JOHN C. FITZPATRICK – CAPSTONE PROJECT

Table of Contents

Personal Message

Hello and thank you for taking the time to review my project submission. To start with, it's fairly obvious that I ventured way outside of the project guidelines and used Microsoft's Power BI as the dashboarding tool instead of Microstrategy Desktop. I did this mainly because of stability issues. I found that Microstrategy was constantly corrupting my work before I could get half way through the first dashboard. I tried multiple computers with decent components (+16GB Ram, Intel i7, SSDs, etc) and had the same thing happen on all of them. Another student posted that they also had problems with memory issues and when watching my task manager, I saw Microstrategy sucking up 13Gb of memory before crashing. For the same action, Power Bi was at 180Mb. The other student's suggestion was to do all the data modeling in SQL before bringing the data into Microstrategy. However since this bypasses the part of the project for creating a data model, I decided to use Power Bi instead.

I hope that my deviation from the specified direction will be accepted as I believe I have been able to achieve all the same tasks using Power Bi as the Capstone Project requirements and this course are designed to teach and verify.

Since the mentors correcting my submission may not know much about Power Bi, I'll start this off by explaining a few things about this tool and will then get into the dashboard section answers and explain how to use the interactive nature of Power Bi to find specific data points.

Power Bi Intro

Power Bi is Microsoft's platform to build a powerful self-service analytics platform which leverages many of their other platforms abilities. For those who have heard of Power Bi as a tool before July 2015, then you are familiar with the prior version which is completely different and in my opinion, should not be called the same thing as it has caused a lot of confusion.

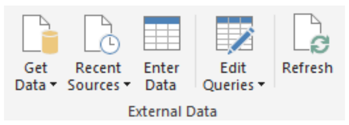
The Power Bi offering currently has two main components, the local which is provided by the Power Bi desktop application and the distributed which is offered through the Power Bi portal hosted on the Azure platform. While it is possible to do most things completely online with the platform there are two parts which are only possible using the Desktop app which are the data modeling and DAX steps. Also, the Desktop app does not have the ability to create the online dashboards which combine views and data from multiple reports.

Power Bi Components

There are 5 main components of Power Bi which fall into alignment with the concepts taught in this course. You start with the ETL process, bring in the data from anywhere. Then create a model linking various parts of data together. After the model is made, an optional step is to add advanced measures which can be calculated using the relationships from the model. Next comes the report building where different visualizations are added based on the data from the model. When the reports are complete, if the information is to be shared, publish it to the online portal. Once the reports are published it's possible to create online dashboards based off of components from various reports.

ETL with Power Bi (PowerQuery)

When working with Power Bi, it has a similar workflow to Microstrategy. The first step is the ETL process. Power Bi does this through the use of many external data connectors. The ETL tool is currently called "Edit Queries" in Power Bi, however it is also known as PowerQuery inside of Excel and uses the same language called "M" for data processing. To access this part of Power Bi, use the "External Data" section of the Desktop application.



Most of the steps needed for typical data ingestion are available on the top ribbon but there are cases when more advanced queries are required. This is possible through the "Advanced Editor" link. For this assignment, I will attach the scripts used in Appendix A for each of the different data sources used. It is also possible to copy a data set from one report to another by copying and pasting the scripts from the Advanced Editor. This saves a significant amount of time when creating new reports with similar data sources. It's also possible to add custom functions which generate data. This is a great way to generate date dimensions which include things like business days taking into account holidays, etc.

Data Modeling with Power Bi (Relationships)

The relationships in the data model are created using a fairly intuitive drag and drop interface where each data source added in the ETL process is displayed and can be linked using either 1-to-1 or 1-to-Many relationships. Many-to-Many is also possible using a calculated bridge table which would need to be created in the previous step.

Data Analysis Expression (DAX) Language (Data)

DAX is a fairly recent language created by Microsoft to make complex data analysis easier for the masses. I like to think of it as a mix of MDX, SQL and Excel in a solution slightly harder to master than basic excel, but which is still fairly simple to get started with. For this project, I used some DAX to create some aggregated measures spanning different data sources. I also used DAX to create a few tables based on other data sources to add dynamic tables to the data model.

Visualization (Report)

This is a drag and drop tool similar to the main window of Microstrategy. Select a chart from the visualization pane, then drag fields from the data sources into the different chart regions, same as in Microstrategy. Once a chart is loaded into a report, depending on the chart type, it has various levels of interaction with other charts. The options are Filter, Highlight or none which are selectable when the "Edit Interactions" option is toggled on the Format menu in the ribbon. For most of this capstone project, I am using the filter option which removes all non-applicable data from other visualizations.

Dashboards (Online Only)

Once a report is published, different parts of the reports can be published to a dashboard. While a report is typically published to display top level metrics upon initial viewing with the ability to drill into more specific information, a dashboard can have different panes with pre-filtered values displayed. It also has the ability to use natural language query.

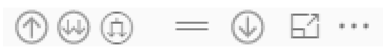
In this project, I created a dashboard using the natural language query tool with the answers for the questions in the different topics already created. Unfortunately, dashboards are only accessible by people with either work or school, Microsoft accounts and can not be viewed using a personal account.

Drill Through Methods

Power Bi has a very powerful drill-through method previously unseen in any other BI tool. Since it is different, it takes a bit to grab the concepts of how to use it but when used correctly, it can give one chart the ability to answer questions which would require many charts before. In my use of this tool, I've built hierarchies 16 levels deep that included time, customer, internal org, and product details.

There are 3 ways to use the Drill Through tool. To access the drill-through options, click or hover over a chart and a local header bar will appear with 6 icons above the chart. The options are also available in the "Explore" menu in the online portal.

I make extensive use of this capability in this project.



1. Show Next Level – This moves the chart axis from displaying one level of the axis hierarchy to the next while ignoring any grouping from previous levels.
2. Expand to Next Level – This expands the next level of the axis hierarchy while keeping the groups from the previous level.
3. Drill Down – Allows the user to drill into a specific part of the chart on the axis by clicking on a chart component. This filters the next level to only include data points which were part of the selected value in the previous level. To use, it works like a toggle, click once on the icon to turn it on and it will change appearance to be a black circle with a white arrow. Click again to disable.

My Submission:

Accessing my Project Submission

There are multiple ways to access my submission.

Desktop App

Either open the file I submitted to Coursera or download the Power Bi file from my OneDrive

John Fitzpatrick – Capstone Project Analysis.pbix - https://1drv.ms/u/s!As75pGudLIZylj6M7nyn4kcSa8nT

Published to the Public

Report - https://goo.gl/d7eFxR

> NOTE: to navigate between the different pages of the project, use the arrows in the middle of the page footer.

In Power Bi Online

Dashboard - https://goo.gl/IsrfAk

> NOTE: If Power-Bi responds with an error about access permissions, please send your email address to me at john@jandtdesign.com and I will add you to the list who can access the dashboard. This requires a Microsoft Work or School account and will not work with a personal Microsoft account. If you do not wish to do this, just use either the local file in the Desktop app or the Public report.

ETL

The ETL process used has a query created for each of the various data tables from the Oracle DW. It is possible to import the full model from Oracle with all the links already made based on the FKs in the DW, however due to the requirements of the capstone project, I decided to import each table separately and create the joins myself. The PowerQuery scripts used are are accessible in Appendix A.

Data Model

Instead of separating out the data model into different DataSets like this assignment suggests to do in Microstrategy, I combined all the tables into one full data model. One of the capabilities that allows a model like this to work inside of Power Bi is the ability to pick which direction cross filters are applied with either **Single** or **Both** as options. In my data model I used Both to link the fact tables and Single for all the dimensions. I also made the active relationship for each of the dimensions as the link to the most granular fact table while creating inactive relationships to the other facts as well. The full model is viewable in Appendix B.

There are also quite a few custom tables, columns and measures I added to the data model using DAX. All of the equations used are listed in Appendix C.

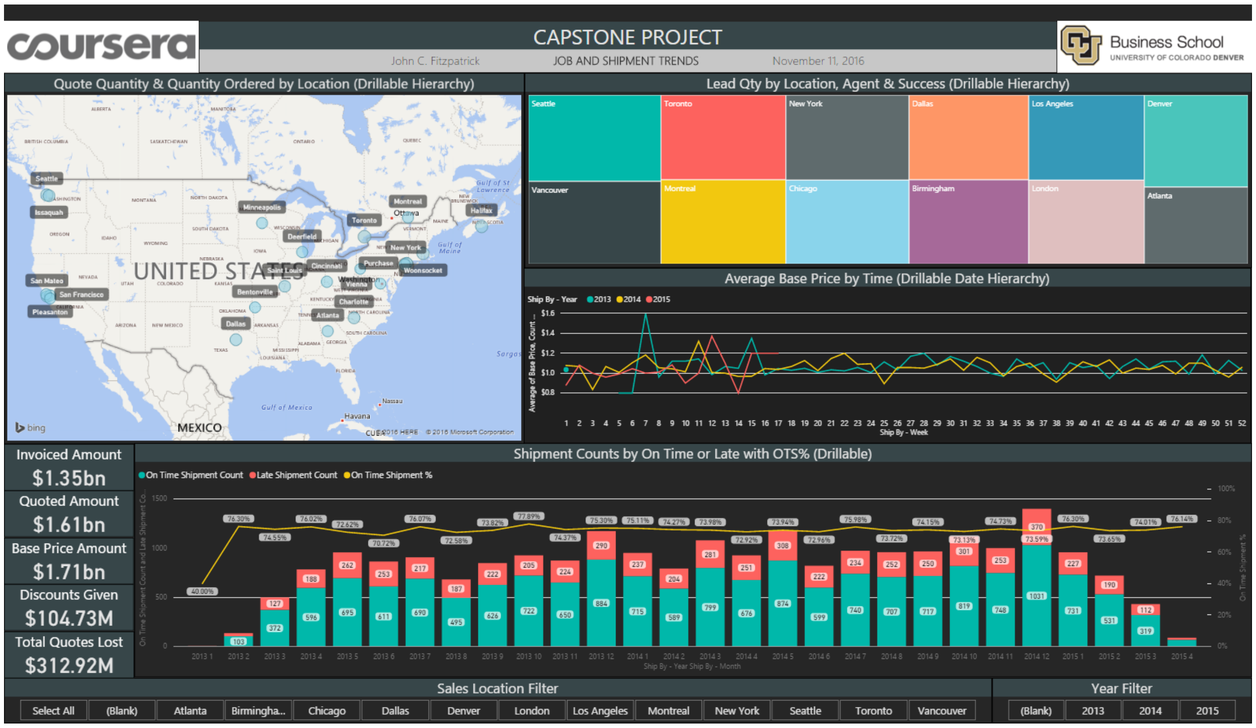
Dashboards

Notes for all

All dashboards have slicers added at the bottom for Location and Year. This is useful in drilling into the data.

I made extensive use of Tooltips to display additional aggregated but non-visual information for each of the various charts. To utilize, mouse over any visual component on any chart and a tooltip will display showing multiple values. One limitation to tooltips is the max lines a tooltip can show. One place where this limitation effects the usefulness of tooltips is on the “Average Base Price by Time” chart on D1. Without a filter for year applied, only 2013 & 2014 will show on the tooltip. Use the year slicer to select only 2015 to see the values in the tooltip.

Dashboard 1



D1 V1 - Quote Quantity and Quantity Ordered by Location

For this chart, I used a map component with a location hierarchy of Country, State, City and Zip. Upon initial viewing the default level is drilled to the City level, use the Drill Through buttons as described above to view the other levels.

The tooltip shows the total revenue generated for the customer locations as well as the number of Jobs, SubJobs and Shipments for customers located in each location.

D1 V2 - Lead Generation by Location and Agent

Instead of a matrix chart I used a Treemap visual with a grouping hierarchy built with Location and Agent. The default view is set to location and can be changed using the Drill through buttons. The Agent level is quite cluttered when viewed for all locations, it’s best to use a sales location filter or customer location filter from V1 to limit the number of values. The chart is also filtered to only include successful leads

The tooltip shows the Total Quote Amount, Base Price Amount, Discount Given, and Revenue generated by either location or agent.

D1 V3 - Job and Shipment Trends-To Locations

I decided to use a line chart instead of a clustered bar chart for this visualization to compare values between years. While this doesn't show the various locations in the chart, you can instead view the values for Average price based on Customer Location through clicking on different locations on the map in V1 or by sales location or agent through V2. You can also use the time periods in V4 or the slicers at the bottom to apply filters to all the visualizations on the page.

The tooltip shows the average of the base price, count of leads in the time, number of jobs and the Quote quantity and amounts.

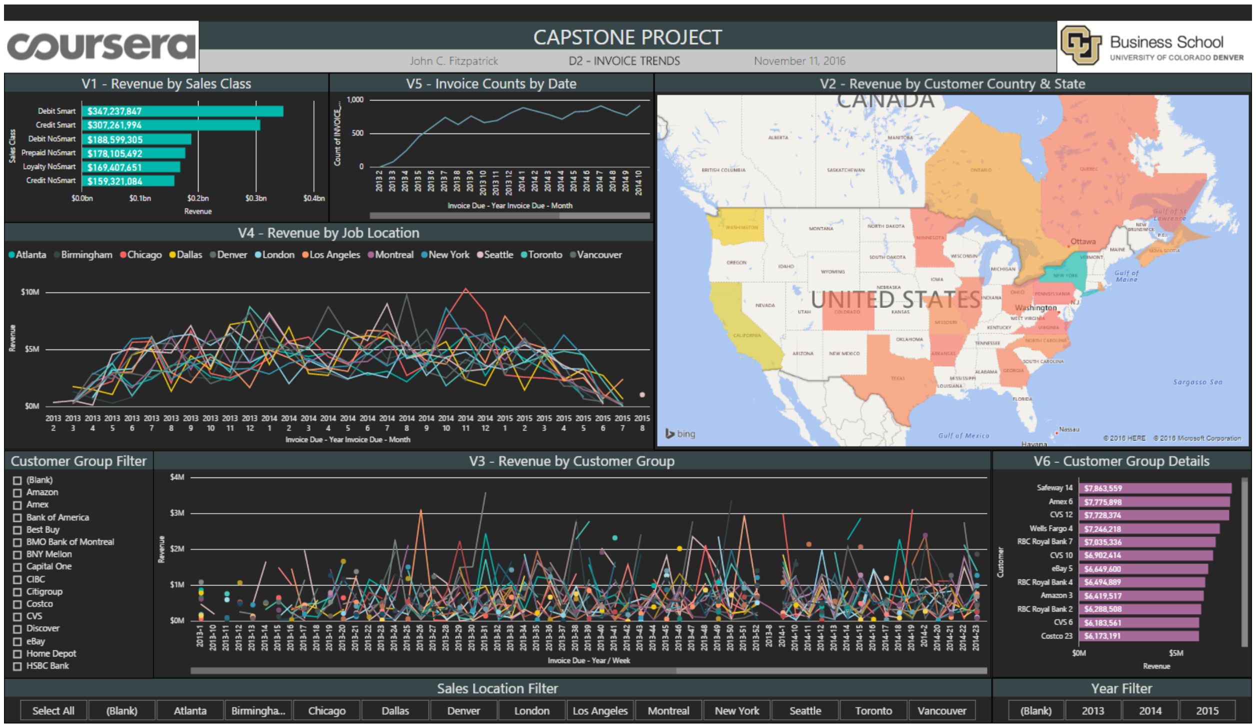
D1 V4 - Late shipment trend by number of days

I decided to use a combination chart for this instead of a table. In trying to measure a locations success the On-Time-Delivery or in this case Shipment % is one of the most important KPIs to accurately track. To measure this successfully, you need to account for all shipments grouped by when they are supposed to have shipped. If the shipment is either shipped >= 1 day after the requested date or if it hasn't shipped yet and the current date is after the requested date, it needs to be counted as late.

The shared axis has four levels based on Year, Month, Week and Fully Shipped. The drill through capability of this axis is best used with the Expand all down to the next level or the toggle to drill into a specific level. The lowest level can be a bit cluttered but the tooltip provides the applicable data. The default level is Expanded to Month.

The tooltip shows the total shipment count, the On-Time shipment count, the Late Shipment count, the On-Time Shipment %, as well as the Actual Qty and Requested Qty.

Dashboard 2



As noted in Appendix A, I added a Customer Group to bin the customers into parent company groups. This makes the customer visualizations easier to use. The slicer in the lower left has all of the top level customer groups and the V6 chart can be drilled down to the individual customers. When combined this allows ranking of separate branches in a top level customer.

D2 V1 - Total invoiced amount by Sales class

I chose a simple bar chart for this visual to display the revenue generated for each Sales Class.

The tooltip displays the total revenue, shipped amount, subjob amount, base price amount and the quoted amount.

D2 V2 - Invoiced amounts by State

I used a highlighted map with a diverging color scheme of red for lowest to green for highest revenue generated per each state. Drill through is not enabled for this chart.

The tooltips include details for revenue, discount given, quoted amount, base price and counts of leads and invoices.

D2 V3 - Customer Invoice Trends

One limitation to Power Bi is that time values can not be included on a dot plot as suggested in the assignment document. To bypass this, I chose to use a line chart at a very granular time axis of weeks (default) drillable to days. I left out the less granular levels of month, quarter and year as they are not very effective visuals.

The Lines are also quite cluttered without applying a filter for a specific customer group using the slicer. However, when selecting a few customers, this chart makes comparison between different customers in a specific time period quite easy.

The tooltip displays the total revenue for the selected time period.

D2 V4 - Invoice Amounts by Date and Location

I chose to use a line chart to display variation between the job locations over time for revenue. It is quite cluttered to start with but can be filtered using the location slicer at the bottom or by selections on other charts.

The tooltip only includes revenue due to the number of locations in the legend. If not for the limitation of the number of values a tooltip can display, I would include the invoice counts over time on this chart and avoid creating V5.

D2 V5 - Invoice Counts by Time

I added this chart to answer the question to see invoice counts by time. The axis hierarchy has level for year, quarter, month, week and day.

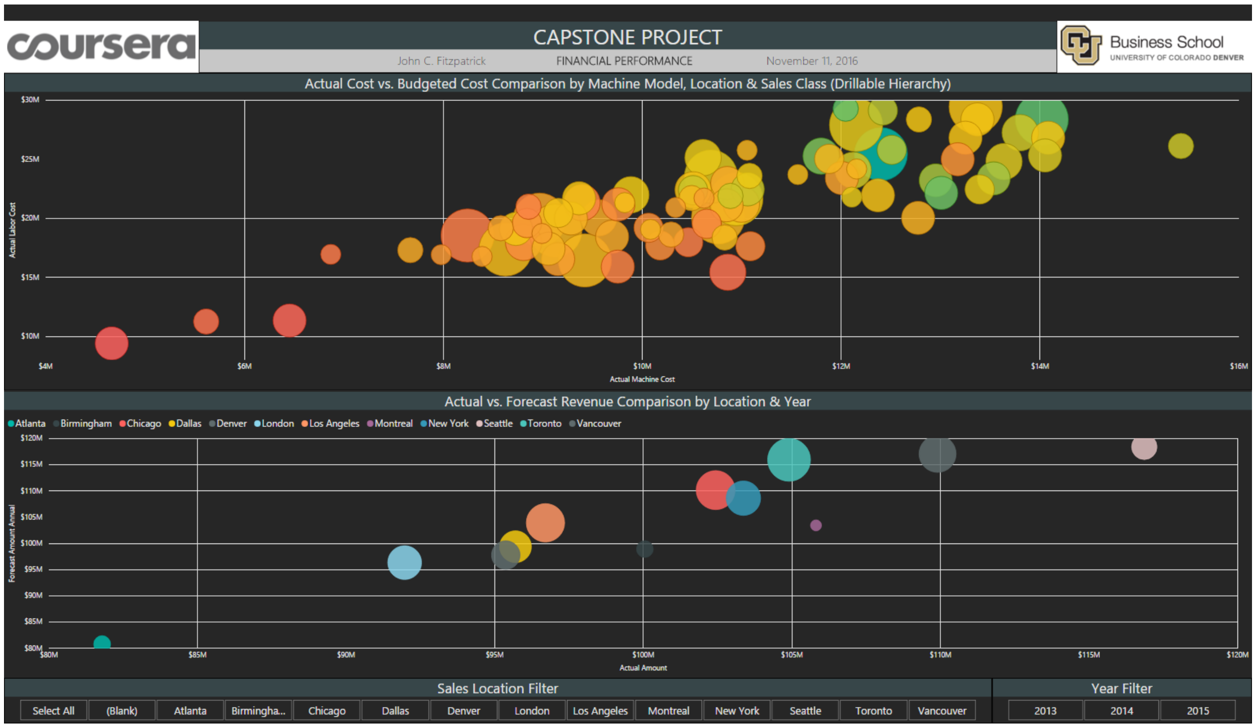
The tooltip displays the revenue, shipped qty, invoiced qty and returned qty.

D2 V6 - Customer Group Invoice Trends

I added this chart to quickly show the specifics for the customers groups and individual customers. There are two levels on the axis, the Customer Group and Customer. At the top level it ranks all the parent customers against each other. At the bottom level, when combined with the customer group filter on the left, ranks the individual customer franchises within a specific parent.

The tooltip displays the total revenue, shipped amount, subjob amount, base price amount and the quoted amount.

Dashboard 3



D3 V1 –

I used the suggested chart type for this visualization as it does the best job of displaying a comparison between the labor and machine costs. However I changed it to have the details as either Model or Location.

The tooltip displays the number of machines, machine type, location, actual machine cost, actual labor cost, total actual cost and the budget cost as well as the margin between budget and actual.

D3 V2 –

I chose to use a scatter chart for the second visualization on D3 as well instead of the combo line & bar chart due to a limitation of Power Bi to not split lines and bars by categories like Microstrategy does. However, for a comparison like this where we want to view multiple values against multiple categories, the scatter chart works best. I included drillable detail splits for all time and year of sales.

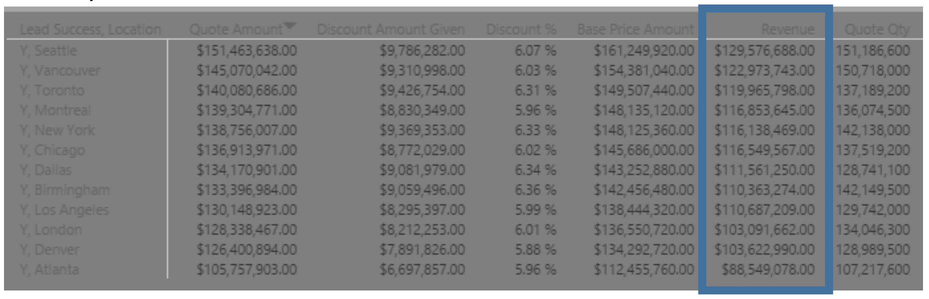
A few filters are also applied. I only include points where Actual Total Cost is not 0. As well as only including the last period for each year. When looking at the forecast, it's monthly value is the pre-aggregated time period goal. Since the Actual amount is cumulative sum for the year, we need to compare the last full value of actual against the annual forecast. To do this we need to filter the period to include 2013-12, 2014-12 and 2015-08 only.

The tooltip displays the Actual amount, Forecast amount as well as the difference between the two and the % difference.

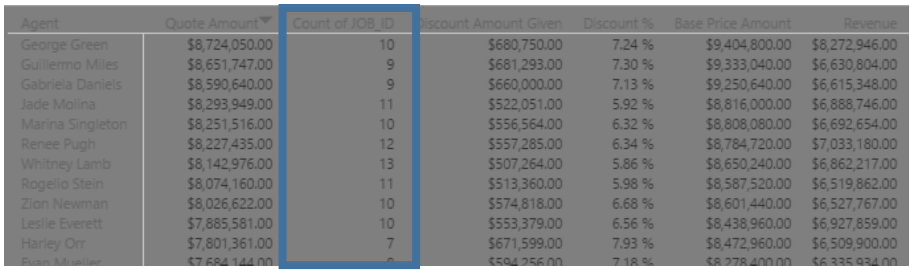
Project Answers

I also answered all of the questions below using the Natural Language Query part of Power Bi. Please see Appendix D, for a screenshot of the dashboard. If you have either a work or school Microsoft account, please send your email address to me at john@jandtdesign.com and I will share the live dashboard with you.

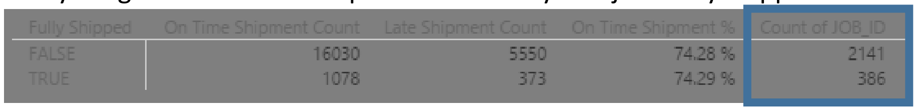
Dashboard 1 – Job and Shipment Trends

- How much revenue does a company generate from its job bookings?
    - D1 shows this in multiple ways. It can be split by Location, Sales Agent or Time. One way to view by company location is to expand V2 "Lead Qty by Location, Agent & Success" chart using the focus mode button in the upper right and then select the "See Data" option on the Explore menu on-line or in the Drill menu in the desktop app. This allows you to see a data table with the actual revenues for each location.
    - 

Lead Success, Location	Quote Amount	Discount Amount Given	Discount %	Base Price Amount	Revenue	Quote Qty
Y, Seattle	$151,463,638.00	$9,786,282.00	6.07 %	$161,249,920.00	$129,576,688.00	151,186,600
Y, Vancouver	$145,070,042.00	$9,310,998.00	6.03 %	$154,381,040.00	$122,973,743.00	150,718,000
Y, Toronto	$140,080,686.00	$9,426,754.00	6.31 %	$149,507,440.00	$119,965,798.00	137,189,200
Y, Montreal	$139,304,771.00	$8,830,349.00	5.96 %	$148,135,120.00	$116,853,645.00	136,074,500
Y, New York	$138,756,007.00	$9,369,353.00	6.33 %	$148,125,360.00	$116,138,469.00	142,138,000
Y, Chicago	$136,913,971.00	$8,772,029.00	6.02 %	$145,686,000.00	$116,549,567.00	137,519,200
Y, Dallas	$134,170,901.00	$9,081,979.00	6.34 %	$143,252,880.00	$111,561,250.00	128,741,100
Y, Birmingham	$133,396,984.00	$9,059,496.00	6.36 %	$142,456,480.00	$110,363,274.00	142,149,500
Y, Los Angeles	$130,148,923.00	$8,295,397.00	5.99 %	$138,444,320.00	$110,687,209.00	129,742,000
Y, London	$128,338,467.00	$8,212,253.00	6.01 %	$136,550,720.00	$103,091,662.00	134,046,300
Y, Denver	$126,400,894.00	$7,891,826.00	5.88 %	$134,292,720.00	$103,622,990.00	128,989,500
Y, Atlanta	$105,757,903.00	$6,697,857.00	5.96 %	$112,455,760.00	$88,549,078.00	107,217,600

- How many jobs does each sales agent book?
    - To see this on D1, use V2 and drill all to the Sales Agent level of the hierarchy, then either use the tooltips or click on the Focus Mode / See Data options to view the number of jobs for each sales agent.
    - 

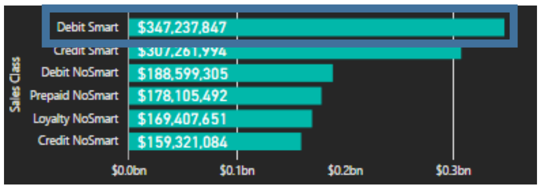
Agent	Quote Amount	Count of JOB_ID	iscount Amount Given	Discount %	Base Price Amount	Revenue
George Green	$8,724,050.00	10	$680,750.00	7.24 %	$9,404,800.00	$8,272,946.00
Guillermo Miles	$8,651,747.00	9	$681,293.00	7.30 %	$9,333,040.00	$6,630,804.00
Gabriela Daniels	$8,590,640.00	9	$660,000.00	7.13 %	$9,250,640.00	$6,615,348.00
Jade Molina	$8,293,949.00	11	$522,051.00	5.92 %	$8,816,000.00	$6,888,746.00
Marina Singleton	$8,251,516.00	10	$556,564.00	6.32 %	$8,808,080.00	$6,692,654.00
Renee Pugh	$8,227,435.00	12	$557,285.00	6.34 %	$8,784,720.00	$7,033,180.00
Whitney Lamb	$8,142,976.00	13	$507,264.00	5.86 %	$8,650,240.00	$6,862,217.00
Rogelio Stein	$8,074,160.00	11	$513,360.00	5.98 %	$8,587,520.00	$6,519,862.00
Zion Newman	$8,026,622.00	10	$574,818.00	6.68 %	$8,601,440.00	$6,527,767.00
Leslie Everett	$7,885,581.00	10	$553,379.00	6.56 %	$8,438,960.00	$6,927,859.00
Harley Orr	$7,801,361.00	7	$671,599.00	7.93 %	$8,472,960.00	$6,509,900.00
Evan Mueller	$7,684,144.00	[illegible]	$594,256.00	7.18 %	$8,278,400.00	$6,335,934.00

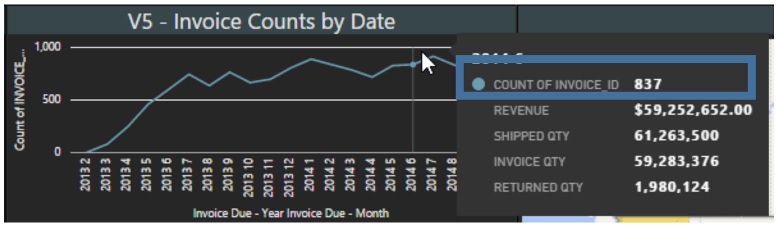
- How many jobs have not yet shipped or have only partially shipped?
    - To view this on D1, use V4 use the Drill through buttons to drill up to the top level and use the "Go to the next level" button to go all the day down to "Fully Shipped". Then click See Data.
    - To create this metric, I created a separate table in PowerQuery which compared the Shipped Qty vs Requested Qty for each job. The results show a dismal rate of shipping everything the customers requested with only 386 jobs fully shipped out of 2527.
    - 

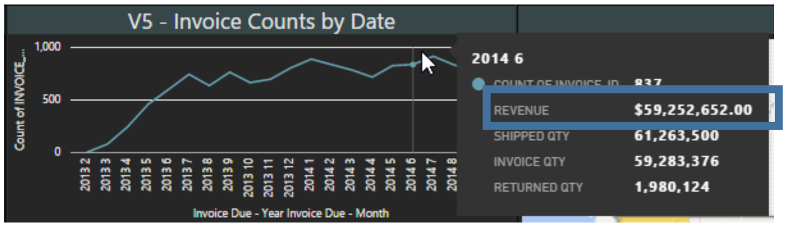
Fully Shipped	On Time Shipment Count	Late Shipment Count	On Time Shipment %	Count of JOB_ID
FALSE	16030	5550	74.28 %	2141
TRUE	1078	373	74.29 %	386

Dashboard 2 – Invoice Trends

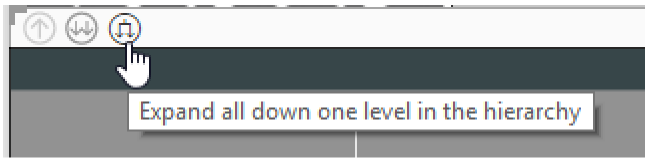
- Which sales class generate the highest invoice amounts?

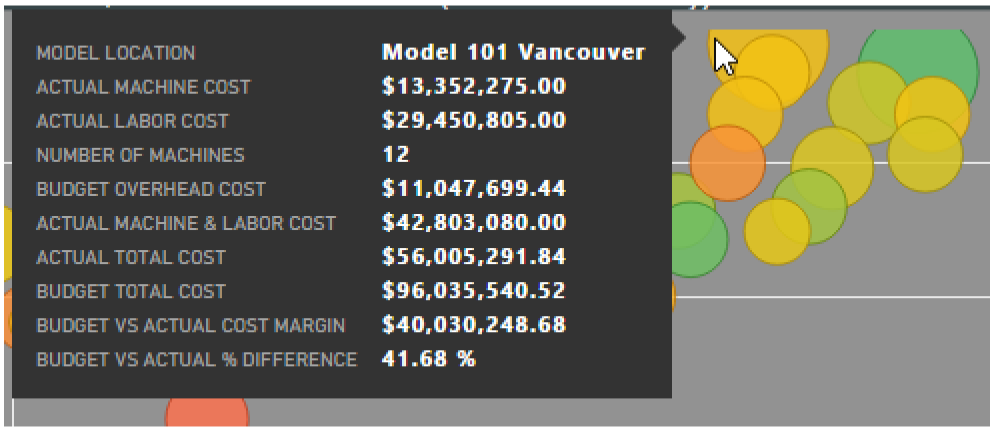
    - This can be seen in D1 V1 without any drill through needed.
    - Debit Smart is the overall most effective Sales Class.
    - 

- How many invoices are generated for a time period?
    - Chart D2 V5 was created to display this value.
    - 

- What is the total amount invoiced for a time period?
    - This is also included in chart D2 V5.
    - 

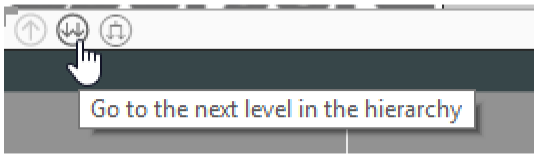
Dashboard 3 – Financial Performance

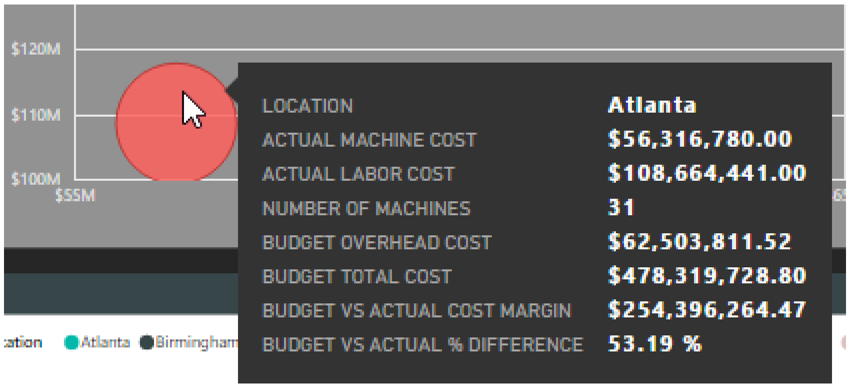
- Determine the location and the machine which have the highest overall machine and labor cost.
    - The answer is **Model 101 in Vancouver** with a combined value of **$42,803,080.00** for the machine & labor costs.
    - Can be determined by Expanding all down one level in the hierarchy and mousing over the largest, highest and furthest right dots to find the one with the highest *Actual Machine & Labor Cost*.
        - 



MODEL LOCATION	Model 101 Vancouver
ACTUAL MACHINE COST	$13,352,275.00
ACTUAL LABOR COST	$29,450,805.00
NUMBER OF MACHINES	12
BUDGET OVERHEAD COST	$11,047,699.44
ACTUAL MACHINE & LABOR COST	$42,803,080.00
ACTUAL TOTAL COST	$56,005,291.84
BUDGET TOTAL COST	$96,035,540.52
BUDGET VS ACTUAL COST MARGIN	$40,030,248.68
BUDGET VS ACTUAL % DIFFERENCE	41.68 %

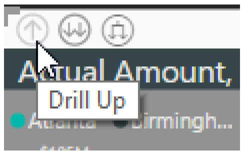
- Determine which location has the lowest budget overhead cost.
  - The answer is **Atlanta** with a total budgeted overhead cost of **$62,503,811.52**
  - Can be determined by drilling the 2nd level of the details hierarchy on the top chart, then looking for the dot that is the darkest red. To see the value over the mouse cursor over the dot and read the value from the tooltip.

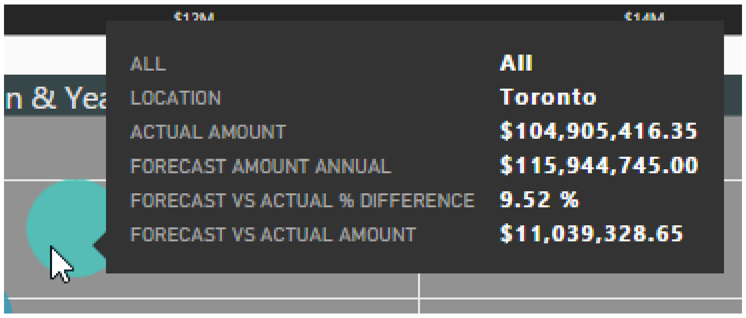
1. 

2. 

LOCATION	Atlanta
ACTUAL MACHINE COST	$56,316,780.00
ACTUAL LABOR COST	$108,664,441.00
NUMBER OF MACHINES	31
BUDGET OVERHEAD COST	$62,503,811.52
BUDGET TOTAL COST	$478,319,728.80
BUDGET VS ACTUAL COST MARGIN	$254,396,264.47
BUDGET VS ACTUAL % DIFFERENCE	53.19 %

- Which location is seen to have higher forecast amount in comparison to the actual amount on the basis of time period?
  - Of all the locations, **Toronto** has the highest forecast amount compared with the actual for a difference of **$11,039,328.65** or a **9.52%** miss.
  - To get these values, use the *Actual vs. Forecast Revenue Comparison by Location & Year* chart on Dashboard 3 and Drill Up to the highest Level. Then mouse over the largest circles to figure out which one has the greatest difference as seen in the Tooltips

1. 

2. 

ALL	All
LOCATION	Toronto
ACTUAL AMOUNT	$104,905,416.35
FORECAST AMOUNT ANNUAL	$115,944,745.00
FORECAST VS ACTUAL % DIFFERENCE	9.52 %
FORECAST VS ACTUAL AMOUNT	$11,039,328.65

Appendix A – ETL Scripts

SalesAgent

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_SALES_AGENT_D = CAPMOD6{[Name="W_SALES_AGENT_D"]}[Data],
    #"Removed Columns" = Table.RemoveColumns(W_SALES_AGENT_D,{"CAPMOD6.W_INVOICELINE_F",
"CAPMOD6.W_JOB_F", "CAPMOD6.W_LEAD_F"}),
    #"Renamed Columns" = Table.RenameColumns(#"Removed Columns",{{"COUNTRY",
"SALES_AGENT_COUNTRY"}}),
    #"Changed Type" = Table.TransformColumnTypes(#"Renamed Columns",{{"SALES_AGENT_ID",
Int64.Type}}),
    #"Renamed Columns1" = Table.RenameColumns(#"Changed Type",{{"SALES_AGENT_NAME", "Agent"},
{"SALES_AGENT_STATE", "Agent State"}, {"SALES_AGENT_COUNTRY", "Agent Country"}})
in
    #"Renamed Columns1"
```

Customer

I added in a custom column to separate out the customer groups. When profiling the data, I noticed that many of the individual customers were different locations for the same parent customer.

```
let
    Source = Oracle.Database("XE", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_CUSTOMER_D1 = CAPMOD6{[Name="W_CUSTOMER_D"]}[Data],
    #"Removed Columns" = Table.RemoveColumns(W_CUSTOMER_D1,{"CAPMOD6.W_CUST_LOCATION_D",
"CAPMOD6.W_INVOICELINE_F", "CAPMOD6.W_JOB_F", "CAPMOD6.W_LEAD_F", "CAPMOD6.W_SUB_JOB_F"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"CUST_KEY", Int64.Type}}),
    #"Reordered Columns" = Table.ReorderColumns(#"Changed Type",{"CUST_KEY", "CUST_NAME", "CITY",
"CUST_STATE", "ZIP", "COUNTRY", "E_MAIL_ADDRESS", "TERMS_CODE", "CREDIT_LIMIT"}),
    #"Replaced Value" = Table.ReplaceValue(#"Reordered Columns","USA","United States of
America",Replacer.ReplaceText,{"COUNTRY"}),
    #"Sorted Rows" = Table.Sort(#"Replaced Value",{{"CUST_NAME", Order.Ascending}}),
    #"Added Custom" = Table.AddColumn(#"Sorted Rows", "Custom", each
Text.Start([CUST_NAME],Text.Length([CUST_NAME])-2)),
    #"Renamed Columns" = Table.RenameColumns(#"Added Custom",{{"Custom", "Customer Group"}}),
    #"Reordered Columns1" = Table.ReorderColumns(#"Renamed Columns",{"CUST_KEY", "Customer
Group", "CUST_NAME", "CITY", "CUST_STATE", "ZIP", "COUNTRY", "E_MAIL_ADDRESS", "TERMS_CODE",
"CREDIT_LIMIT"}),
    #"Trimmed Text" = Table.TransformColumns(#"Reordered Columns1",{{"Customer Group",
Text.Trim}}),
    #"Renamed Columns1" = Table.RenameColumns(#"Trimmed Text",{{"CUST_NAME", "Customer"},
{"CUST_STATE", "State"}, {"ZIP", "Zip"}, {"TERMS_CODE", "Terms Code"}, {"CREDIT_LIMIT", "Credit
Limit"}, {"COUNTRY", "Country"}, {"CITY", "City"}, {"E_MAIL_ADDRESS", "eMail"}})
in
    #"Renamed Columns1"
```

Sales Class

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_SALES_CLASS_D1 = CAPMOD6{[Name="W_SALES_CLASS_D"]}[Data],
    #"Removed Columns" =
Table.RemoveColumns(W_SALES_CLASS_D1,{"CAPMOD6.W_FINANCIAL_SUMMARY_COST_F",
"CAPMOD6.W_FINANCIAL_SUMMARY_SALES_F", "CAPMOD6.W_INVOICELINE_F", "CAPMOD6.W_JOB_F",
"CAPMOD6.W_JOB_SHIPMENT_F", "CAPMOD6.W_LEAD_F", "CAPMOD6.W_SUB_JOB_F"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"SALES_CLASS_ID",
Int64.Type}}),
    #"Renamed Columns" = Table.RenameColumns(#"Changed Type",{{"SALES_CLASS_DESC", "Sales
Class"}, {"BASE_PRICE", "Base Price"}})
in
    #"Renamed Columns"
```

Location

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_LOCATION_D1 = CAPMOD6{[Name="W_LOCATION_D"]}[Data],
    #"Removed Columns" = Table.RemoveColumns(W_LOCATION_D1,{"CAPMOD6.W_FINANCIAL_SUMMARY_COST_F",
"CAPMOD6.W_FINANCIAL_SUMMARY_SALES_F", "CAPMOD6.W_INVOICELINE_F", "CAPMOD6.W_JOB_F",
"CAPMOD6.W_JOB_SHIPMENT_F", "CAPMOD6.W_LEAD_F", "CAPMOD6.W_SUB_JOB_F"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"LOCATION_ID",
Int64.Type}}),
    #"Renamed Columns" = Table.RenameColumns(#"Changed Type",{{"LOCATION_NAME", "Location"}})
in
    #"Renamed Columns"
```

Machine Type

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_MACHINE_TYPE_D = CAPMOD6{[Name="W_MACHINE_TYPE_D"]}[Data],
    #"Removed Columns" =
Table.RemoveColumns(W_MACHINE_TYPE_D,{"CAPMOD6.W_FINANCIAL_SUMMARY_COST_F",
"CAPMOD6.W_SUB_JOB_F"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"RATE_PER_HOUR",
Currency.Type}, {"MACHINE_TYPE_ID", Int64.Type}}),
    #"Renamed Columns" = Table.RenameColumns(#"Changed Type",{{"MACHINE_MODEL", "Model"},
{"MANUFACTURER", "Manufacturer"}, {"NUMBER_OF_MACHINES", "Number of Machines"}, {"RATE_PER_HOUR",
"Rate per Hour"}})
in
    #"Renamed Columns"
```

CustLocation

```
let
    Source = Oracle.Database("XE", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_CUST_LOCATION_D1 = CAPMOD6{[Name="W_CUST_LOCATION_D"]}[Data],
    #"Changed Type" = Table.TransformColumnTypes(W_CUST_LOCATION_D1,{{"CUST_LOC_KEY",
Int64.Type}}),
    #"Removed Columns" = Table.RemoveColumns(#"Changed Type",{"CAPMOD6.W_CUSTOMER_D",
"CAPMOD6.W_JOB_SHIPMENT_F"}),
    #"Reordered Columns" = Table.ReorderColumns(#"Removed Columns",{"CUST_LOC_KEY", "CUST_KEY",
"CITY", "COUNTRY", "E_MAIL_ADDRESS", "CUST_LOCATION_STATE", "ZIP", "CUST_NAME", "CREDIT_LIMIT"}),
    #"Changed Type1" = Table.TransformColumnTypes(#"Reordered Columns",{{"CUST_KEY",
Int64.Type}}),
    #"Reordered Columns1" = Table.ReorderColumns(#"Changed Type1",{"CUST_LOC_KEY", "CUST_KEY",
"CITY", "CUST_LOCATION_STATE", "ZIP", "COUNTRY", "E_MAIL_ADDRESS", "CUST_NAME", "CREDIT_LIMIT"}),
    #"Renamed Columns" = Table.RenameColumns(#"Reordered Columns1",{{"CUST_LOCATION_STATE",
"STATE"}}),
    #"Reordered Columns2" = Table.ReorderColumns(#"Renamed Columns",{"CUST_LOC_KEY", "CUST_KEY",
"CUST_NAME", "CITY", "STATE", "ZIP", "COUNTRY", "E_MAIL_ADDRESS", "CREDIT_LIMIT"}),
    #"Renamed Columns1" = Table.RenameColumns(#"Reordered Columns2",{{"CITY", "City"},
{"COUNTRY", "Country"}, {"CREDIT_LIMIT", "Credit Limit"}, {"CUST_NAME", "Customer"},
{"E_MAIL_ADDRESS", "eMail"}, {"STATE", "State"}, {"ZIP", "Zip"}})
in
    #"Renamed Columns1"
```

Date

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_TIME_D1 = CAPMOD6{[Name="W_TIME_D"]}[Data],
    #"Changed Type1" = Table.TransformColumnTypes(W_TIME_D1,{{"TIME_ID", type text}}),
    #"Removed Columns" = Table.RemoveColumns(#"Changed
Type1",{"CAPMOD6.W_FINANCIAL_SUMMARY_COST_F(TIME_ID)",
"CAPMOD6.W_FINANCIAL_SUMMARY_COST_F(TIME_ID) 2", "CAPMOD6.W_FINANCIAL_SUMMARY_SALES_F(TIME_ID)",
"CAPMOD6.W_FINANCIAL_SUMMARY_SALES_F(TIME_ID) 2", "CAPMOD6.W_INVOICELINE_F(TIME_ID)",
"CAPMOD6.W_INVOICELINE_F(TIME_ID) 2", "CAPMOD6.W_JOB_F(TIME_ID)", "CAPMOD6.W_JOB_F(TIME_ID) 2",
"CAPMOD6.W_JOB_F(TIME_ID) 3", "CAPMOD6.W_JOB_SHIPMENT_F(TIME_ID)",
"CAPMOD6.W_JOB_SHIPMENT_F(TIME_ID) 2", "CAPMOD6.W_LEAD_F", "CAPMOD6.W_SUB_JOB_F(TIME_ID)",
"CAPMOD6.W_SUB_JOB_F(TIME_ID) 2"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"TIME_ID", type date},
{"TIME_YEAR", Int64.Type}, {"TIME_QUARTER", Int64.Type}, {"TIME_MONTH", Int64.Type}, {"TIME_DAY",
Int64.Type}, {"TIME_WEEK", Int64.Type}})
in
    #"Changed Type"
```

Leads

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_LEAD_F1 = CAPMOD6{[Name="W_LEAD_F"]}[Data],
    #"Removed Columns" = Table.RemoveColumns(W_LEAD_F1,{"CAPMOD6.W_CUSTOMER_D",
"CAPMOD6.W_LOCATION_D", "CAPMOD6.W_SALES_AGENT_D", "CAPMOD6.W_SALES_CLASS_D",
"CAPMOD6.W_TIME_D"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"CREATED_DATE", type
text}}),
    #"Reordered Columns" = Table.ReorderColumns(#"Changed Type",{"LEAD_ID", "CUST_ID",
"LOCATION_ID", "SALES_CLASS_ID", "SALES_AGENT_ID", "JOB_ID", "CREATED_DATE", "LEAD_SUCCESS",
"QUOTE_QTY", "QUOTE_PRICE"}),
    #"Changed Type1" = Table.TransformColumnTypes(#"Reordered Columns",{{"CREATED_DATE", type
date}, {"JOB_ID", Int64.Type}, {"SALES_AGENT_ID", Int64.Type}, {"SALES_CLASS_ID", Int64.Type},
{"LOCATION_ID", Int64.Type}, {"CUST_ID", Int64.Type}, {"LEAD_ID", Int64.Type}}),
    #"Renamed Columns" = Table.RenameColumns(#"Changed Type1",{{"LEAD_SUCCESS", "Lead Success"},
{"QUOTE_PRICE", "Quote Price"}, {"QUOTE_QTY", "Quote Qty"}, {"CREATED_DATE", "Created Date"}})
in
    #"Renamed Columns"
```

Jobs

```
let
    Source = Oracle.Database("XE", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_JOB_F1 = CAPMOD6{[Name="W_JOB_F"]}[Data],
    #"Removed Columns" = Table.RemoveColumns(W_JOB_F1,{"CAPMOD6.W_CUSTOMER_D",
"CAPMOD6.W_LOCATION_D", "CAPMOD6.W_SALES_AGENT_D", "CAPMOD6.W_SALES_CLASS_D",
"CAPMOD6.W_TIME_D(CONTRACT_DATE)", "CAPMOD6.W_TIME_D(DATE_PROMISED)",
"CAPMOD6.W_TIME_D(DATE_SHIP_BY)"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"CONTRACT_DATE", type
text}, {"DATE_PROMISED", type text}, {"DATE_SHIP_BY", type text}}),
    #"Reordered Columns" = Table.ReorderColumns(#"Changed Type",{"JOB_ID", "SALES_AGENT_ID",
"SALES_CLASS_ID", "LOCATION_ID", "CUST_ID_ORDERED_BY", "CONTRACT_DATE", "DATE_PROMISED",
"DATE_SHIP_BY", "NUMBER_OF_SUBJOBS", "UNIT_PRICE", "QUANTITY_ORDERED", "QUOTE_QTY"}),
    #"Changed Type1" = Table.TransformColumnTypes(#"Reordered Columns",{{"CONTRACT_DATE", type
date}, {"DATE_PROMISED", type date}, {"DATE_SHIP_BY", type date}, {"CUST_ID_ORDERED_BY",
Int64.Type}, {"LOCATION_ID", Int64.Type}, {"SALES_CLASS_ID", Int64.Type}, {"SALES_AGENT_ID",
Int64.Type}, {"JOB_ID", Int64.Type}}),
    #"Renamed Columns" = Table.RenameColumns(#"Changed Type1",{{"CONTRACT_DATE", "Contract
Date"}, {"DATE_PROMISED", "Promised Date"}, {"DATE_SHIP_BY", "Ship By Date"}, {"UNIT_PRICE",
"Unit Price"}, {"QUOTE_QTY", "Quote Qty"}, {"QUANTITY_ORDERED", "Order Qty"},
{"NUMBER_OF_SUBJOBS", "Number of SubJobs"}})
in
    #"Renamed Columns"
```

SubJobs

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_SUB_JOB_F1 = CAPMOD6{[Name="W_SUB_JOB_F"]}[Data],
    #"Removed Columns" = Table.RemoveColumns(W_SUB_JOB_F1,{"CAPMOD6.W_CUSTOMER_D",
"CAPMOD6.W_LOCATION_D", "CAPMOD6.W_MACHINE_TYPE_D", "CAPMOD6.W_SALES_CLASS_D",
"CAPMOD6.W_TIME_D(DATE_PROD_BEGIN)", "CAPMOD6.W_TIME_D(DATE_PROD_END)"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"DATE_PROD_BEGIN", type
text}, {"DATE_PROD_END", type text}}),
    #"Reordered Columns" = Table.ReorderColumns(#"Changed Type",{"SUB_JOB_ID", "JOB_ID",
"LOCATION_ID", "CUST_ID_ORDERED_BY", "SALES_CLASS_ID", "MACHINE_TYPE_ID", "DATE_PROD_BEGIN",
"DATE_PROD_END", "COST_LABOR", "COST_MATERIAL", "COST_OVERHEAD", "MACHINE_HOURS",
"QUANTITY_PRODUCED", "SUB_JOB_AMOUNT"}),
    #"Changed Type1" = Table.TransformColumnTypes(#"Reordered Columns",{{"DATE_PROD_BEGIN", type
date}, {"DATE_PROD_END", type date}, {"COST_LABOR", Currency.Type}, {"COST_MATERIAL",
Currency.Type}, {"COST_OVERHEAD", Currency.Type}, {"SUB_JOB_AMOUNT", Currency.Type},
{"MACHINE_TYPE_ID", Int64.Type}, {"SALES_CLASS_ID", Int64.Type}, {"CUST_ID_ORDERED_BY",
Int64.Type}, {"LOCATION_ID", Int64.Type}, {"JOB_ID", Int64.Type}, {"SUB_JOB_ID", Int64.Type}}),
    #"Renamed Columns" = Table.RenameColumns(#"Changed Type1",{{"COST_LABOR", "Cost Labor"},
{"COST_MATERIAL", "Cost Material"}, {"COST_OVERHEAD", "Cost Overhead"}, {"DATE_PROD_BEGIN", "Date
Production Begin"}, {"DATE_PROD_END", " Date Production End"}, {"MACHINE_HOURS", "Machine
Hours"}, {"QUANTITY_PRODUCED", "Produced Qty"}, {"SUB_JOB_AMOUNT", "SubJob Amount"}})
in
    #"Renamed Columns"
```

Shipments

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_JOB_SHIPMENT_F1 = CAPMOD6{[Name="W_JOB_SHIPMENT_F"]}[Data],
    #"Removed Columns" = Table.RemoveColumns(W_JOB_SHIPMENT_F1,{"CAPMOD6.W_CUST_LOCATION_D",
"CAPMOD6.W_LOCATION_D", "CAPMOD6.W_SALES_CLASS_D", "CAPMOD6.W_TIME_D(ACTUAL_SHIP_DATE)",
"CAPMOD6.W_TIME_D(REQUESTED_SHIP_DATE)"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"REQUESTED_SHIP_DATE", type
text}, {"ACTUAL_SHIP_DATE", type text}}),
    #"Reordered Columns" = Table.ReorderColumns(#"Changed Type",{"JOB_SHIPMENT_ID", "INVOICE_ID",
"CUST_ID_SHIP_TO", "LOCATION_ID", "SALES_CLASS_ID", "SUB_JOB_ID", "ACTUAL_SHIP_DATE",
"REQUESTED_SHIP_DATE", "ACTUAL_QUANTITY", "REQUESTED_QUANTITY", "BOXES", "QUANTITY_PER_BOX",
"QUANTITY_PER_PARTIAL_BOX", "SHIPPED_AMOUNT"}),
    #"Changed Type1" = Table.TransformColumnTypes(#"Reordered Columns",{{"REQUESTED_SHIP_DATE",
type date}, {"ACTUAL_SHIP_DATE", type date}, {"SUB_JOB_ID", Int64.Type}, {"SALES_CLASS_ID",
Int64.Type}, {"LOCATION_ID", Int64.Type}, {"CUST_ID_SHIP_TO", Int64.Type}, {"INVOICE_ID",
Int64.Type}, {"JOB_SHIPMENT_ID", Int64.Type}, {"SHIPPED_AMOUNT", Currency.Type}}),
    #"Renamed Columns" = Table.RenameColumns(#"Changed Type1",{{"ACTUAL_QUANTITY", "Actual Qty"},
{"ACTUAL_SHIP_DATE", "Actual Ship Date"}, {"BOXES", "Number of Boxes"}, {"QUANTITY_PER_BOX", "Qty
per Box"}, {"QUANTITY_PER_PARTIAL_BOX", "Qty per Partial Box"}, {"REQUESTED_QUANTITY", "Requested
Qty"}, {"REQUESTED_SHIP_DATE", "Requested Ship Date"}, {"SHIPPED_AMOUNT", "Shipped Amount"}})
in
    #"Renamed Columns"
```

InvoiceLines

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_INVOICELINE_F1 = CAPMOD6{[Name="W_INVOICELINE_F"]}[Data],
    #"Removed Columns" = Table.RemoveColumns(W_INVOICELINE_F1,{"CAPMOD6.W_CUSTOMER_D",
"CAPMOD6.W_LOCATION_D", "CAPMOD6.W_SALES_AGENT_D", "CAPMOD6.W_SALES_CLASS_D",
"CAPMOD6.W_TIME_D(INVOICE_DUE_DATE)", "CAPMOD6.W_TIME_D(INVOICE_SENT_DATE)"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"INVOICE_DUE_DATE", type
text}, {"INVOICE_SENT_DATE", type text}}),
    #"Reordered Columns" = Table.ReorderColumns(#"Changed Type",{"INVOICE_ID", "SALES_CLASS_ID",
"CUST_KEY", "LOCATION_ID", "SALES_AGENT_ID", "INVOICE_DUE_DATE", "INVOICE_SENT_DATE",
"INVOICE_QUANTITY", "INVOICE_AMOUNT", "QUANTITY_SHIPPED"}),
    #"Changed Type1" = Table.TransformColumnTypes(#"Reordered Columns",{{"INVOICE_ID",
Int64.Type}, {"SALES_CLASS_ID", Int64.Type}, {"CUST_KEY", Int64.Type}, {"LOCATION_ID",
Int64.Type}, {"SALES_AGENT_ID", Int64.Type}, {"INVOICE_DUE_DATE", type date},
{"INVOICE_SENT_DATE", type date}, {"INVOICE_AMOUNT", Currency.Type}}),
    #"Renamed Columns" = Table.RenameColumns(#"Changed Type1",{{"INVOICE_AMOUNT", "Revenue"},
{"QUANTITY_SHIPPED", "Shipped Qty"}, {"INVOICE_QUANTITY", "Invoice Qty"}, {"INVOICE_DUE_DATE",
"Invoice Due Date"}, {"INVOICE_SENT_DATE", "Invoice Sent Date"}})
in
    #"Renamed Columns"
```

Shipments Completed

```
let
```

```
    Source = Shipment,
    #"Removed Columns" = Table.RemoveColumns(Source,{"JOB_SHIPMENT_ID", "INVOICE_ID",
"CUST_ID_SHIP_TO", "LOCATION_ID", "SALES_CLASS_ID", "Actual Ship Date", "Requested Ship Date",
"Number of Boxes", "Qty per Box", "Qty per Partial Box", "Shipped Amount"}),
    #"Merged Queries" = Table.NestedJoin(#"Removed
Columns",{"SUB_JOB_ID"},SubJob,{"SUB_JOB_ID"},"NewColumn",JoinKind.LeftOuter),
    #"Expanded NewColumn" = Table.ExpandTableColumn(#"Merged Queries", "NewColumn", {"JOB_ID"},
{"JOB_ID"}),
    #"Reordered Columns" = Table.ReorderColumns(#"Expanded NewColumn",{"JOB_ID", "SUB_JOB_ID",
"Actual Qty", "Requested Qty"}),
    #"Removed Columns1" = Table.RemoveColumns(#"Reordered Columns",{"SUB_JOB_ID"}),
    #"Grouped Rows" = Table.Group(#"Removed Columns1", {"JOB_ID"}, {{"Actual Qty", each
List.Sum([Actual Qty]), type number}, {"Requested Qty", each List.Sum([Requested Qty]), type
number}}),
    #"Added Custom" = Table.AddColumn(#"Grouped Rows", "Fully Shipped ", each if [Requested
Qty]=[Actual Qty] then true else false)
in
    #"Added Custom"
```

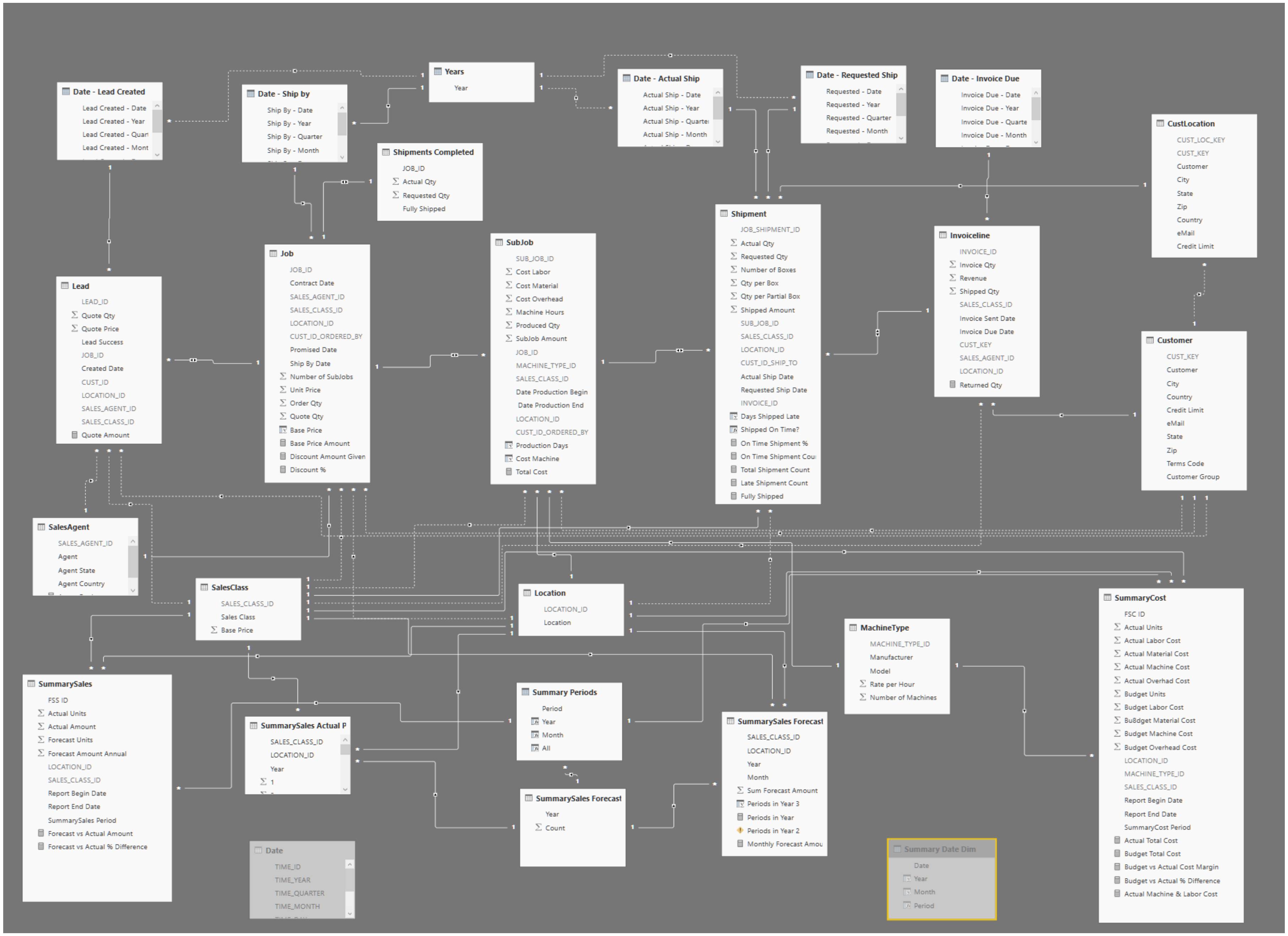
SummaryCost

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_FINANCIAL_SUMMARY_COST_F1 = CAPMOD6{[Name="W_FINANCIAL_SUMMARY_COST_F"]}[Data],
    #"Removed Columns" = Table.RemoveColumns(W_FINANCIAL_SUMMARY_COST_F1,{"CAPMOD6.W_LOCATION_D",
"CAPMOD6.W_MACHINE_TYPE_D", "CAPMOD6.W_SALES_CLASS_D", "CAPMOD6.W_TIME_D(REPORT_BEGIN_DATE_ID)",
"CAPMOD6.W_TIME_D(REPORT_END_DATE_ID)"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"REPORT_END_DATE_ID", type
text}, {"REPORT_BEGIN_DATE_ID", type text}}),
    #"Reordered Columns" = Table.ReorderColumns(#"Changed Type",{"FINANCIAL_SUMMARY_COST_ID",
"SALES_CLASS_ID", "MACHINE_TYPE_ID", "LOCATION_ID", "REPORT_BEGIN_DATE_ID", "REPORT_END_DATE_ID",
"ACTUAL_UNITS", "ACTUAL_LABOR_COST", "ACTUAL_MATERIAL_COST", "ACTUAL_MACHINE_COST",
"ACTUAL_OVERHEAD_COST", "BUDGET_UNITS", "BUDGET_LABOR_COST", "BUDGET_MATERIAL_COST",
"BUDGET_MACHINE_COST", "BUDGET_OVERHEAD_COST"}),
    #"Changed Type1" = Table.TransformColumnTypes(#"Reordered
Columns",{{"FINANCIAL_SUMMARY_COST_ID", Int64.Type}, {"SALES_CLASS_ID", Int64.Type},
{"MACHINE_TYPE_ID", Int64.Type}, {"LOCATION_ID", Int64.Type}, {"REPORT_BEGIN_DATE_ID", type
date}, {"REPORT_END_DATE_ID", type date}, {"ACTUAL_LABOR_COST", Currency.Type},
{"ACTUAL_MATERIAL_COST", Currency.Type}, {"ACTUAL_MACHINE_COST", Currency.Type},
{"ACTUAL_OVERHEAD_COST", Currency.Type}, {"BUDGET_LABOR_COST", Currency.Type},
{"BUDGET_MATERIAL_COST", Currency.Type}, {"BUDGET_MACHINE_COST", Currency.Type},
{"BUDGET_OVERHEAD_COST", Currency.Type}}),
    #"Renamed Columns" = Table.RenameColumns(#"Changed Type1",{{"ACTUAL_LABOR_COST", "Actual
Labor Cost"}, {"ACTUAL_MACHINE_COST", "Actual Machine Cost"}, {"ACTUAL_MATERIAL_COST", "Actual
Material Cost"}, {"ACTUAL_OVERHEAD_COST", "Actual Overhad Cost"}, {"ACTUAL_UNITS", "Actual
Units"}, {"BUDGET_LABOR_COST", "Budget Labor Cost"}, {"BUDGET_MACHINE_COST", "Budget Machine
Cost"}, {"BUDGET_MATERIAL_COST", "Bu8dget Material Cost"}, {"BUDGET_OVERHEAD_COST", "Budget
Overhead Cost"}, {"BUDGET_UNITS", "Budget Units"}, {"FINANCIAL_SUMMARY_COST_ID", "FSC ID"},
{"REPORT_BEGIN_DATE_ID", "Report Begin Date"}, {"REPORT_END_DATE_ID", "Report End Date"}}),
    #"Added Custom" = Table.AddColumn(#"Renamed Columns", "SummaryCost Period", each
Text.Start(Text.From([FSC ID]),4) & "-" & Text.End(Text.Start(Text.From([FSC ID]),6),2)),
    #"Changed Type2" = Table.TransformColumnTypes(#"Added Custom",{{"SummaryCost Period", type
text}})
in
    #"Changed Type2"
```

SummarySales

```
let
    Source = Oracle.Database("xe", [HierarchicalNavigation=true]),
    CAPMOD6 = Source{[Schema="CAPMOD6"]}[Data],
    W_FINANCIAL_SUMMARY_SALES_F1 = CAPMOD6{[Name="W_FINANCIAL_SUMMARY_SALES_F"]}[Data],
    #"Removed Columns" =
Table.RemoveColumns(W_FINANCIAL_SUMMARY_SALES_F1,{"CAPMOD6.W_LOCATION_D",
"CAPMOD6.W_SALES_CLASS_D", "CAPMOD6.W_TIME_D(REPORT_BEGIN_DATE_ID)",
"CAPMOD6.W_TIME_D(REPORT_END_DATE_ID)"}),
    #"Changed Type" = Table.TransformColumnTypes(#"Removed Columns",{{"REPORT_BEGIN_DATE_ID",
type text}, {"REPORT_END_DATE_ID", type text}}),
    #"Reordered Columns" = Table.ReorderColumns(#"Changed Type",{"FINANCIAL_SUMMARY_SALES_ID",
"SALES_CLASS_ID", "LOCATION_ID", "REPORT_END_DATE_ID", "REPORT_BEGIN_DATE_ID", "ACTUAL_UNITS",
"ACTUAL_AMOUNT", "FORCAST_UNIT", "FORCAST_AMOUNT"}),
    #"Changed Type1" = Table.TransformColumnTypes(#"Reordered Columns",{{"REPORT_BEGIN_DATE_ID",
type date}, {"REPORT_END_DATE_ID", type date}, {"FORCAST_AMOUNT", Currency.Type},
{"ACTUAL_AMOUNT", Currency.Type}, {"SALES_CLASS_ID", Int64.Type}, {"LOCATION_ID", Int64.Type}}),
    #"Reordered Columns1" = Table.ReorderColumns(#"Changed Type1",{"FINANCIAL_SUMMARY_SALES_ID",
"SALES_CLASS_ID", "LOCATION_ID", "REPORT_END_DATE_ID", "REPORT_BEGIN_DATE_ID", "ACTUAL_UNITS",
"ACTUAL_AMOUNT", "FORCAST_UNIT", "FORCAST_AMOUNT"}),
    #"Renamed Columns" = Table.RenameColumns(#"Reordered Columns1",{{"ACTUAL_AMOUNT", "Actual
Amount"}, {"ACTUAL_UNITS", "Actual Units"}, {"FINANCIAL_SUMMARY_SALES_ID", "FSS ID"},
{"FORCAST_AMOUNT", "Forecast Amount"}, {"FORCAST_UNIT", "Forecast Units"},
{"REPORT_BEGIN_DATE_ID", "Report Begin Date"}, {"REPORT_END_DATE_ID", "Report End Date"}}),
    #"Added Custom" = Table.AddColumn(#"Renamed Columns", "SummarySales Period", each
Text.Start(Text.From([FSS ID]),4) & "-" & Text.End(Text.Start(Text.From([FSS ID]),6),2)),
    #"Changed Type2" = Table.TransformColumnTypes(#"Added Custom",{{"SummarySales Period", type
text}}),
    #"Renamed Columns1" = Table.RenameColumns(#"Changed Type2",{{"Forecast Amount", "Forecast
Amount Annual"}})
in
    #"Renamed Columns1"
```

Appendix B – Data Model



Appendix C – DAX Equations

Custom Tables

Date Tables

I used DAX to create multiple date dimension tables which are copies of the W_TIME_D table in the DW. I first imported the table from the DW using PowerQuery as seen in Appendix A. I then used DAX to create a duplicate table in the data model without duplicating the import. I did this for the following dates: Actual Ship, Invoice Due, Lead Created, Requested Ship and Ship By date fields. I didn't do this for all dates as not all dates in the model were needed for time-intelligence functions.

```
Date – {Date Name} = 'Date'
```

Summary Date Dim

I wanted a separate date dimension for the Summary tables with only the date periods applicable to both tables. Use the DAX Calendar function with the first cost period and the last sales period as bounds.

```
Summary Date Dim = CALENDAR(MIN(SummaryCost[Report Begin Date].[Date]),MAX(SummarySales[Report End Date]))
```

Custom Columns

I added multiple columns to different tables in the model for use in either visualizations, filters or measures.

Date

```
Year - Week = 'Date'[TIME_YEAR]&"-"&'Date'[TIME_WEEK]
```

Job

```
Base Price = LOOKUPVALUE(SalesClass[Base Price],SalesClass[SALES_CLASS_ID],[SALES_CLASS_ID])
```

Shipment

```
Days Shipped Late = DATEDIFF(Shipment[Requested Ship Date],Shipment[Actual Ship Date],DAY)

Shipped On Time? = IF(Shipment[Days Shipped Late]=0,TRUE,FALSE)
```

SubJob

```
Cost Machine = SubJob[Machine Hours]*LOOKUPVALUE(MachineType[Rate per Hour], MachineType[MACHINE_TYPE_ID],
SubJob[MACHINE_TYPE_ID])

Production Days = DATEDIFF(SubJob[Date Production Begin],SubJob[ Date Production End],DAY)
```

Summary Date Dim

```
Month = MONTH('Summary Date Dim'[Date])

Year = YEAR('Summary Date Dim'[Date])

Period = FORMAT('Summary Date Dim'[Date],"yyyy-mm")
```

SummaryPeriods

```
Month = RIGHT('Summary Periods'[Period],2)

Year = LEFT('Summary Periods'[Period],4)

All = "All"
```

Custom Measures

InvoiceLine

```
Returned Qty = CALCULATE(SUM(Invoiceline[Shipped Qty])-SUM(Invoiceline[Invoice Qty]))
```

Job

```
Base Price Amount = SUMX(Job,Job[Base Price]*Job[Quote Qty])

Discount Amount Given = [Base Price Amount]-[Quote Amount]

Discount % = IF(ISBLANK([Discount Amount Given]),0,[Discount Amount Given]/[Base Price Amount])
```

Lead

```
Quote Amount = SUMX('Lead','Lead'[Quote Qty]*'Lead'[Quote Price])
```

SalesAgent

```
Agent Rank =
IF(HASONEVALUE(SalesAgent[Agent]),RANKX(ALLSELECTED(SalesAgent[Agent]),CALCULATE(SUM(Invoiceline[Revenue]))))
```

Shipment

```
Fully Shipped = IF(SUM(Shipment[Requested Qty])=SUM(Shipment[Actual Qty]),TRUE(),FALSE())

Late Shipment Count = CALCULATE(COUNT(Shipment[JOB_SHIPMENT_ID]),Shipment[Shipped On Time?]=FALSE())

On Time Shipment % = CALCULATE(COUNT(Shipment[JOB_SHIPMENT_ID]),Shipment[Shipped On Time?]=TRUE())/[Total
Shipment Count]

On Time Shipment Count = CALCULATE(COUNT(Shipment[JOB_SHIPMENT_ID]),Shipment[Shipped On Time?]=TRUE())

Total Shipment Count = CALCULATE(COUNT(Shipment[JOB_SHIPMENT_ID]),ALLEXCEPT('Date - Ship by','Date - Ship
by'[Ship By - Year],'Date - Ship by'[Ship By - Quarter],'Date - Ship by'[Ship By - Month],'Date - Ship by'[Ship
By - Week],'Date - Ship by'[Ship By - Day]))
```

SubJob

```
Total Cost = CALCULATE(SUM(SubJob[Cost Labor])+ SUM(SubJob[Cost Material])+ SUM(SubJob[Cost Overhead])+
SUM(SubJob[Cost Machine]))
```

SummaryCosts

```
Actual Machine & Labor Cost = SUM(SummaryCost[Actual Labor Cost])+SUM(SummaryCost[Actual Machine Cost])

Actual Total Cost = CALCULATE(SUM(SummaryCost[Actual Labor Cost])+SUM(SummaryCost[Actual Machine
Cost])+SUM(SummaryCost[Actual Material Cost])+SUM(SummaryCost[Actual Overhad Cost]))

Budget Total Cost = CALCULATE(SUM(SummaryCost[Budget Labor Cost])+SUM(SummaryCost[Budget Machine
Cost])+SUM(SummaryCost[Bu8dget Material Cost])+SUM(SummaryCost[Budget Overhead Cost]))

Budget vs Actual % Difference = [Budget vs Actual Cost Margin]/[Budget Total Cost]

Budget vs Actual Cost Margin = [Budget Total Cost]-[Actual Total Cost]
```

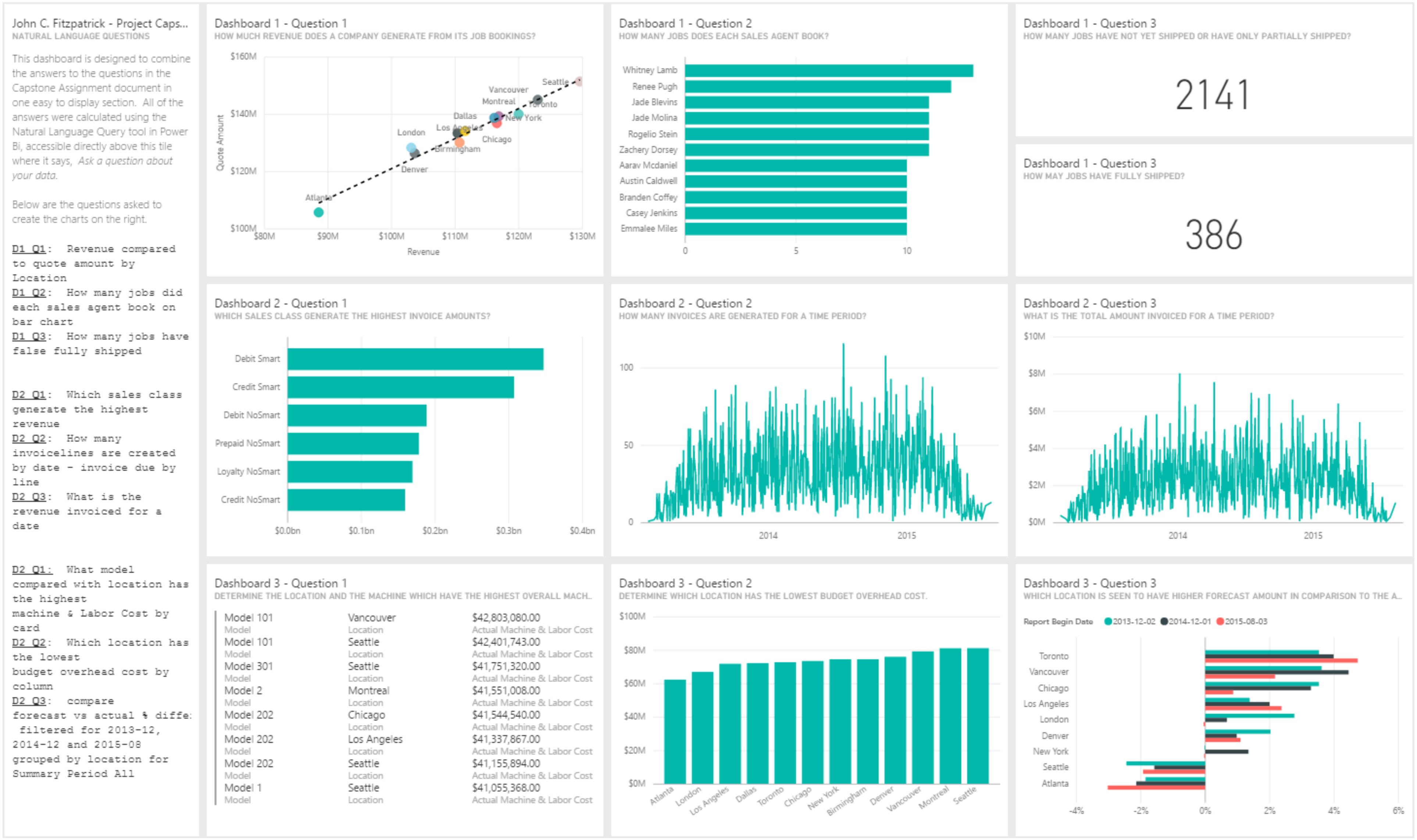
SummarySales

```
Forecast vs Actual % Difference = [Forecast vs Actual Amount]/SUM(SummarySales[Forecast Amount Annual])

Forecast vs Actual Amount = SUM(SummarySales[Forecast Amount Annual])-SUM(SummarySales[Actual Amount])
```

Appendix D – Power Bi Dashboard Using Natural Language Query

The live dashboard is available at https://goo.gl/IsrfAk. Please send your work or school Microsoft account address to john@jandtdesign.com to request access.



John C. Fitzpatrick - Project Caps...
NATURAL LANGUAGE QUESTIONS

This dashboard is designed to combine the answers to the questions in the Capstone Assignment document in one easy to display section. All of the answers were calculated using the Natural Language Query tool in Power Bi, accessible directly above this tile where it says, *Ask a question about your data.*

Below are the questions asked to create the charts on the right.

```
D1 Q1:  Revenue compared
to quote amount by
Location
D1 Q2:  How many jobs did
each sales agent book on
bar chart
D1 Q3:  How many jobs have
false fully shipped


D2 Q1:  Which sales class
generate the highest
revenue
D2 Q2:  How many
invoicelines are created
by date - invoice due by
line
D2 Q3:  What is the
revenue invoiced for a
date


D2 Q1:  What model
compared with location has
the highest
machine & Labor Cost by
card
D2 Q2:  Which location has
the lowest
budget overhead cost by
column
D2 Q3:  compare
forecast vs actual % diffe:
 filtered for 2013-12,
2014-12 and 2015-08
grouped by location for
Summary Period All
```

Dashboard 1 - Question 1
HOW MUCH REVENUE DOES A COMPANY GENERATE FROM ITS JOB BOOKINGS?

Dashboard 1 - Question 2
HOW MANY JOBS DOES EACH SALES AGENT BOOK?

Dashboard 1 - Question 3
HOW MANY JOBS HAVE NOT YET SHIPPED OR HAVE ONLY PARTIALLY SHIPPED?

2141

Dashboard 1 - Question 3
HOW MAY JOBS HAVE FULLY SHIPPED?

386

Dashboard 2 - Question 1
WHICH SALES CLASS GENERATE THE HIGHEST INVOICE AMOUNTS?

Dashboard 2 - Question 2
HOW MANY INVOICES ARE GENERATED FOR A TIME PERIOD?

Dashboard 2 - Question 3
WHAT IS THE TOTAL AMOUNT INVOICED FOR A TIME PERIOD?

Dashboard 3 - Question 1
DETERMINE THE LOCATION AND THE MACHINE WHICH HAVE THE HIGHEST OVERALL MACH...

Model 101 Model	Vancouver Location	$42,803,080.00 Actual Machine & Labor Cost
Model 101 Model	Seattle Location	$42,401,743.00 Actual Machine & Labor Cost
Model 301 Model	Seattle Location	$41,751,320.00 Actual Machine & Labor Cost
Model 2 Model	Montreal Location	$41,551,008.00 Actual Machine & Labor Cost
Model 202 Model	Chicago Location	$41,544,540.00 Actual Machine & Labor Cost
Model 202 Model	Los Angeles Location	$41,337,867.00 Actual Machine & Labor Cost
Model 202 Model	Seattle Location	$41,155,894.00 Actual Machine & Labor Cost
Model 1 Model	Seattle Location	$41,055,368.00 Actual Machine & Labor Cost

Dashboard 3 - Question 2
DETERMINE WHICH LOCATION HAS THE LOWEST BUDGET OVERHEAD COST.

Dashboard 3 - Question 3
WHICH LOCATION IS SEEN TO HAVE HIGHER FORECAST AMOUNT IN COMPARISON TO THE A...