Table of Contents

1) Grain or Granularity

- Size or level of detail.
    - i. What is the grain or granularity of the sales table?
        1. Is the grain of the sales table at the invoice line item product level?
        2. Is the grain of the sales table at the Total Invoice Level?
    - ii. Is the granularity of the 1) helper column or 2) iteration of the DAX Formula or 3) of the report at the day, month, year level?
- More or Less Granularity?
    - i. The more granular, the smaller the size, like a grain of sand is small.
    - ii. The less granularity, the more aggregated the number is (the bigger the number is), like moving from adding total sales from daily sales to monthly sales to year sales and so on.
- Granularity will be important when designing a data model and creating formulas in both Power Query and the Data Model.
- Example of Grain or Granularity:

Grain or Granularity = Size or level of detail.

For Criteria in Report ==>> Granularity is decreasing

Grain = Invoice Line or Product Level
The **more granular**, the smaller the size.

Grain = Invoice Level
The **less granularity**, the more aggregated the number is (bigger number)

| Day | Month | Year |
| Grain | Grain | Grain |

Date	Invoice #	Units	SalesRepKey	ProductKey	Sales
10/20/17	27002	44	1	4	$620.95
10/20/17	27002	37	1	3	$484.31
10/20/17	27003	38	2	5	$376.69
10/20/17	27003	82	2	1	$1,141.96
10/20/17	27004	56	3	2	$725.45
10/20/17	27004	25	3	2	$222.67
10/20/17	27005	130	4	3	$1,038.10
10/21/17	27006	10	3	1	$154.41
10/24/17	27007	82	5	5	$1,236.00
10/24/17	27008	53	3	4	$684.48
10/24/17	27009	12	5	6	$127.10
10/24/17	27010	22	4	5	$269.52
10/25/17	27011	82	3	7	$739.69
10/25/17	27012	172	5	7	$1,201.01
10/25/17	27012	78	5	6	$546.06
10/25/17	27012	12	5	1	$162.45

Date	Invoice #	SalesRepKey	Sales
10/20/17	27002	1	$1,105.26
10/20/17	27003	2	$1,518.65
10/20/17	27004	3	$948.12
10/20/17	27005	4	$1,038.10
10/21/17	27006	3	$154.41
10/24/17	27007	5	$1,236.00
10/24/17	27008	3	$684.48
10/24/17	27009	5	$127.10
10/24/17	27010	4	$269.52
10/25/17	27011	3	$739.69
10/25/17	27012	5	$1,909.52

Date	Day	Month	MonthNumber	Year
10/1/17	1	Oct	10	2017
10/2/17	2	Oct	10	2017
10/3/17	3	Oct	10	2017
10/4/17	4	Oct	10	2017
10/5/17	5	Oct	10	2017
10/6/17	6	Oct	10	2017
10/7/17	7	Oct	10	2017
10/8/17	8	Oct	10	2017
10/9/17	9	Oct	10	2017
10/10/17	10	Oct	10	2017
10/11/17	11	Oct	10	2017
10/12/17	12	Oct	10	2017
10/13/17	13	Oct	10	2017
10/14/17	14	Oct	10	2017
10/15/17	15	Oct	10	2017
10/16/17	16	Oct	10	2017
10/17/17	17	Oct	10	2017

2) Keys
- Primary Key
    - i. Unique Identifier for each row in table so there are no duplicates (like no duplicate Product Names or Sales Reps).
    - ii. Primary Keys are the first column in Dimension / Lookup Tables and are used on the one-side of a one-to-many-relationship.
    - iii. If you are coming from the Excel side, you already know what a Primary Key is because all lookup tables are required to have a Unique List in the first column, which is, of course, a Primary Key.
- Foreign Key
    - i. A Column that is allowed duplicates and is used on the many-side of a one-to-many-relationship.
- Primary & Foreign Keys are used together with:
    - i. VLOOKUP in Excel
    - ii. Relationships in the Data Model
    - iii. Merges in Power Query

Foreign Key	Fact	Foreign Key	Foreign Key	Fact
Date	Units	SalesRepKey	ProductKey	Sales
10/20/17	44	1	4	$620.95
10/20/17	37	1	3	$484.31
10/20/17	38	2	5	$376.69
10/20/17	82	2	1	$1,141.96
10/20/17	56	3	2	$725.45
10/20/17	25	3	2	$222.67
10/20/17	130	4	3	$1,038.10
10/21/17	10	3	1	$154.41
10/24/17	82	5	5	$1,236.00
10/24/17	53	3	4	$684.48
10/24/17	12	5	6	$127.10
10/24/17	22	4	5	$269.52
10/25/17	82	3	7	$739.69
10/25/17	172	1	7	$1,201.01
10/25/17	78	4	6	$546.06
10/25/17	12	3	1	$162.45

Primary Key	Attributes For Filtering	Attributes For Filtering
SalesRepKey	Sales Rep	Region
1	Gigi	USA
2	Freddy	Canada
3	Chin	Mexico
4	June	USA
5	Tyrone	Europe

Primary Key	Attributes For Filtering	Values used in Calculations	Values used in Calculations
ProductKey	Product	Price	SCost
1	Fun Fly	9.99	6.49
2	Sunshine	19.99	8.8
3	Fast Catch	23.45	14.54
4	Yanaki	24.95	15.22
5	Carlota	25.99	17.15
6	Crested Beaut	32.99	14.19
7	Quad	43.69	22.72

3) **Fact Table / Transaction Table**
- Table that has:
    i. Numbers we need to summarize (like Sales or Units)
    ii. Foreign Key columns that we will use in relationships (like Date, SalesRepKey, ProductKey)
    iii. After we complete our Data Model, the DAX Measures (Formulas for reports) will usually be listed in the Fact Table.
- "Fact" = measurements of business activities (like amount of sales, or how many units sold, or clicks on web links)
    i. Now we know where the DAX word "Measure" came from!
- Fact Tables are usually large. Dimension Tables are comparatively small.
- Grain of Fact Table
    i. The grain of the Fact Table defines what dimensions you can use to filter the calculation.
        1. If you have a Fact Table with Invoice totals, but no product totals, you can't filter a report by product.
    ii. In general, you want the Fact Table grain to be the smallest size that has not been aggregated and will allow the most dimensionality
        1. More dimension tables = more filtering and varied types of queries.
    iii. The grain of the Fact Table usually comes from the Measured Business Event, like the measurements of a line item sold: sales amount, price, quantity.
    iv. Data at its most granular or atomic level (not aggregated) is the foundation of every Fact Table.
        1. Example: If you sell pallets of plywood (one product) to construction companies and the wood comes from more than one supplier, you might have to break the Product Level Sales Amount Fact Table up into a smaller grain to show sales amounts by supplier.

4) **Dimension Table / Lookup Table / Entity Tables**
- Table that has:
    i. First column is a Primary Key (Unique Identifier) for the Entity (Product, Sales Rep and so on) used in relationship with Fact Table.
    ii. Remaining columns are attributes that we can use as:
        1. Criteria / Filters / Categories / Report Labels for our Reports & Dashboards.
        2. Values we can lookup (like Price).
        3. Helper Columns (like Sort Helper Columns or Intermediate calculations).
- Fact Tables are usually large. Dimension Tables are comparatively small.
    i. An Insurance company might have a large Customer Dimension Table.
- Date Table
    i. Type of Dimension table required by MS Power Tools for DAX Time Intelligence Functions.
    ii. We use Data Tables rather than the Group By Date feature in a PivotTable.
- Synonyms for Filtering columns:
    i. Attribute, Criteria, Filters, Category, Categorical attributes, Constraints, Groupings, Report labels for the reports and analytics.
- When you choose a column as a filter for a report, you are choosing the grain of the calculation. For example: Day, Month or Year columns.

- Example of Fact and Dimension Tables:

Fact Table =
Table that has numbers we need to summarize (like Sales and Units)
and
Foreign Keys that we will use in relationships (like Date, SalesRepKey, ProductKey)
"Fact" = measurements of business activities (like amount of sales or how many units)

Foreign Key	Fact	Foreign Key	Foreign Key	Fact
Date	Units	SalesRepKey	ProductKey	Sales
10/20/17	44	1	4	$620.95
10/20/17	37	1	3	$484.31
10/20/17	38	2	5	$376.69
10/20/17	82	2	1	$1,141.96
10/20/17	56	3	2	$725.45
10/20/17	25	3	2	$222.67
10/20/17	130	4	3	$1,038.10
10/21/17	10	3	1	$154.41
10/24/17	82	5	5	$1,236.00
10/24/17	53	3	4	$684.48
10/24/17	12	5	6	$127.10
10/24/17	22	4	5	$269.52
10/25/17	82	3	7	$739.69
10/25/17	172	1	7	$1,201.01
10/25/17	78	4	6	$546.06
10/25/17	12	3	1	$162.45

Dimension Table / Lookup Table = Tables with:
1) First column is a Primary Key (Unique Identifier) used in relationship with Fact Table
2) Remaining columns are attributes that we can use as:
    Criteria or Filters for our Reports & Dashboards
    Values we can lookup
    Helper Columns.

Primary Key	Attributes For Filtering	Attributes For Filtering
SalesRepKey	Sales Rep	Region
1	Gigi	USA
2	Freddy	Canada
3	Chin	Mexico
4	June	USA
5	Tyrone	Europe

Primary Key	Attributes For Filtering	Values used in Calculations	Values used in Calculations
ProductKey	Product	Price	SCost
1	Fun Fly	9.99	6.49
2	Sunshine	19.99	8.8
3	Fast Catch	23.45	14.54
4	Yanaki	24.95	15.22
5	Carlota	25.99	17.15
6	Crested Beaut	32.99	14.19
7	Quad	43.69	22.72

5) **Is a number a fact or an attribute?**
- If Price or Cost does not change much and is discrete, it seems like a dimension
- If Price or Cost changes often or for each transaction, then it seems more like a fact.

6) **History of Fact and Dimension**
- Terms "Fact" and "Dimension" go back to 1960s (General Mills and Dartmouth University), 1970s (AC Nielsen & IRI), but was popularized by Ralph Kimball in the 1980s.

7) Relationships
- Connects tables so we can build solutions based on fields from multiple tables.
- Relationships between tables allows us to:
    i. Use Criteria from Dimension tables to filter Fact Table Calculations (Measures).
        1. The Arrow Direction on the Relationship Arrow indicates the direction of the filter (From Dimension to Fact Table)
        2. What makes the filtering of a Fact Table so efficient, is that a Measure (Formula) will then have to make its calculation over fewer rows. If you are an Excel person, compare this to Array Formulas that must calculate over all rows in a column even if there are rows that do not match the criteria. Array Formulas are notoriously slow to calculate. DAX Measures that can work over a filtered Fact Table are usually fast to complete the calculations.
    ii. Lookup values
    iii. For Excel people, it helps avoid VLOOKUP driven solutions that may have many spreadsheet formulas that slows down query time (change criteria on report)
- Types of Relationships:
    i. One-To-Many
        1. Example: There is one unique product in each row of a Product Dimension / Lookup Table, but there can be many products in the Sales Fact Table. The Product Dimension Table is on the One-Side and the Sales Fact Table is on the Many-Side.
        2. Excel users are used to a One-To-Many Relationships from using the VLOOKUP Function. When you lookup a product price in the sales table, the sales side is the Many-Side and the product lookup table is the One-Side.
    ii. Many-To-Many
        1. Examples:
            - You may have an Order Table and an Invoice Table, where orders can be partially invoiced resulting in one order with multiple invoices; and multiple orders may be filled in one invoice.
            - Bakery:
                i. Finished Goods and Ingredients, Many To Many
            - Manufacturer
                i. Machines and parts, Many To Many
            - Chris Web Showed an Example where the totals were correct even though there were duplicate values in the row area of the report
    iii. One-To-One
        1. Example: If you have a Sales Rep Table and a Auto Table and each Sale Rep is issued only one car.

8) Data Models

- Structure of Data Model must come from the type of Measures & Reports that the Business Decision Maker requires.
- Requirements for good Data Model:
    i. Contains necessary data for end solution
    ii. Easy for the decision maker to use
    iii. Allows fast queries (change criteria and solution updates quickly)
    iv. Is easy to update when new data is available
    v. Easy to update structurally, if needed
- Data Models can be stored in Power Pivot & Power BI Desktop
- Data Model can Contain:
    i. Fact Table / Tables
    ii. Dimension Tables
    iii. Relationships
    iv. Helper Columns / Tables
    v. DAX Calculated Columns, DAX Measures, DAX Tables
    vi. Hidden / Not Hidden Tables and Columns
    vii. Formatting
- Data Model is also known as: "Semantic Model", which is defined as a model that represents the real word relationships between Facts and Entities and contains the data, relationships, calculated columns and other complexities together as a single model and allows us to create DAX Queries / Measures (formulas, calculations) that can be used in multiple reports or end solutions. This is in comparison to storing data in a relational or object-oriented database that require that you redefine the relationships and other complexities each time you create a Query or Measure.
- Some types of models:
    i. Flat Table = one table with all columns necessary to make report. Facts and Dimensions were mixed together in one table.
        1. Before Power Pivot, we had to create one Flat Table when we wanted to make PivotTable Reports.
        2. If we created the flat table with formulas, it could slow down the calculation speed of the workbook.
    ii. **Star Schema Data Model = Dimensional Model** :
        1. Fact Table with most atomic granular data surrounded by Dimension Tables using One-To-Many Relationships
        2. Star Schema Models tend to:
            - Be easier to use than other models.
            - Allow faster queries than other types of models.
            - Any Criteria or Filter from Dimension Tables can accurately filter the Measures (calculations) in the final reports.
            - Create One Formula that can be used in many reports
        3. Power Pivot & Power BI Desktop are designed to work efficiently with the Star Schema Data Model.
        4. Example of simple Star Schema Data Model and PivotTable Report:

Total Sales	Manufacturer ▼			
Sales Rep ▼	Colorado	Darnell	Gel Booms	Grand Total
Chin	$684.48	$316.86	$1,687.81	$2,689.15
Freddy		$1,141.96	$376.69	$1,518.65
Gigi	$620.95	$484.31	$1,201.01	$2,306.27
June	$546.06	$1,038.10	$269.52	$1,853.68
Tyrone	$127.10		$1,236.00	$1,363.10
Grand Total	$1,978.59	$2,981.23	$4,771.03	$9,730.85

   iii. **Snow Flake Data Model** =
1. Follows Database Normalization rules where columns in tables should not have duplicate values and must be broken apart into separate tables.
2. Example: Rather than have a Manufacturer Column in a Product Table Snow Flake Schema would have a Product Table and Manufacturer Table with a relationship between the two tables.
3. In this class if we encounter a Snow Flake Schema, we will use Power Query to transform it to Star Schema.
4. Why we will avoid Snow Flake:
   - i. Columnar Database will store Dimension Tables with duplicates in an efficient way. No need to complicate model with many tables.
   - ii. According to tests done by Marco Russo and Alberto Ferrari, for larger models the extra relationships used in a Snow Flake Schema can be less efficient than a single Dimension table with Duplicate Values (for exact models you must test, however).
   - iii. Hierarchies cannot be made from Snow Flake Model.
5. Example of Snow Flake Model and PivotTable Report:

Total Sales	Manufacturer ▼			
Sales Rep ▼	Colorado	Darnell	Gel Booms	Grand Total
Chin	$684.48	$316.86	$1,687.81	$2,689.15
Freddy		$1,141.96	$376.69	$1,518.65
Gigi	$620.95	$484.31	$1,201.01	$2,306.27
June	$546.06	$1,038.10	$269.52	$1,853.68
Tyrone	$127.10		$1,236.00	$1,363.10
Grand Total	$1,978.59	$2,981.23	$4,771.03	$9,730.85

   iv. Other Models
1. There may be cases where we have to build a Data Model that is not Star Schema because of business requirements.

9) **Note: Relational Database = Normalized Database**
   - Database that stores company's raw data with the goals of 1) no-redundancy and 2) accuracy.
   - Relational databases can have many tables and many relationships between tables.
   - In general, values are entered into one and only one location. There are usually not columns with duplicate values.
   - Relational databases serve the goal of storing raw data with accuracy, but they can be complicated to navigate, and queries may not perform as fast as is necessary for end users creating reports.
   - In this class when we connect to a Relational Database, we will use Power Query to extract the necessary data and then transform it and load it into the Star Schema Data Model.

10) **What is Data Modeling and what are some Examples?**
   - Data Modeling = Connecting to data, cleaning, transforming and modeling the data into an efficient form that can allow desired useful information (reports, visualizations, dashboards).
   - For Power Pivot and Power BI, an efficient form is usually a Star Schema Data Model, or Dimensional Model.
   - Dimensional Model consists of a Fact Table connected to Dimensional through Relationships.
   - Date and Time Dimension Tables are usually required when we have date or time calculations. This means that if we get data sets (like CSV files from public databases), we may have to construct Data or Time Dimension Tables during the Data Modeling phase of a project.
   - If we have lots of data and we want to minimize the size of the data stored in the in RAM Memory Columnar Database, we can use these general guidelines:
     i. Only import tables that you will need.
     ii. Only import columns that you will need.
     iii. Only import rows that you need.
     iv. Since the Columnar database stores individual columns as Unique Lists, reducing the cardinality of columns by round numbers, or breaking columns apart like Date-Time Columns can help reduce database size.
     v. Make as many calculations at the data source (like SQL database) or in the Data Modeling phase in Power Query to avoid DAX Calculated Columns. DAX Calculated Columns are not stored as efficiently as raw data columns (Video #14).
   - If you have a Snow Flake Data Model, during the Data Modeling phase in Power Query we can pull the Snow Flaked Tables into a single Dimension Table.
   - If we have Many-To-Many Relationships, we may need to build Bridge Tables.
   - If we have more than one Fact Table, we may need to perform Data Modeling to pull all the facts into one table, or build a Model that can accommodate two or more Fact Tables.
   - Sometimes if we have DAX Calculated Columns that take up too much RAM, we may have to choose:
     i. Between DAX Calculated Column that uses RAM and a Measure that takes time to calculate when a condition is changed in the report or visualizations.
     ii. Or make the calculation in Power Query (which requires time for calculation during data refreshing).
     iii. Or try to have the calculation pre-made at the data source, like at the SQL Database.
   - Sometimes we may have multiple Date Columns (like Order Date and Shipping Date) and we might need to build active and inactive relationships or build two Data Tables (depending on required reports).

11) Columnar Database (Part of Data Model)

- Columnar Database is part of the Data Model in both Excel Power Pivot and Power BI Desktop.
- When you import tables to the Excel Power Pivot or Power BI Desktop Data Model, the data is stored in the Columnar Database.
- Define Columnar Database
    - i. Behind the scenes in RAM Memory Efficient Big Data Analytics Database
        1. **"Behind the scenes"** = When you open the Excel file, the Columnar Database is opened behind the scenes in RAM Memory.
        2. **"Efficient Big Data"** = Database encodes and compresses data and stores it in a structure that allows DAX Formulas to make calculations quickly and produces a small file size.
        3. **"Analytics Database"** = Database is specifically designed to work with the DAX Formula language to make calculations quickly on "Big Data".
- Columnar Database has many synonyms, such as:
    - i. Columnar database
    - ii. VertiPaq engine
    - iii. SSAS Tabular (SQL Server Analysis Services Tabular)
    - iv. Storage Engine
    - v. XVelocity analytics engine
- Overview of What Columnar Database does
    - i. Database takes a table with many columns and stores each column separately as a unique list of values and builds a map to help it reconstruct the table when needed for making DAX Formula calculations.
    - ii. Why store data is separate columns with a unique list of items?
        1. Many calculations can be performed more quickly on individual columns with unique lists of values, rather than making calculation row by row.
        2. File size is reduced, and when the database is loaded into RAM Memory, less RAM is used, more is left for other tasks.
    - iii. This means that for columns with only a few unique values, the file size reduction can be dramatic. (Number of unique items in a column = "Cardinality").
    - iv. Here is a picture that helps to think about how data from a table is stored in the Columnar Database:

Columnar Database takes each column in the table and stores the columns separately, each as a unique list of items.

Original Table:

Sales	Sales Rep	Region
$54.00	Jo	West
$26.00	Nina	East
$54.00	Jo	South
$57.00	Kip	West
$22.00	Gigi	West
$59.00	Gigi	South
$95.00	Kip	East
$99.00	Kip	South
$51.00	Nina	South
$49.00	Nina	East
$12.00	Jo	East
$30.00	Jo	East
$20.00	Nina	West
$92.00	Kip	West
$73.00	Gigi	South

You can picture a Columnar Database like this:

Sales
$54.00
$26.00
$57.00
$22.00
$59.00
$95.00
$99.00
$51.00
$49.00
$12.00
$30.00
$20.00
$92.00
$73.00

Sales Rep
Jo
Nina
Kip
Gigi

Region
West
East
South

- Columnar Database performs "Vertipaq" Compression when a Table is imported
  - i. Steps is VertiPaq Compression:
    1. Read source dataset.
    2. For each Raw Data Column in the table, a unique list of values is determined, and that list is encoded, compressed and stored as a separate unit (memory block with a certain file size).
    3. Relationships are also stored in the Columnar Database.
    4. DAX Calculated Column are also calculated and compressed, but these columns are compressed last in the process and sometimes are not stored as efficiently as Raw Data Columns.
    5. An internal "map" is created that helps the Columnar Database engine can use to reconstruct the original table when it is needed to make a calculation.
       - This map helps DAX Formulas to work quickly.
         - i. If the calculation is on a single column, the map works quickly because it is working on a single unique list of items, a single memory black.
         - ii. If the calculation is on a few columns, the map must work between multiple columns and will take longer than a single column calculation.
         - iii. If the calculation is complex, at some point the map will actually materialize a full table in order to make the calculation. These calculations can sometimes take a long time.
  - ii. The Vertipaq Compression uses many different techniques to encode and compress the data. It depends on such things as data types, variation in the data, mathematical patterns and more. Some of the techniques are proprietary to Microsoft and we cannot know the exact process.
    1. Some of the techniques used are:
       - Value Encoding. For integer values, the Vertipaq engine may determine mathematical patterns that can help encode and compress to save the data in a smaller file size. This is called "Value Encoding".
       - Dictionary Encoding. For strings, the Vertipaq engine may build a dictionary of unique values with a first column of lookup integer values, and then replace the values in the original column with the integer lookup values. This is called "Dictionary Encoding". This can reduce file size. A picture of this is seen here:

Original Column:		Integer Index:		Dictionary:	
Sales Rep		**Sales Rep ID**		**Sales Rep ID**	**Sales Rep**
Jo		0		0	Jo
Nina		1		1	Nina
Jo		0		2	Kip
Kip		2		3	Gigi
Gigi		3			
Gigi		3			
Kip		2			
Kip		2			
Nina		1			
Nina		1			
Jo		0			
Jo		0			
Nina		1			
Kip		2			
Gigi		3			

What gets stored in Columnar Database

- **Run Length Encoding**. After Value Encoding or Dictionary Encoding is performed, Run Length Encoding may be able to further reduce the file size. This method tries to reduce file size by avoiding repeat values. If there is a pattern of many consecutive repeated values, Run Length Encoding will keep the Dictionary, but rather than having an Index Column is build a table that simple counts the repeated values, as seen in the below picture.

Original Column:		Integer Index:	Dictionary:				Dictionary:	
=> Dictionary Encoding =>						=> Run Legth Encoding =>		
Sales Rep		**Sales Rep ID**	**Sales Rep ID**	**Sales Rep**			**Sales Rep ID**	**Sales Rep**
Gigi		0	0	Gigi			0	Gigi
Gigi		0	1	Jo			1	Jo
Gigi		0	2	Kip			2	Kip
Jo		1	3	Nina			3	Nina
Jo		1						
Jo		1					**Data Rows:**	
Jo		1						
Kip		2					**Sales Rep ID**	**Sales Rep**
Kip		2					0	3
Kip		2					1	4
Kip		2					2	4
Nina		3					3	4
Nina		3						
Nina		3						
Nina		3						

What gets stored in Columnar Database
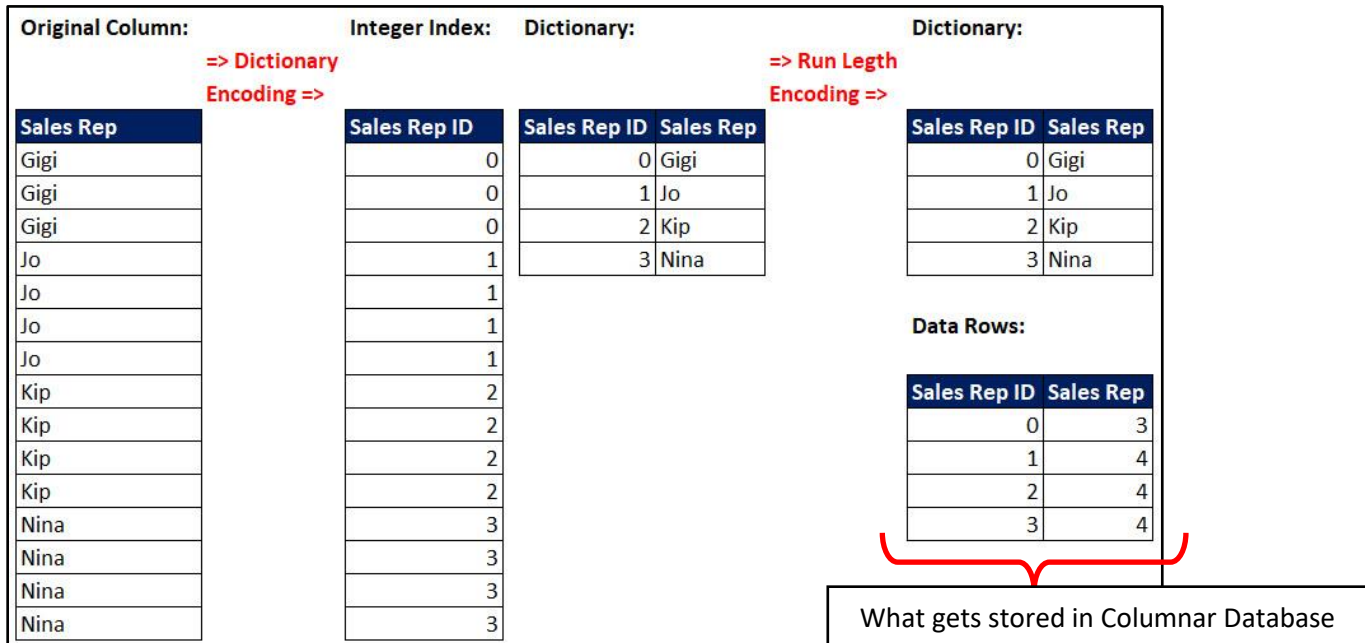
- Value Encoding, Dictionary Encoding, and Run Length Encoding help reduce file size and help formulas calculate more quickly.

iii. **Cardinality**. Number of unique items in a column = "Cardinality".
   1. It is the Cardinality that is the primary factor in determining the file size
iv. **Some Factors in Determining File Size:**
   1. Number of Columns.
   2. Cardinality of columns. The more unique items there are the more space the column will take up in file size.
      - If you have numbers with many decimals, and you don't need them, then round the numbers before you import them. This will result in a column with far fewer unique values and therefore the column will be compressed more and result in a smaller file size.
   3. If there is a pattern of consecutive repeated items in a column, file size may be reduced.
   4. Other techniques also…

12) Clean Raw Data
- Fix unusable raw data so that it can be used to perform data analysis
- Examples:
    i. Split data apart into desired data, like "Gel Boomerangs - CA" ==>> "Gel Boomerangs", "CA"
    ii. Join data together to get desired data, like 2018 and Quarter 1 ==>> "2018 - Quarter 01"
    iii. Other cleaning goals.

13) Transform Data Sets
- Fix unusable data set so that it can be used to perform data analysis
- Examples:
    i. Add, remove or filter columns in data sets
    ii. Combine, merge, append or unpivot data sets
    iii. Taking Relational Database with many tables and converting it into a Star Schema using Power Query
    iv. Other data transformational goals

14) Import Data
- Import data from external sources (single or multiple sources) into Excel or Power Pivot's Data Model or Power BI Desktop;
- Optimally, the import will allow refreshes so Reports and Dashboards update when source data changes.

15) ETL Term
- Extract, Transform and Load Data (term from Data Warehousing)

16) Data Warehousing Term
- Term that means get data from original source and form it so that you can make useful reports and dashboards.
- Term made famous by Ralph Kimball.
- We will be doing a version of Data Warehousing when we build Star Schema Data Models with MS Power Tools.

17) **Steps in Video to Reduce Columnar Database Size using Power Query** :
   i.   Goals:
      1.   Download large CSV File from government web site.
      2.   The Data we will download is City of Seattle 911 Call Data. The full Description of the data is here:
         i.   This data represents police response activity. Each row is a record of a Call for Service (CfS) logged with the Seattle Police Department (SPD) Communications Center. Calls originated from the community and range from in progress or active emergencies to requests for problem solving. Additionally, officers will log calls from their observations of the field...
      3.   The web site we will download the data from is: https://data.seattle.gov/Public-Safety/Call-Data/33kz-ixgy
      4.   Import CSV File into the Excel Power Pivot Columnar Database (Data Model).
      5.   Make transformations to get only data we need.
      6.   Make transformations to reduce the size of the Columnar Database in the Data Model, and thus the file size.
      7.   Create a Dashboard that shows the count of Top 20 Call Types that can be sliced by the police precinct, sector, and beat. The Dashboard might look like this:

ii. Steps:
1. Download Seattle City 911 Call Data Excel CSV File with these steps:
    i. Go to this web site: https://data.seattle.gov/Public-Safety/Call-Data/33kz-ixgy
    ii. Click Export button.
    iii. Click CSV for Excel button.

iii. Open the file named "021-MSPTDA-ReduceDataModelSize-Start.xlsx".
iv. After you have download CSV file, in the Data Ribbon Tab, in the Get & Transform group, click the "From Text/CSV" button, as seen here:
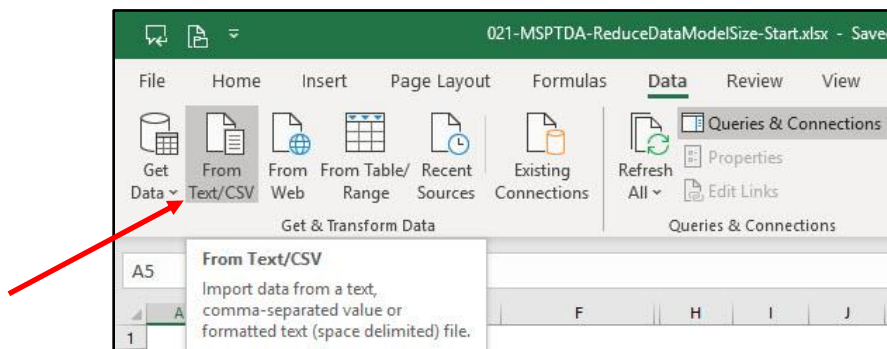
v. Download to a location and then click the Import button, as seen here:
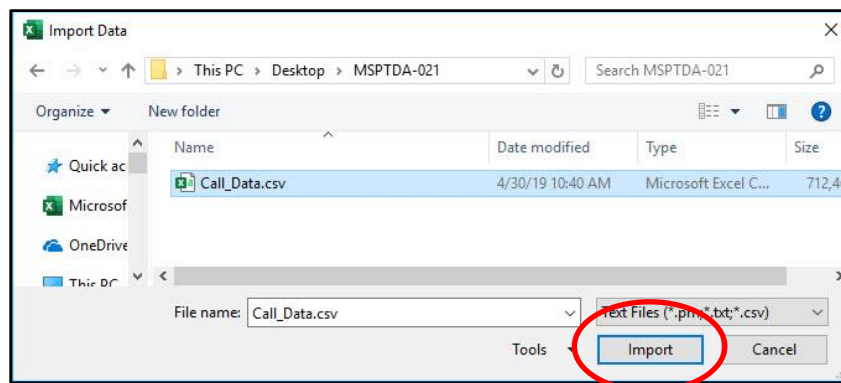
vi. In the next dialog box, make sure the Delimiter is Comma, then click the Transform Data button (instead of the Transform Data button, it may be the Edit button), as seen here:
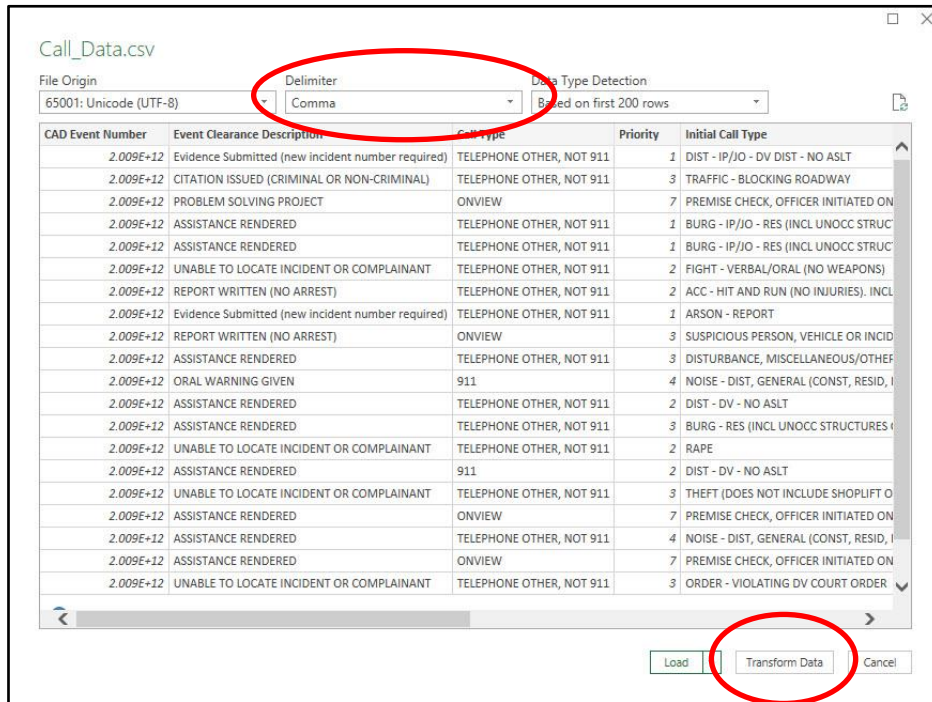
vii. In the Power Query Editor:

1. Give the data set a good name (default name of Call_Data is an okay name).
2. In the Home Ribbon Tab, in the Close group, click the Close & Load dropdown arrow, then click on Close & Load To…

viii. In the Import Data dialog box:

1. Select dialog button for Only Create Connections.
2. Check the checkbox for "Add this data to the Data Model".

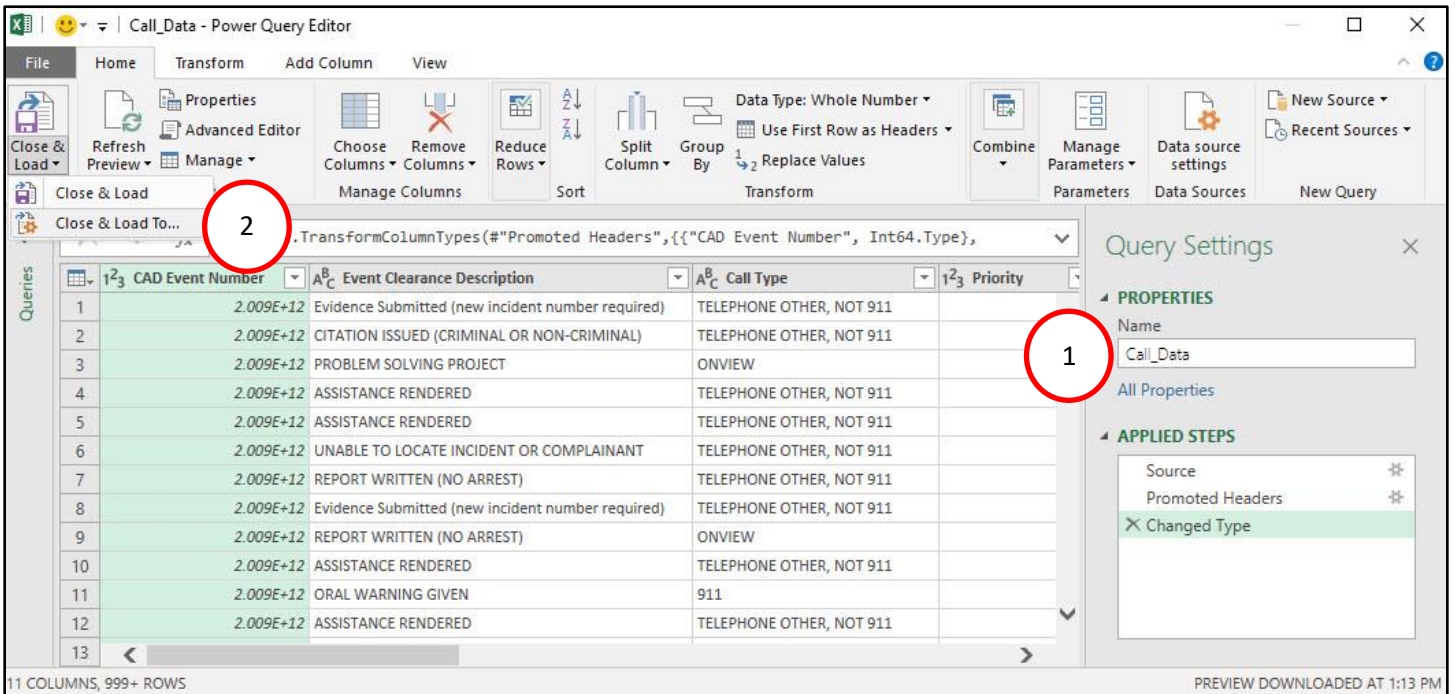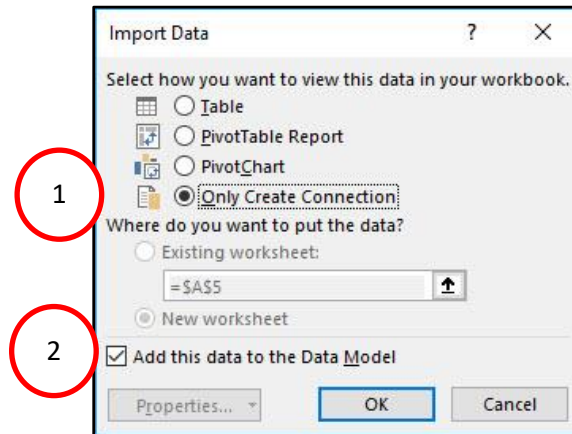ix. After the data is loaded to the Data Model, notice in the below picture that the Queries & Connections Pane shows that the Call_Data Query has loaded 3.9 million rows of data. Although 3.9 million rows of data were loaded, during the load, the data was compressed into the Columnar Database (part of the Data Model) which contains one column for each column in the table, where each column stores a unique list of items.
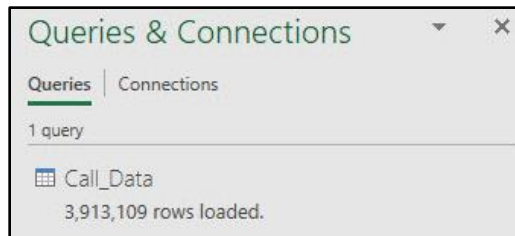
x. Click Ctrl + S to save the loaded data in the Columnar Database to the Excel file. If we look at the size of the file in Windows Explorer, it is over 180 MB in size, as seen here:

021-MSPTDA-ReduceDataModelSize-Start.xlsx   183,760 KB
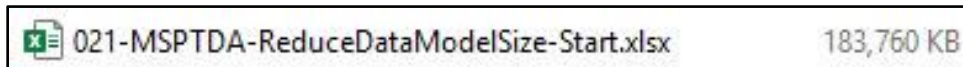
xi. To edit the query, in the Queries & Connections Pane, double-click the Call_Data Query, or right-click and click on Edit.

xii. Filter Rows: Scrolling over, to find the Original Time Queued column. Click the Filter Arrow at the top of the column, then point to Date/Time Filters, then click on Between. Complete the Filter Rows dialog box so that only the rows where the date is equal to or after 1/1/2017 AND is before or equal to 12/31/2018, as seen here:
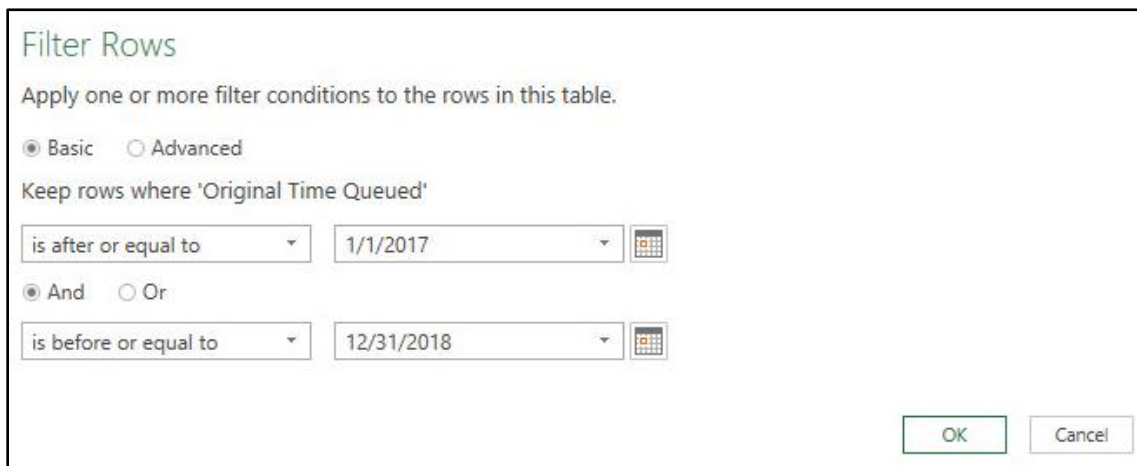
xiii. Click the Close & Load Button to re-load the filtered data to the Columnar Database in the Data Model. Because we filtered rows in the table, there are now only 858,323 rows of data loaded. In the Queries & Connections Pane we can see that there are 858,323 rows loaded to the Data Model, as seen here:
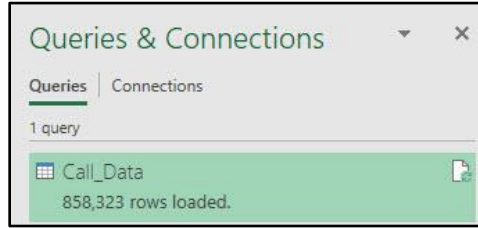
xiv. When the data was re-loaded, each column of unique items had to be re-built. Because we filtered rows out of the data set, many of the columns of unique items, now have many fewer items. If we look at the file size it is about 40 MB.
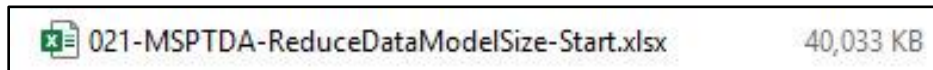
xv. **Remove Columns**: Now if we go back to the Power Query Editor and select only the five columns we want (Final Call Type, Original Time Queued, Precinct, Sector and Beat), and then right-click any one of the columns and click on Remove Other Columns, we get only the columns that we need for our analysis.
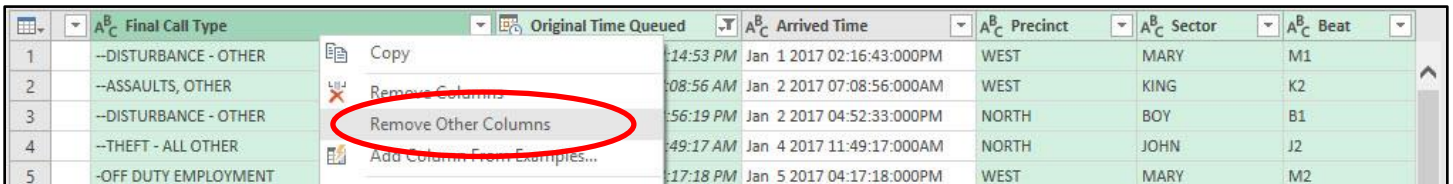
xvi. When we click Close & Load, the Columnar Database is re-built again, but now with only the five columns that we need. Although the number of rows does not change, the fact that we removed columns will significantly reduce the size of the Columnar Database in the Data Model and our file size. The most significant contribution in size reduction will come from the CAD Event Number column which contains over 3.9 million unique values. As we see in the below picture, our file size fell to about 3 MB:
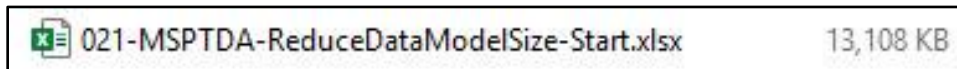

xvii. **Reduce Cardinality**: If we open up the Power Query Editor and look at the Original Time Queued column, we can see that there are many unique date-time possibilities. And in fact, if you count the unique items, there are almost as many unique items in the Original Time Queued column as there are records in the full data set (3.9 million). If you do not need this sort of detail (records down to the second) for your final report, then we should transform the column to the size or grain of detail that you need. For example, if we only need day and hour, this presents an opportunity for us to significantly reduce the cardinality of the column and therefore significantly reduce the size the of unique lists for each column that is stored in the Columnar database. If we can extract the day and hour data from this column into two columns, the date column will contain 730 unique days and the time column will contains 24 unique times. This transformation will significantly reduce the size of the Columnar Database in the Data Model and the file size.

xviii. To extract the hour and minute into a new column, we can:

1. **Add Column to Reduce Cardinality**: In the Add Column Ribbon Tab, in the General group, click the Custom Column button, as seen here:
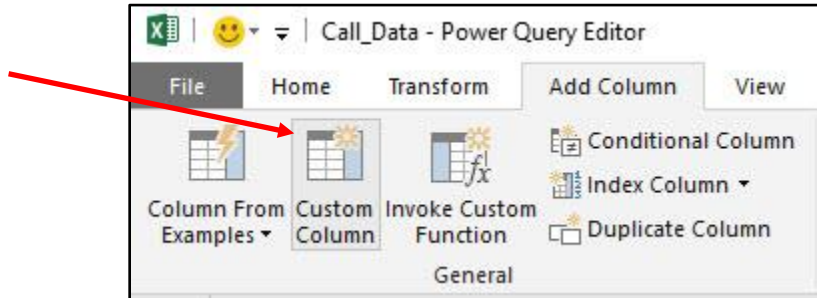
2. In the Custom Column dialog box, create the formula to extract an hour-minute time value from the Original Time Queued column, as seen below.

    i. The #time Power function has three arguments: #time(Hour,Minute,Seconds)

    ii. The Time.Hour Power Query function can extract the hour value from a date-time value.

    iii. The Time.Minute Power Query function can extract the minute value from a date-time value.
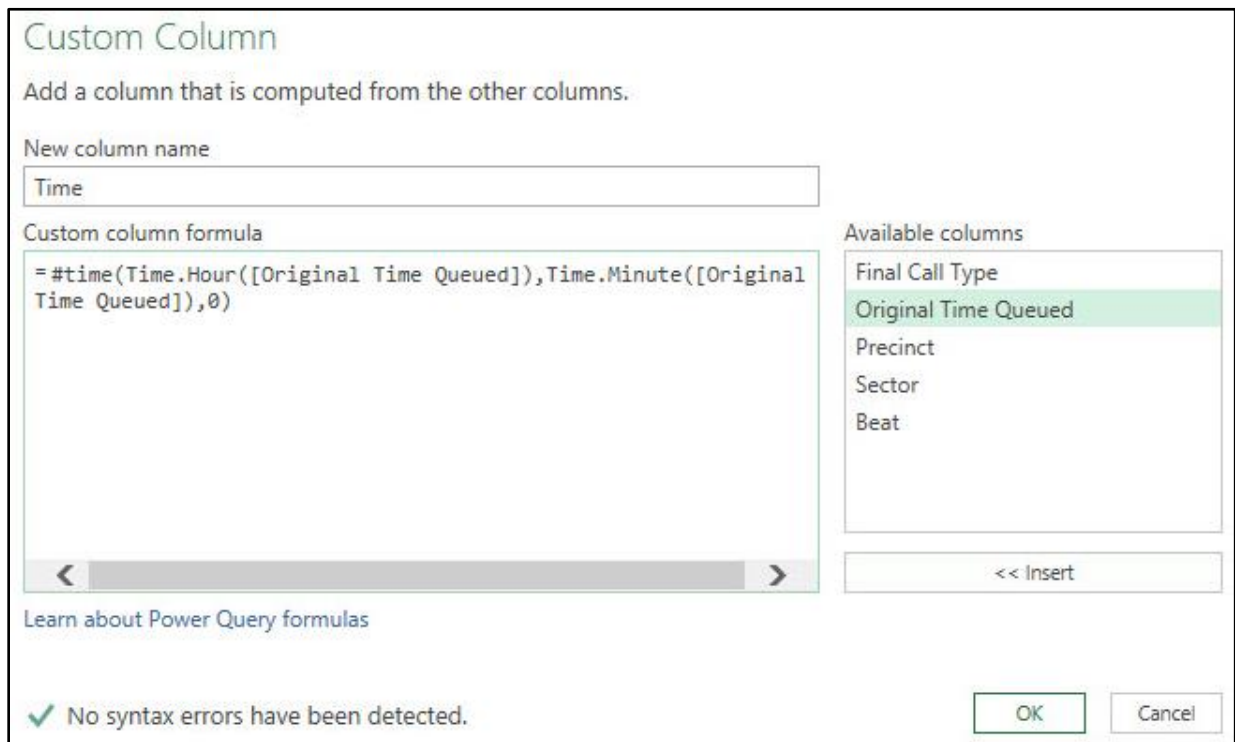
3. After we click OK in the Custom Column dialog box, we need to edit the Table.AddColumn function in the Formula Bar to add a Time Date Type to the new column, as seen here:

In the 4ᵗʰ argument of the Table.AddColumn function, add the type time
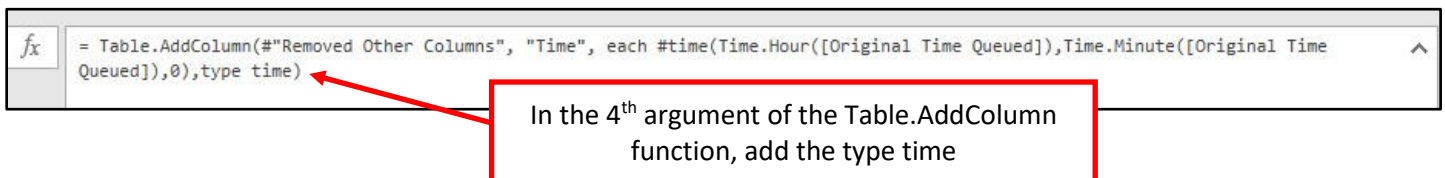
4. **Transform Column To Reduce Cardinality**: Select the Original Time Queued column, which has about 3.9 million unique values. In the Transform Ribbon Tab, in the Date & Time Calculation group, we want to find the Date dropdown that will allow us to transform the column into just date values.
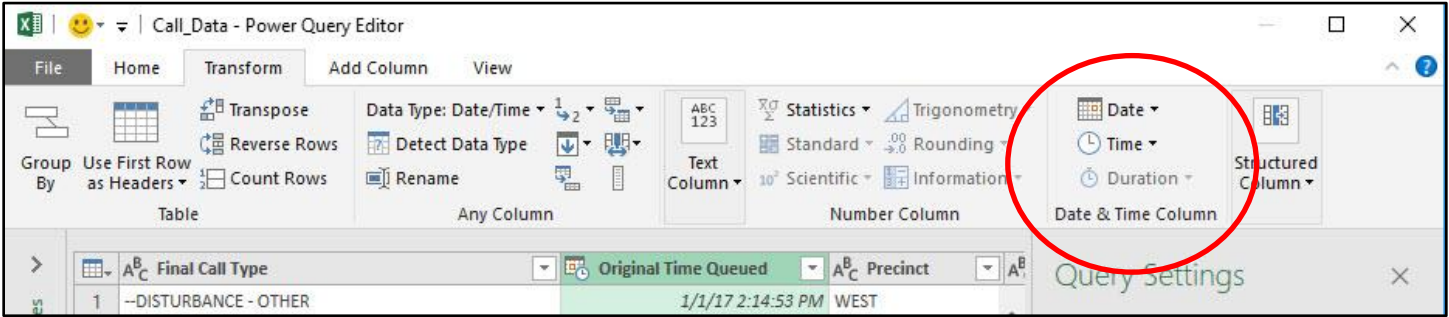
5. Click the Date dropdown, then click on the Date Only option, as seen here:
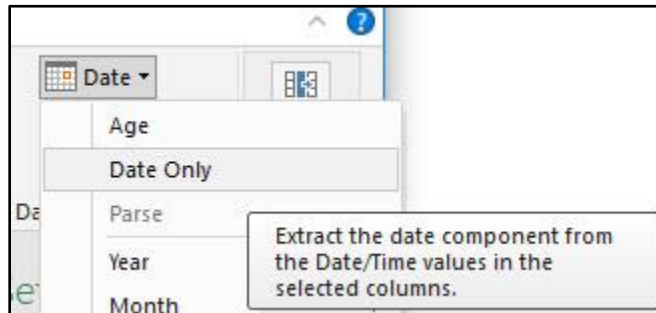
6. After the Original Time Queued column is transformed to date only, remane the column to "Date".

7. In the picture below we can see the resulting Date and Time columns.
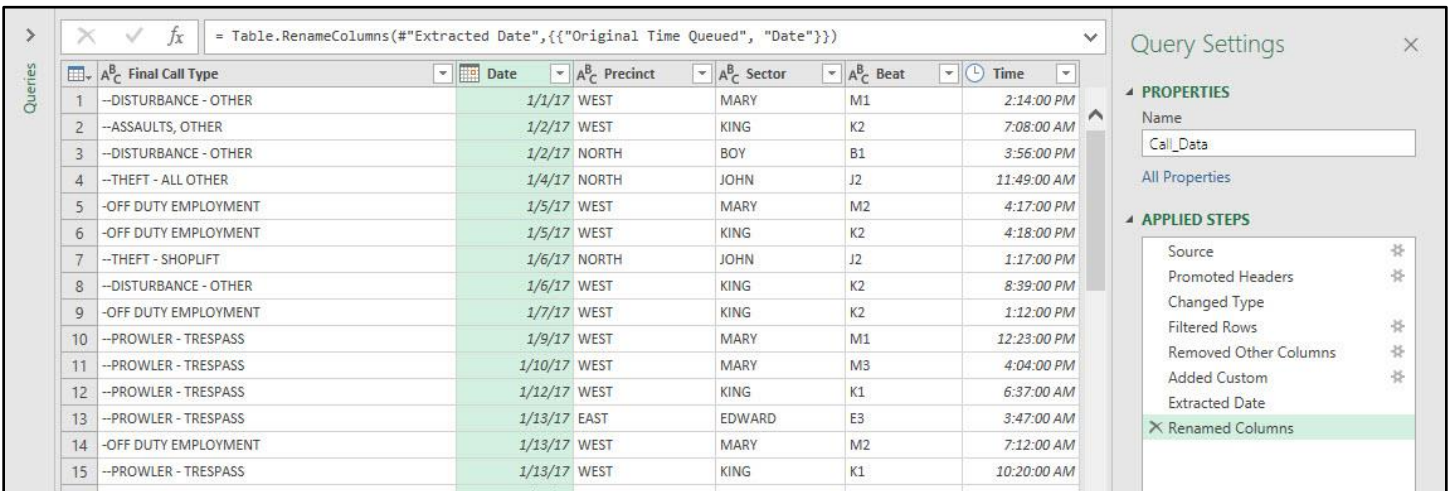
8. After we Close and Load the Power Query transformation to the Data Model, and then save the file, we can see the file size is down to about 3 MB.

9. If our analysis will only use date and hour as conditions, we can further reduce the size of the Columnar Database by editing the formula for the time column to just extract the hour, rather than hour and minute, as seen here:

10. Our Final Columnar Database Data Model and file size is seen here:
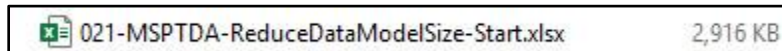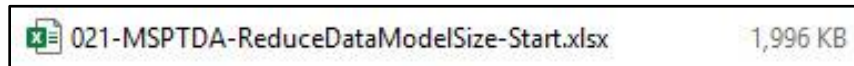

021-MSPTDA-ReduceDataModelSize-Start.xlsx   1,996 KB

18) **Steps to create Final Data Model and Dashboard** :

i. Using the PowerPivot automatic Data Table feature, we created a Date Table (as seen in this video and earlier in this MSPTDA class).

ii. Create Relationship between Date Table and Call_Data Table (as seen in this video and earlier in this MSPTDA class).

iii. Create Measure to Count Rows in the Call_Data Table (as seen in this video and earlier in this MSPTDA class).

iv. Used Show Values As feature to create % Difference From (as seen in this video and earlier in this MSPTDA class).

v. Added Slicers for Sector, Precinct and Beat (as seen in this video and earlier in this MSPTDA class).

vi. Filter the PivotTable to show Top 20 counts of Call Type (as seen in this video and earlier in this MSPTDA class).

vii. Sort the count of Call Type Measure Z to A (as seen in this video and earlier in this MSPTDA class).

viii. The final Dashboard can be seen on the next page:

Seattle 911 Call Data by Call Type, Precinct, Sector and Beat

911 Call Type	2017 911 Call Count	2018 911 Call Count	Change	Total 911 Call Count
--PREMISE CHECKS - CRIME PREVENTION	40,274	35,441	-12.00%	75,715
--DISTURBANCE - OTHER	32,924	32,151	-2.35%	65,075
--SUSPICIOUS CIRCUM. - SUSPICIOUS PERSON	29,895	31,113	4.07%	61,008
--TRAFFIC - MOVING VIOLATION	21,796	24,806	13.81%	46,602
--TRAFFIC - PARKING VIOL (EXCEPT ABANDONED CAR)	21,214	24,151	13.84%	45,365
--ASSIST PUBLIC - OTHER (NON-SPECIFIED)	17,687	18,453	4.33%	36,140
--TRAFFIC - MV COLLISION INVESTIGATION	18,398	17,343	-5.73%	35,741
-OFF DUTY EMPLOYMENT	15,104	15,479	2.48%	30,583
--PROWLER - TRESPASS	13,638	13,461	-1.30%	27,099
--DIRECTED PATROL ACTIVITY	8,924	12,234	37.09%	21,158
--CRISIS COMPLAINT - GENERAL	10,238	10,786	5.35%	21,024
--MISCHIEF OR NUISANCE - GENERAL	8,777	9,677	10.25%	18,454
--SUSPICIOUS CIRCUM. - SUSPICIOUS VEHICLE	9,559	8,486	-11.23%	18,045
--THEFT - CAR PROWL	8,319	6,880	-17.30%	15,199
--THEFT - ALL OTHER	7,331	7,204	-1.73%	14,535
--ASSIST OTHER AGENCY - CITY AGENCY	6,938	7,352	5.97%	14,290
--ALARM-COMM (INC BANK, ATM, SCHOOLS, BSN)	6,392	6,746	5.54%	13,138
--THEFT - SHOPLIFT	5,997	6,517	8.67%	12,514
--ASSAULTS, OTHER	5,733	6,305	9.98%	12,038
--INTOX & LIQ VIOLS - INTOXICATED PERSON	4,092	7,723	88.73%	11,815
Grand Total	**293,230**	**302,308**	**3.10%**	**595,538**

Precinct: EAST, NORTH, SOUTH, SOUTHWEST, UNKNOWN, WEST

Sector: BOY, CHARLIE, DAVID, EDWARD, FRANK, GEORGE, JOHN, KING, LINCOLN, MARY, NORA, NULL, OCEAN, QUEEN, ROBERT, SAM, UNION, WILLIAM

Beat: 99, B1, B2, B3, BS, C1, C2, C3, CCD, CD, CDD, CMD, COMM, CS, CTY, D1, D2, D3, DBB, DCD, DET, DND, DS, DSPV, DSPVD, E, E1, E2, E3, ED, EP, ES, F1, F2, F3, FS, G1, G2, G3, GS, H1, H3, HBR, HS, INV, J1, J2, J3, JS, K1, K2, K3, KS, L1, L2, L3, LAPT, LS, M1, M2, M3, MS, N, N1, N2, N3, ND, NP, NS, O1, O2, O3, ODD, OS, Q1, Q2, Q3, QS, R1, R2, R3, RS, S, S1, S2, S3, SCTR1, SD, SOUTH, SP, SPVD, SPVDD, SS, SW, SWT, T, TAC2, TAC9, TRF, U1, U2, U3, US, W, W1, W2, W3, WP, WS

Measure is:
=COUNTROWS(Call_Data)

Show Values As calculation:
% Difference From (Year,previous)