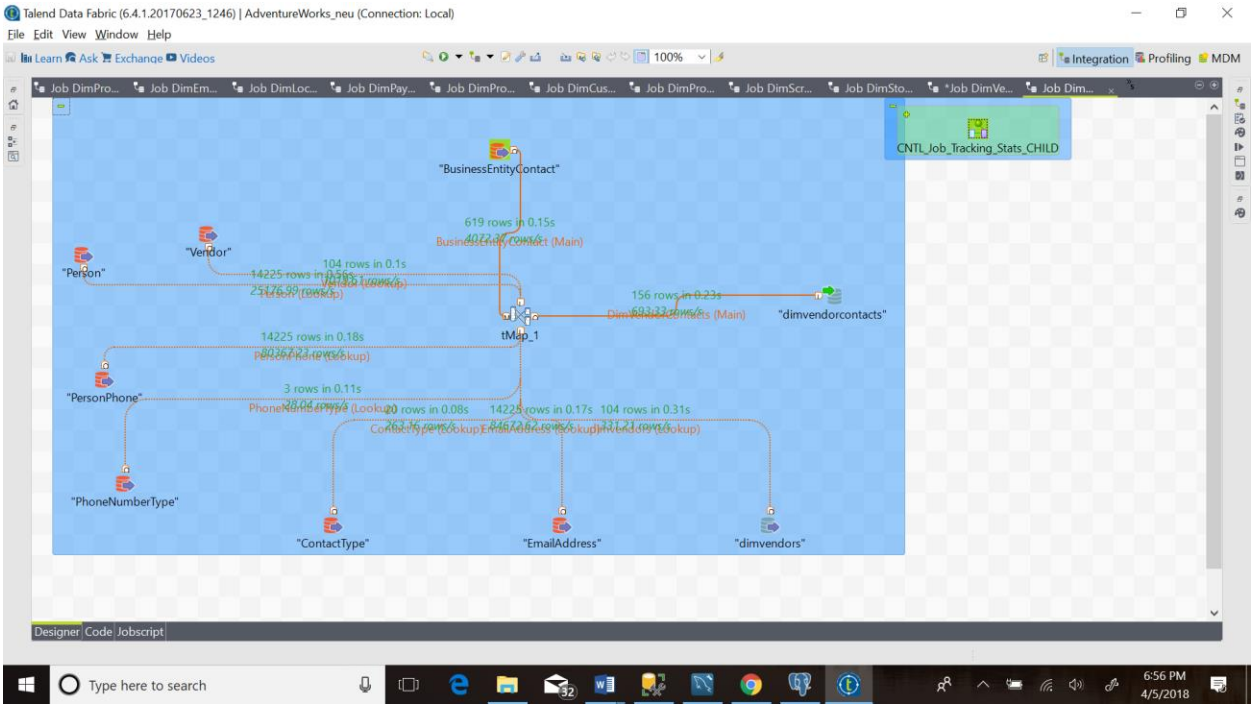
Row Count 427

DimVendors



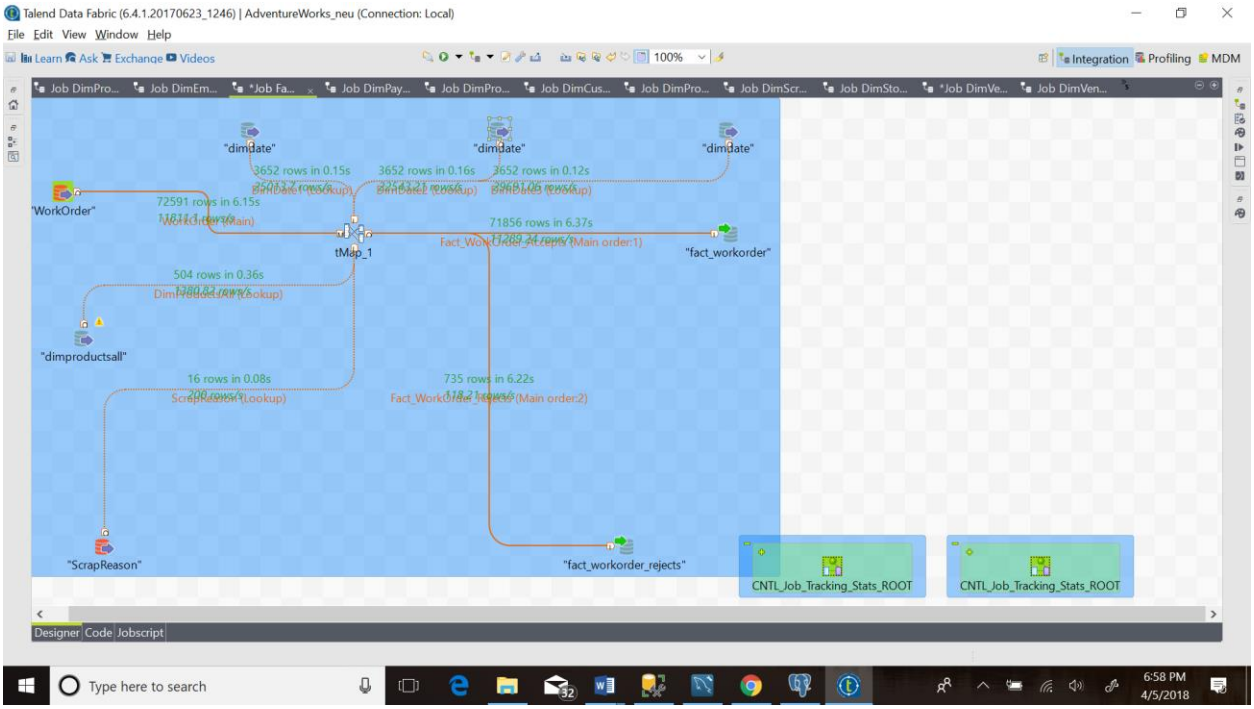
Row Count= 104

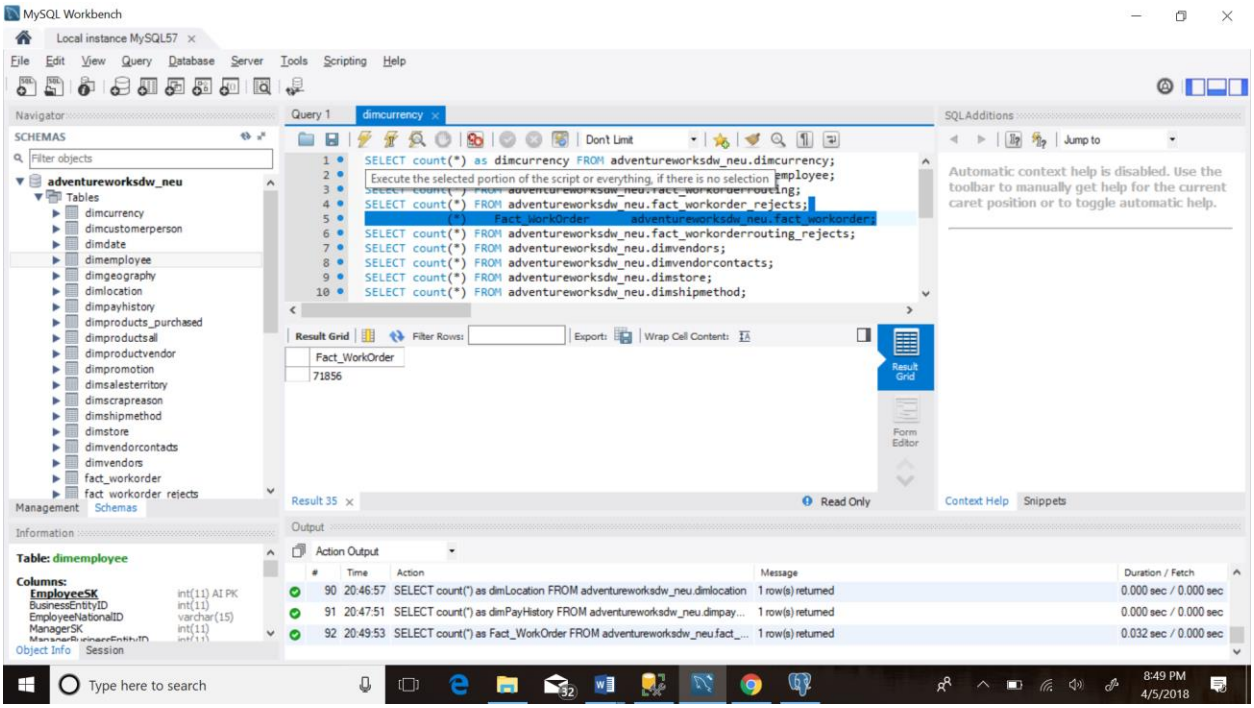
DimVendorContacts



Row Count = 156

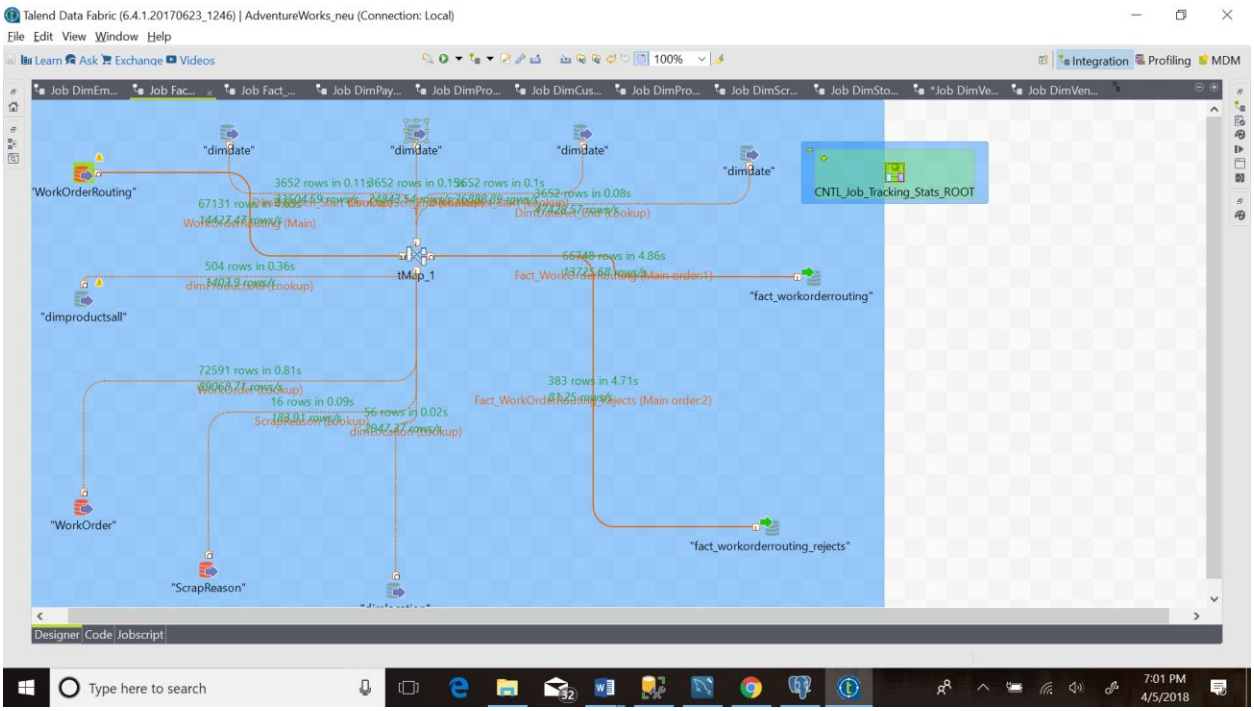
Fact_WorkOrder

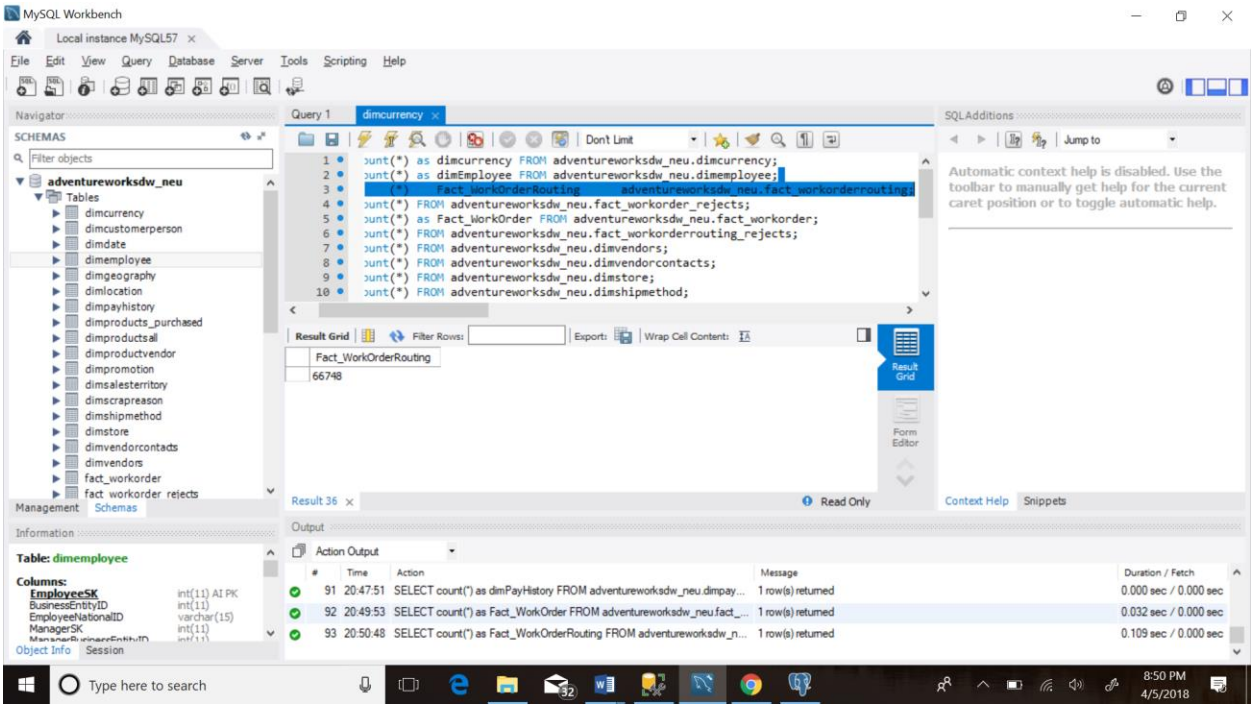




Row Count = 71856, Rejects = 735

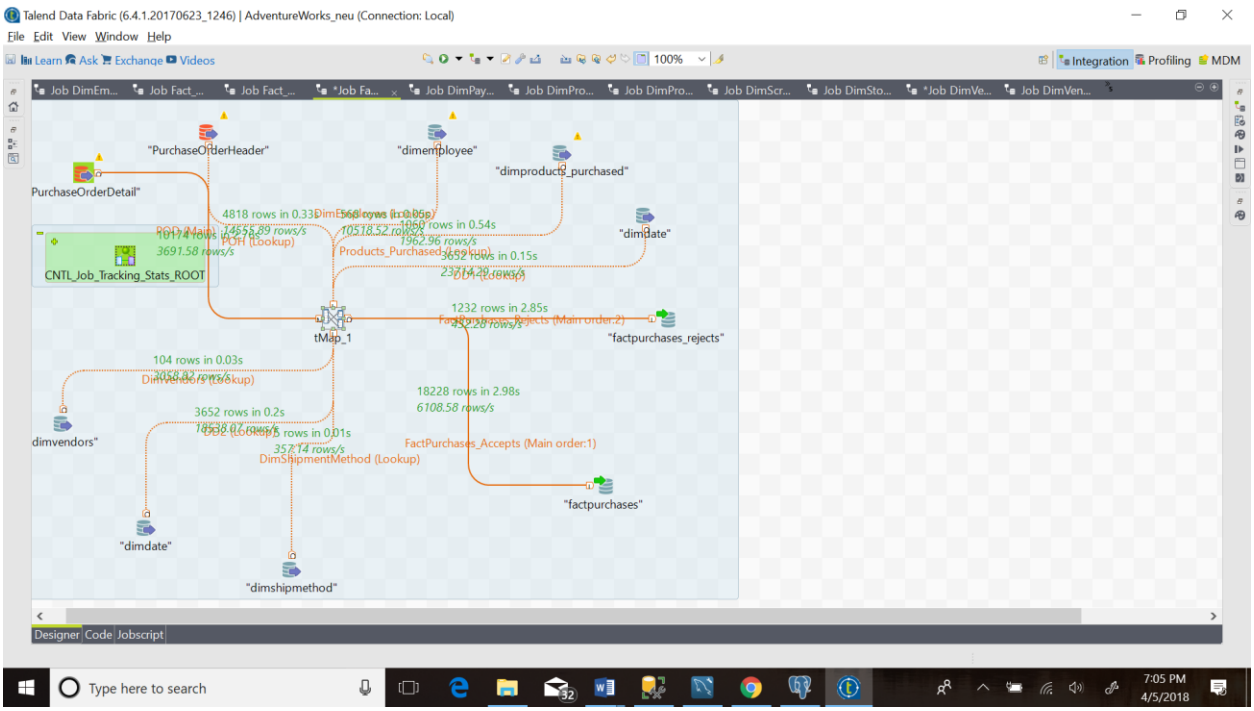
Fact_WorkOrderRouting

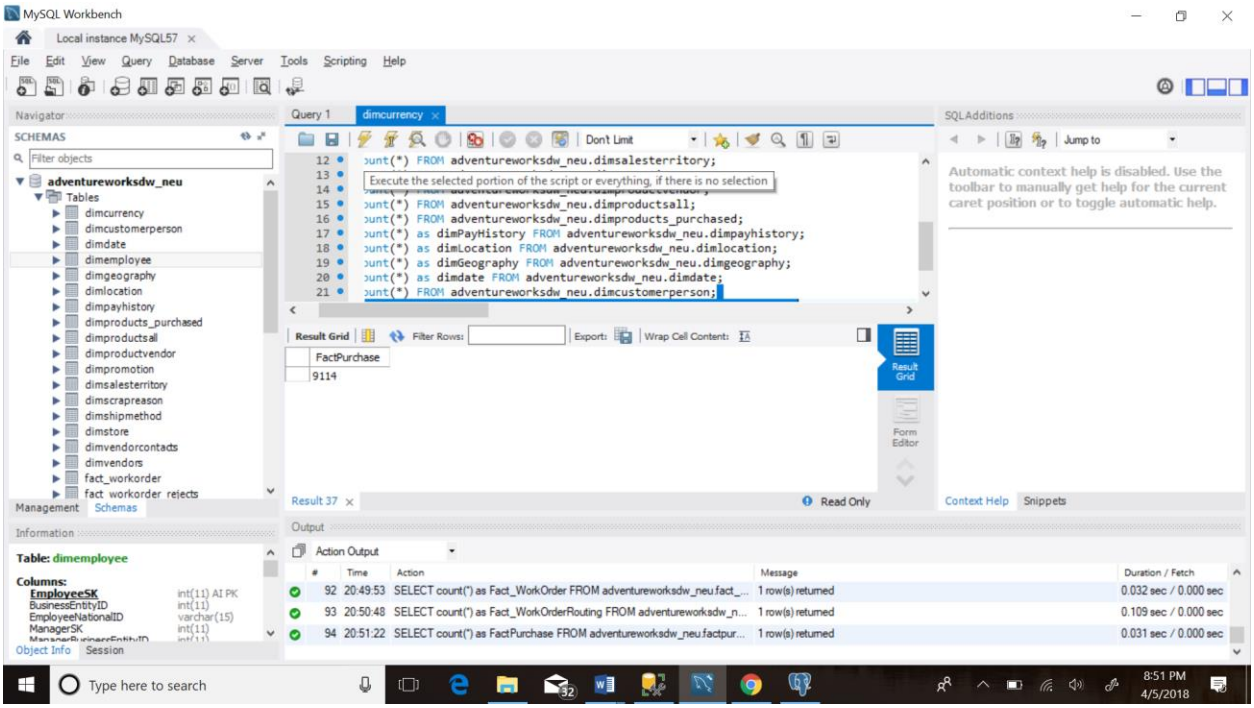




Row Count = 66748, Rejects = 383

FactPurchases





Row Count = 18228, Rejects = 1232

****

Some critical Project Standards:

Each table (dimensions & facts) need to record:

- SOR_ID
- *DI_Job_ID*
- *DI_Create_Date*
- *DI_Modified_Date*

Dimension or Fact Table

Column	Data Type	Key
table_id		PK
attributes….		
DI_Job_ID	**Integer**	**FK**
SOR_ID	**Integer**	**FK**
DI_Create_Date	**DateTime**	
DI_Modified_Date	**DateTime**	

(from BI Guidebook - Figure 12-16: Data Integration Table - Job Audit Columns)

- *SOR_ID* – This is the SOR identifier that will tie this row to a particular SOR. Use this when the table the row is sourced from multiple systems of records and enables the row to be tied to the specific SOR.
- *DI_Job_ID* – the data integration job identifier is be the job id that the data integration tool generated. This identifier is a foreign key to the data integration tool's processing metadata. If that metadata is available, then this link provides a powerful mechanism to analyze data integration processing and performance down to the level of a table's row. *Note: The datatype for this column is specific to the data integration product used.*
- *DI_Create_Date* – This is the date and time that this row was originally created in this table. Often a database trigger is used to insert the current time, but the data integration job could also insert the current time directly.
- *DI_Modified_Date* - This is the most recent date and time that this row has been modified in this table. Often a database trigger is used to insert the current time, but the data integration job could also insert the current time directly. It is often a standard practice to populate this column with an initial value far in the future such as "9999-12-31" rather than leaving it a NULL to avoid queries with NULLs when analyzing this column.

For SQL Server the SQL syntax for these columns is:

- DI_Job_ID `nvarchar(20) NULL`
- DI_Create_Date `datetime NOT NULL DEFAULT (getdate())`
- DI_Modified_Date `datetime NOT NULL DEFAULT (getdate())`

Every Talend job needs to tract job stats & errors

The steps to completing this:

1. Insert Joblet (supplied) that tracks job stats & errors (below)
2. Create columns as specified above
3. Insert values into those columns (below)

Joblets

- Every job created must have a Joblet that tracks stats via tStatCatcher, tLogCatcher & tAssetCatcher.
- For this project the Joblets provided:

- CNTL_Job_Tracking_Stats_ROOT – placed in top level job also referred to as the Root job
- CNTL_Job_Tracking_Stats_CHILD – Placed in every non-root job

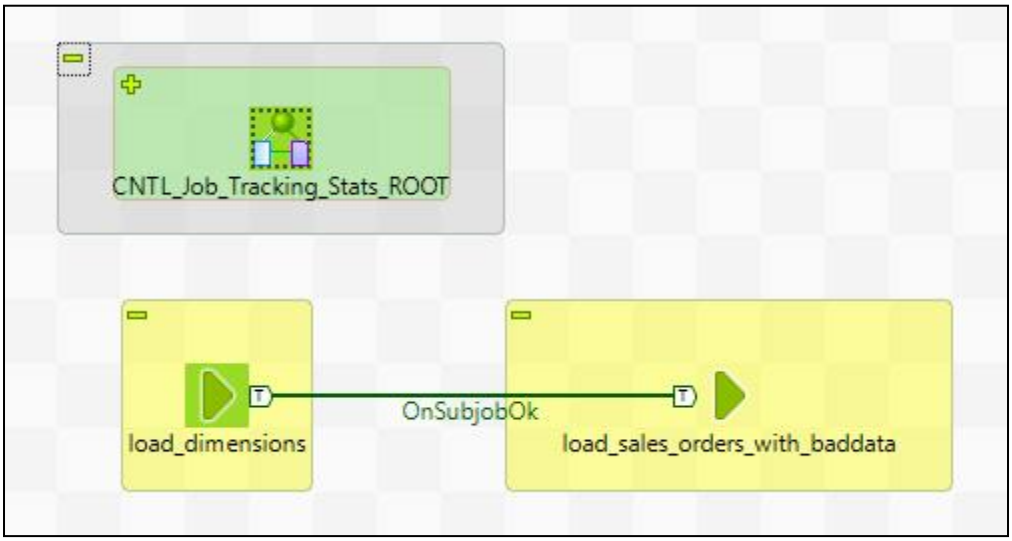


Figure 1: Joblet to track stats & errors in Root job

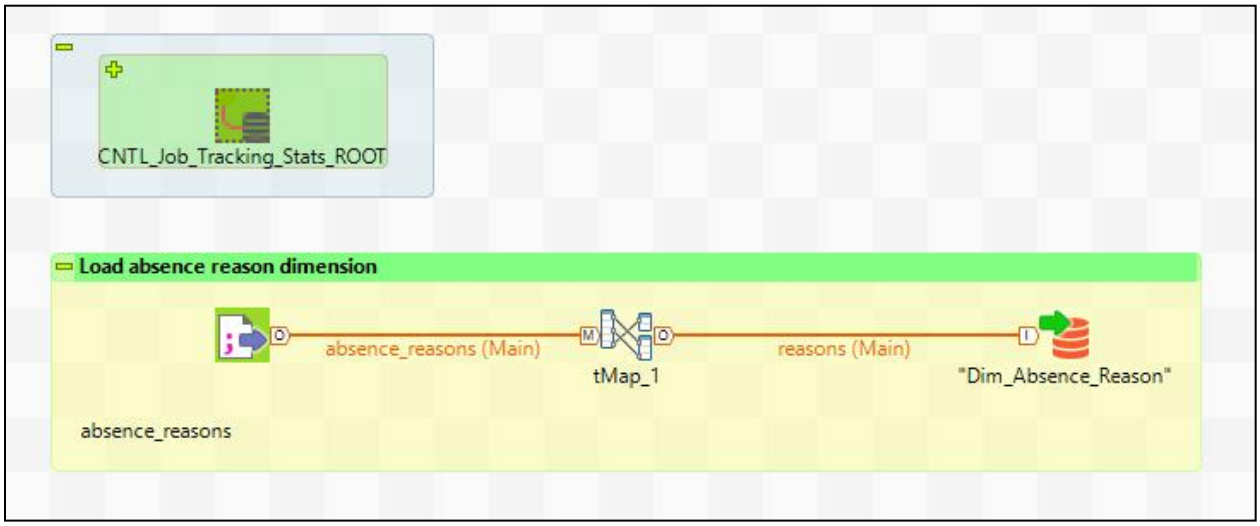


Figure 2: Joblet to track stats & errors in Child job

Insert values into standard columns

- DI_Job_ID - context.getProperty("vRootPID")
- DI_Create_Date - TalendDate.getCurrentDate()
- DI_Modified_Date - TalendDate.getCurrentDate()

context.getProperty("vRootPID")	DI_Job_ID
TalendDate.getCurrentDate()	DI_Create_Date
TalendDate.getCurrentDate()	DI_Modified_Date

Figure 3: tMap inserting values into target tables

Converting a date to a date surrogate key:

In the target table use the following function:

Integer.parseInt(routines.TalendDate.formatDate("yyyyMMdd",row1.TheSourceDate))

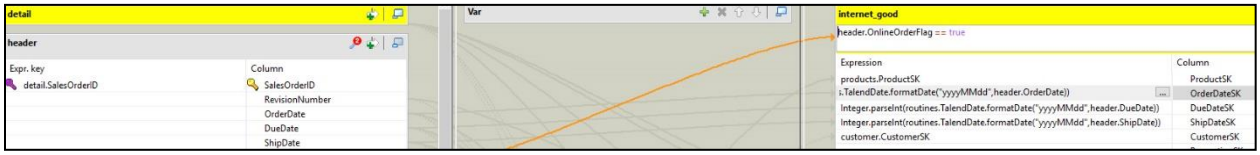


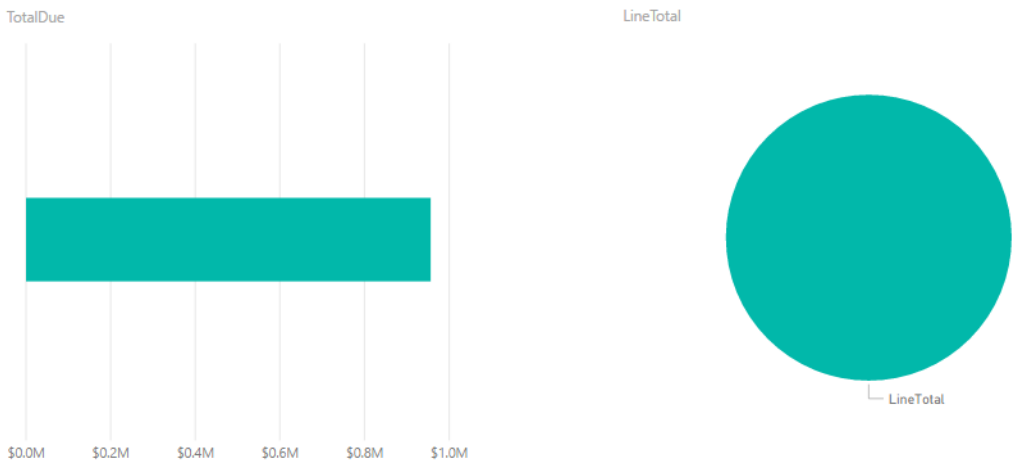
Figure 4: Converting a date to a date surrogate key

The example:

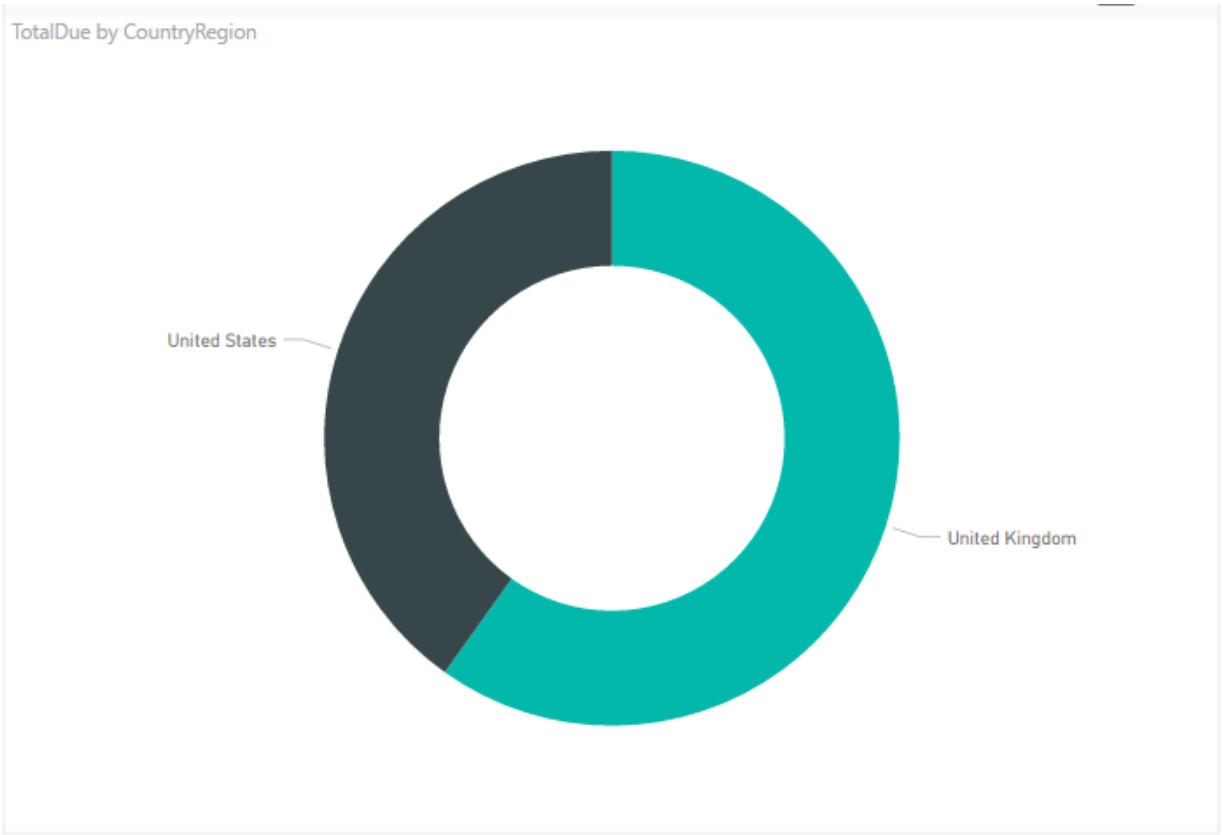
- Source column: header.OrderDate (date or datetime)
- Target column: internet_good.OrderDate_SK (integer)
- Expression:
    - Integer.parseInt(routines.TalendDate.formatDate("yyyyMMdd",header.OrderDate))

Data Visualization

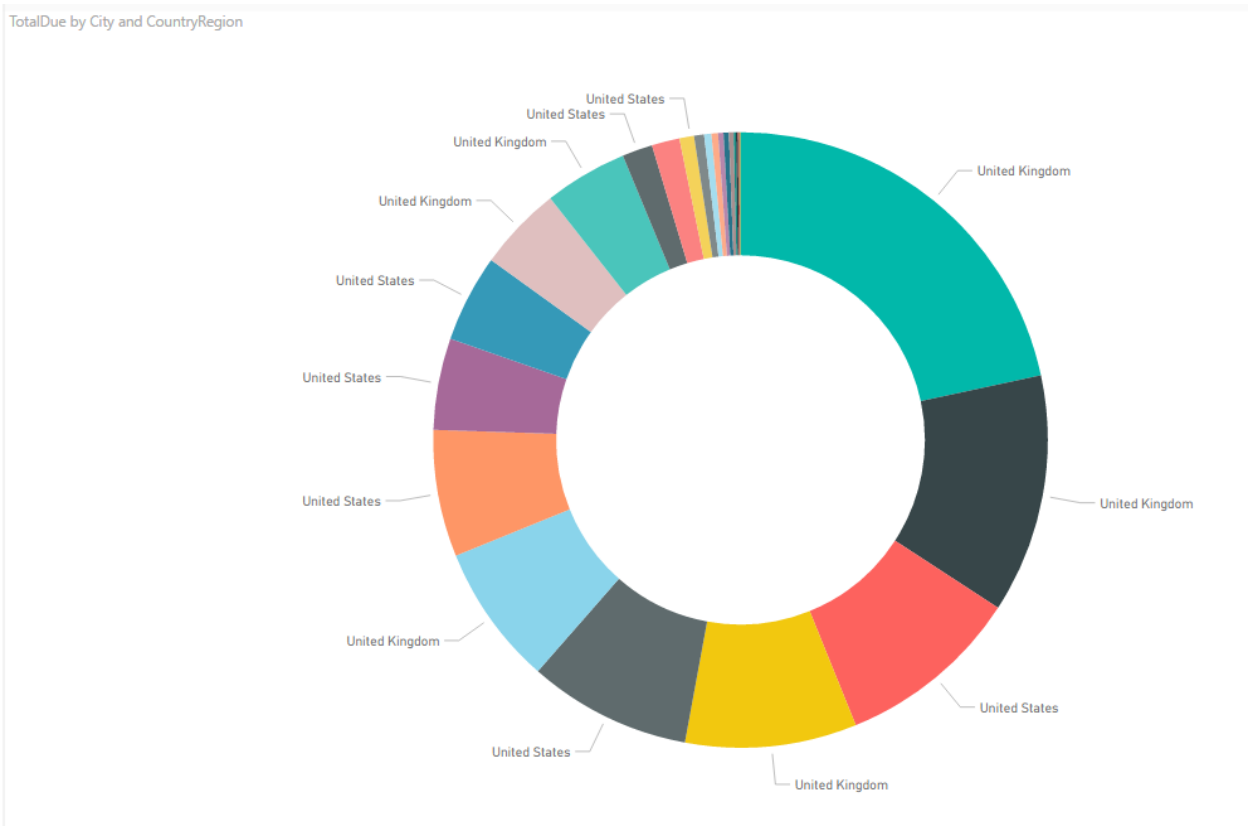
a) Total sales a. Using SalesOrderHeader b. Using SalesOrderDetail



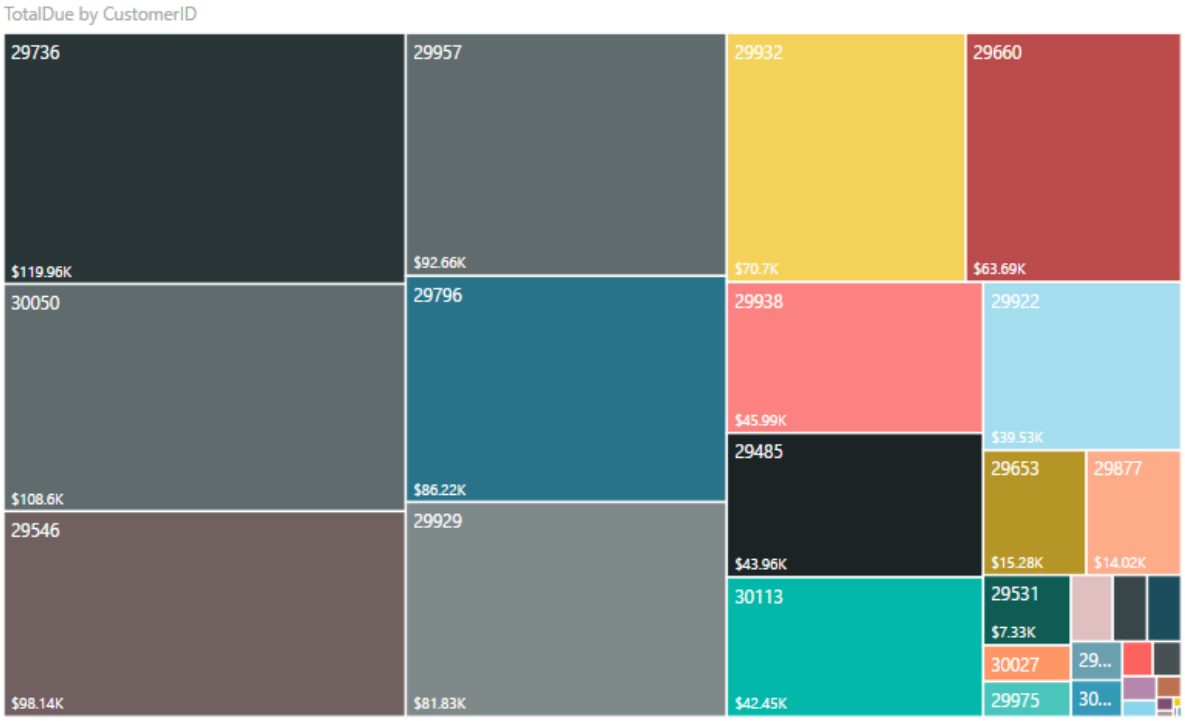
b) Total sales by country – ranked/sorted (highest to lowest)



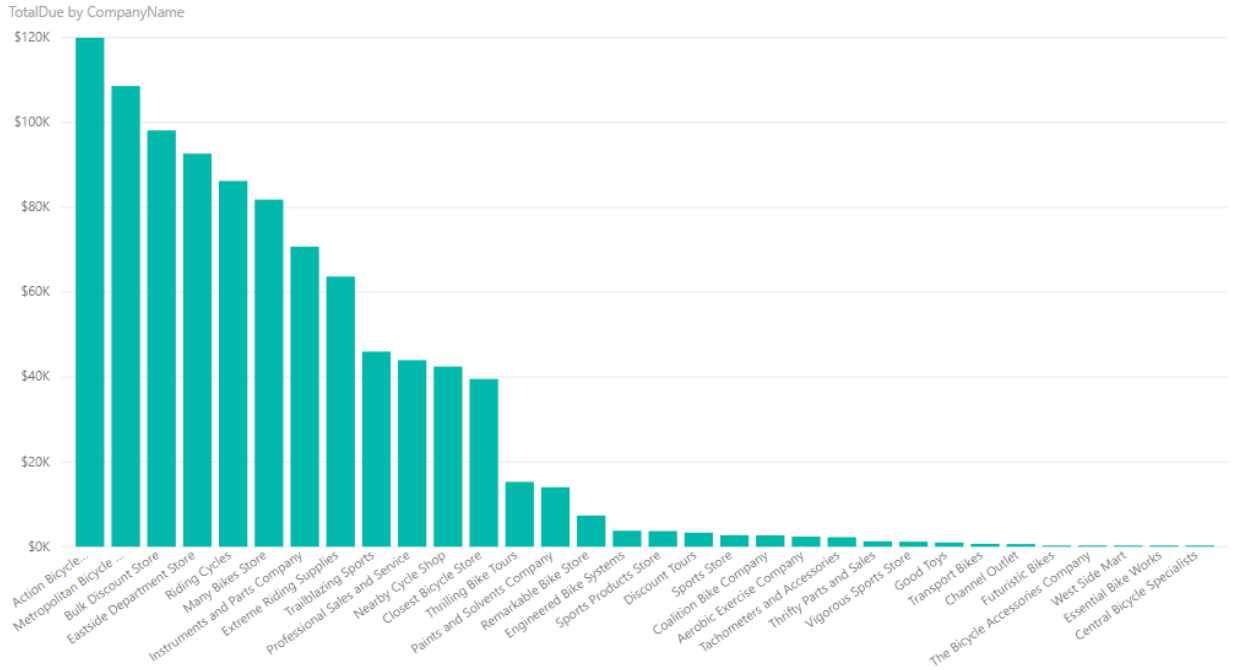
c) Total sales by city & country – ranked/sorted (highest to lowest)



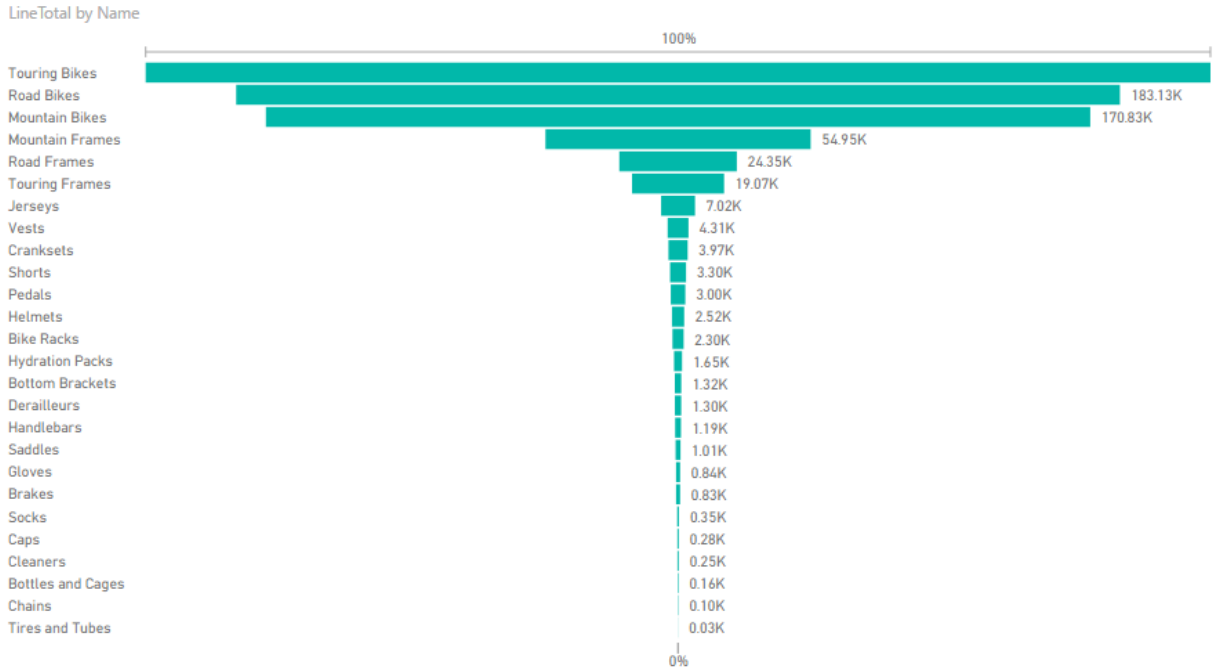
d) Total sales by customer (person) – ranked/sorted (highest to lowest)



e) Total sales by customer (company) – ranked/sorted (highest to lowest)



f) Sales by product category – ranked/sorted (highest to lowest)



g) Sales by product name – ranked/sorted (highest to lowest)