Some Common Rookie Errors That Occur when putting records into groups with CASE() statements

Summary: The prior document demonstrated that CASE statements are useful to

|  |  |  |  |
| --- | --- | --- | --- |
| City | Deaths | Population | Deaths per 100k |
| St Louis | 261 | 294890 | 88.5 |
| Baltimore | 321 | 575584 | 55.8 |
| New Orleans | 194 | 388424 | 49.9 |
| Detroit | 327 | 664139 | 49.2 |
| Cleveland | 175 | 376599 | 46.5 |
| Memphis | 290 | 651011 | 44.5 |
| Kansas City | 174 | 501957 | 34.7 |
| Milwaukee | 191 | 587721 | 32.5 |
| Philadelphia | 469 | 1590402 | 29.5 |
| Atlanta | 150 | 524067 | 28.6 |
| Louisville | 173 | 615924 | 28.1 |
| Chicago | 748 | 2670406 | 28.0 |
| Washington | 197 | 714153 | 27.6 |
| Indianapolis | 214 | 887232 | 24.1 |
| Dallas | 245 | 1400337 | 17.5 |
| Houston | 398 | 3378146 | 11.8 |
| Phoenix | 188 | 1743469 | 10.8 |
| Los Angeles | 343 | 4085014 | 8.4 |
| Seattle | 50 | 737015 | 6.8 |
| New York | 437 | 8622357 | 5.1 |

|  |  |
| --- | --- |
| City | Deaths |
| Chicago | 748 |
| Philadelphia | 469 |
| New York | 437 |
| Houston | 398 |
| Los Angeles | 343 |
| Detroit | 327 |
| Baltimore | 321 |
| Memphis | 290 |
| St Louis | 261 |
| Dallas | 245 |
| Indianapolis | 214 |
| Washington | 197 |
| New Orleans | 194 |
| Milwaukee | 191 |
| Phoenix | 188 |
| Cleveland | 175 |
| Kansas City | 174 |
| Louisville | 173 |
| Atlanta | 150 |
| Seattle | 50 |

a) categorize records by adding a new textual field, that can be used as a slicer in reporting. The report can maintain the grain of the original data or summarize the data  
b) stop divide by zero errors when calculating averages  
c) change the granularity of the data by putting records into geographic groups

This document accentuates a few rookie errors that can trip you up. The genesis of this document was a student presentation one day in Pullman. A student group presented homicide data similar to the image on the left and made the claim that Los Angeles was much more dangerous place to live than Seattle, and noted the disparity of homicide deaths. Did the student make a rookie mistake to use frequency counts and make a summary claim on one metric? Certainly Los Angeles is a different place than Seattle, so different to not warrant a direct comparison.

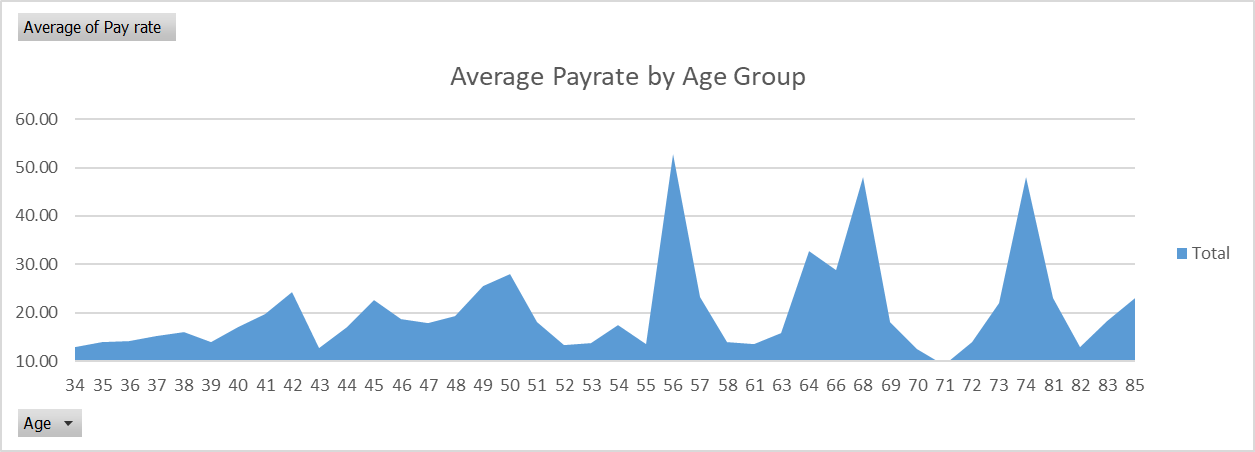
It is debatable whether frequency counts provide an accurate depiction of crime levels, but it certainly is a mistake to make strong assertions from one metric. When adjusting the data to account for population, the table of data on the right suggest that Los Angeles is more dangerous, but not by much.

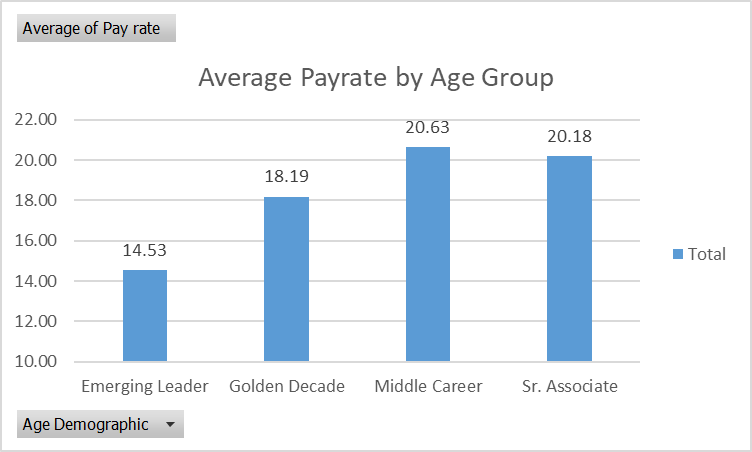
It is not surprising though that an argument for the danger of a city can be based on frequency with no concern for perspective such as deaths per 100,000. Analysts often over-simplify their analysis, sometimes due to lack concern or lack of time. Making a rookie error though can harm your career, so let’s take a look at some scenarios.

This module includes some very useful techniques for the business analyst to categorize records, however when you categorize records sometimes if the data is too granular, then you make false attributions to the data. When then the granularity is too high (too much detail) , and the sheer volume of the data can make it hard to make any sense of the data. You have to be careful though at the level of granularity that you use to analyze the data, sometimes if the grain is too high you can make misattributions. For example based on the chart below, a student suggested that in the hiring plans, to be careful not to hire 55 or 56 year olds.

Here we also categorize records into groups based on levels of a dimension. Using CASE does not change the granularity of the data, however s***imilar to a GROUP BY query when we draw the chart using the new column (the chart is performing a GROUP BY summarization for us, we change the granularity of the data***. So in the next example when the new column is created with the CASE() statement the underlying data is not changed. However when the chart is created, you can summarize the data in different levels, and you have to be careful when aggregating te data. The beauty of the CASE() statement used here is that the analyst envisions and chooses how to categorize the data. This means that given the same data set different analysts will segment and categorize the data differently each creating their own categorization scheme. You just have to be careful to make sure the summarization makes sense.

**Categorizing and Segmenting Data** - When there is too much data it is often useful to categorize (discretize, bin, group) the data and examine each group individually then compare the groups based on some important measures. Again, the beauty is that you the analyst envision, define and implement the categorization that is used to segment the data (which however can introduce bias and error, such as confirmation bias). Often if you look at categories of data one at a time, you can find patterns more easily. For example within a category of data (e.g., market segment) other dimensions can emerge that explain the phenomenon. Also if you create groups, then you can compare the groups to each other on important metrics. As we shall see sometimes the rookie analyst will categorize the data and read too much into the data and interpret it wrongly, so we discuss some examples. This dosument is to caution you the budding analyst to beware some common biases and cognitive errors.

The chart on the left displays the average hourly pay rate for the 300 employees at the AdventureWorks Company. It is easy for the rookie analyst to look at the chart on the left and report that something must be radically changing in the employees when they progress from age 55 to 58, that their wage increases then crashes. Drawing false conclusions from highly categorized data is a rookie mistake. It’s important to maintain your common sense when looking at data. Let’s discretize (group the records into different categories) and see if a different story is told.

When we group the employees using the CASE () statement below we can see a trend. Wages rise from group #2 (29-39 year olds) to group #3 (40 to 50 year olds), then slip back when moving to group #4 (51 to 59 year olds) and then seem to recover when compared to group #5 (60 and older). You have to be careful to NOT jump to conclusions when analyzing the chart. There can be many reasons for this payrate pattern. Rookies want to answer questions quickly and often grasp at the first explanation they uncover, rather than consider rival hypothesis, or confounding variables. Be careful not to make up stuff to explain what you see in charts, and don’t blindly believe pretty charts.

A wise analyst categorizes and cuts the data ***many different ways*** to preclude misinterpretation of the data. In other words more seasoned analysts are more prudent, and understand confirmation bias, where explanations of results are adopted because they fit the bias of the analyst. Here is the code that makes this analysis possible. *Notice the columns are not ordered correctly you need a numeric column to sort the textual column.*

, DATEDIFF(year, [BirthDate], GETDATE()) as Age

, CASE

WHEN DATEDIFF(year, [BirthDate], GETDATE()) < 28 THEN 'Jr. Associate'

WHEN DATEDIFF(year, [BirthDate], GETDATE()) BETWEEN 29 and 39 THEN 'Emerging Leader'

WHEN DATEDIFF(year, [BirthDate], GETDATE()) BETWEEN 40 and 50 THEN 'Middle Career'

WHEN DATEDIFF(year, [BirthDate], GETDATE()) BETWEEN 51 and 59 THEN 'Golden Decade'

WHEN DATEDIFF(year, [BirthDate], GETDATE()) > 59 THEN 'Sr. Associate'

END AS [Age Demographic]

|  |  |
| --- | --- |
| USE [AdventureWorksDW2014]  SELECT [DepartmentName], [Title], [LastName], [FirstName], [BaseRate]  FROM [dbo].[DimEmployee]  ORDER BY [DepartmentName], [Title], [LastName] | Run this warmup query |
| *--now run this query to examine the payrates by age. It is easy to assume the wrong factors are ‘causing this variation*  SELECT DATEDIFF(year, [BirthDate], GETDATE()) as [Age]  , AVG([BaseRate])  FROM [dbo].[DimEmployee]  GROUP BY DATEDIFF(year, [BirthDate], GETDATE())  ORDER BY [Age]  Recall GETDATE() is similar to Excel’s NOW() or TODAY() function to retrieve the time and date from the computer’s clock | While grouping data can provide insight, it can also mislead. SO triangulate and verify results rather than jump on initial findings.  Tip: show your charts to elder analysts and managers before presentations, to screen for rookie errors and misinterpretations. |
| *-- now add this modification*  SELECT DATEDIFF(year, [BirthDate], GETDATE()) as [Age]  , COUNT([EmployeeKey]) as [# employees]  , AVG([BaseRate]) as [Avg Payrate]  FROM [dbo].[DimEmployee]  GROUP BY DATEDIFF(year, [BirthDate], GETDATE())  ORDER BY [Age] | Columns: Number of employees by Age Line: Average payrate |
| USE [AdventureWorksDW2014]  SELECT  CASE  WHEN DATEDIFF(year, [BirthDate], GETDATE()) < 28 THEN '1. Jr. Associate'  WHEN DATEDIFF(year, [BirthDate], GETDATE()) BETWEEN 29 and 39 THEN '2. Associate  WHEN DATEDIFF(year, [BirthDate], GETDATE()) BETWEEN 40 and 50 THEN '3. Middle Career'  WHEN DATEDIFF(year, [BirthDate], GETDATE()) BETWEEN 51 and 59 THEN '4. Golden Decade'  WHEN DATEDIFF(year, [BirthDate], GETDATE()) > 59 THEN '5. Sr. Associate'  END AS [Age Group]  , [EmployeeKey]  , [BaseRate]  FROM [dbo].[DimEmployee]  ORDER BY [Age Group] | Does grouping the data help? |

If the above chart is not useful, what chart and analytics would be useful?  
How could you further analyze the data? (Hint think slicers, and think like a business leader.)

Chart

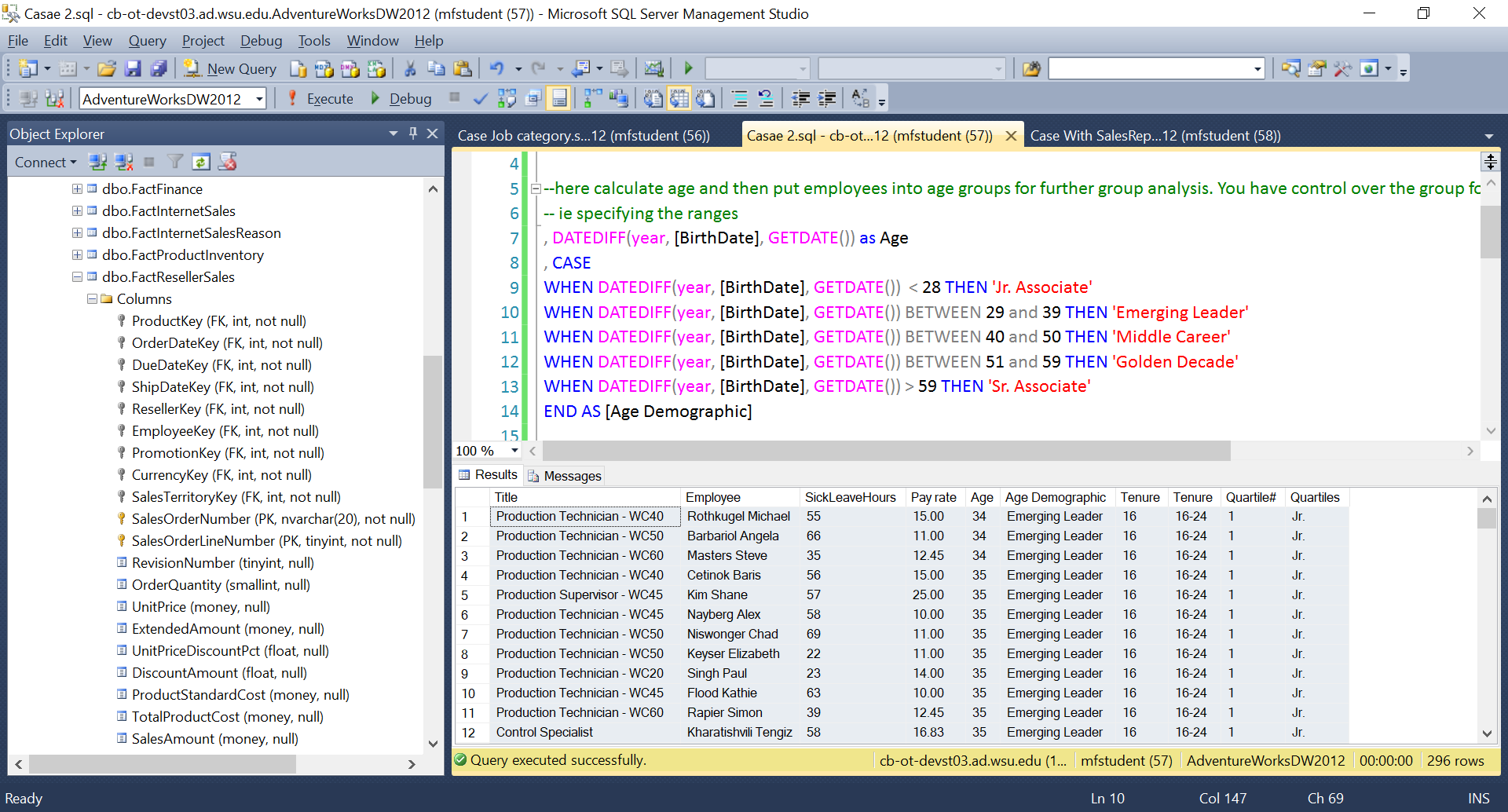
Description automatically generated*For example, how would you further analyze this next chart? Any chance for Rookie Errors? Let’s Discuss.*The chart designer is making the argument that age influences # orders sold by each salesrep. What other dimensions can help to explain the variance in sales rep performance?

***Next Scenario***

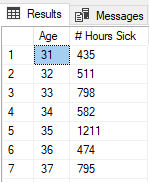
Next we examine the absenteeism of employees by age groups. Do younger or older employees call in sick more often?

First we calculate the age of the employee. The SQL DATEDIFF() formula can calculate the difference in 2 dates – for every row in the table, and then parse the result into days, months, years etc. Here we calculate the age of an employee and create a new column called Age. Next the CASE statement is used to create a textual new column ([Age Demographic]) with a term in it to the describe the group that was created. The group name is the used as a label for the X-axis for charts or used as a slicer.

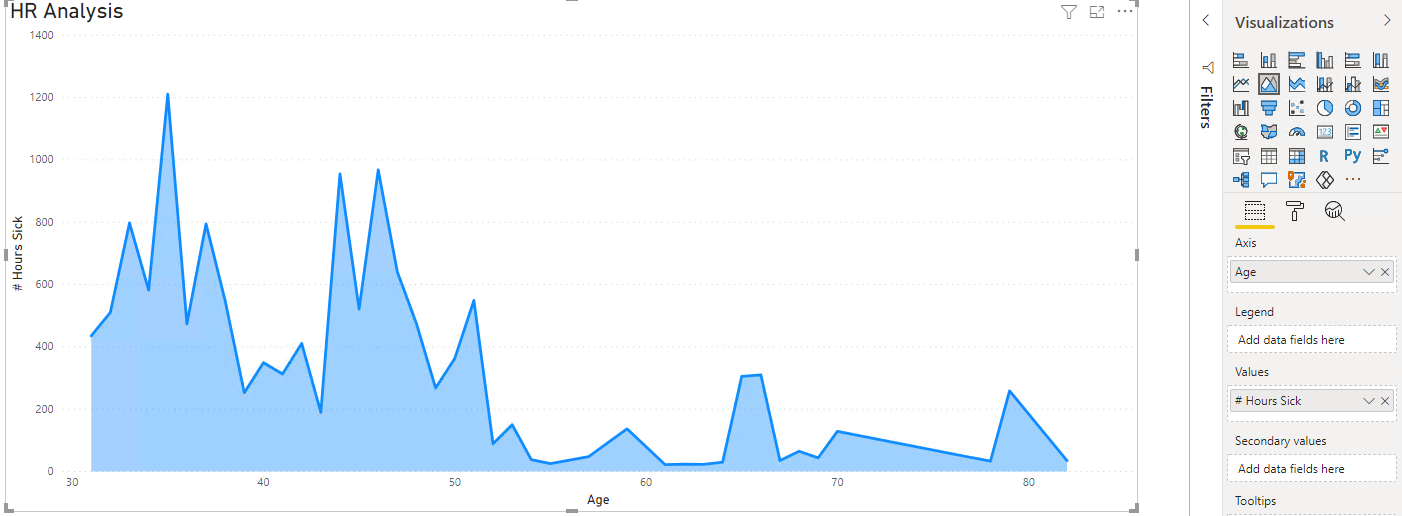
Case statements have been leveraged by data management professionals (DBA programmers) since the 1950’s. Case statements are used in many different programming languages and are commonly used to categorize dta prior to analysis. It is very useful to categorize and compile data. You can then dig deeper into one category or compare categories on important measures and metrics.



Before we talk about how this SQL technology works, a strong caution is given. Keep your common sense and perhaps ‘talk it out’ when interpreting a chart to test your interpretations. In the above scenario, Do you really think that employees gain and lose income based on age? There are so many other factors that could influence the current make-up of the dataset. In this company turnover can also explain why there are dips and gains in attributes of the dataset. So don’t just believe the chart that is drawn, rather ask. “How confident am in these results?, and would I make an important decision based on this report?

It is so easy to get fooled by nice looking charts. Here is another example that if you understand the analytics process, can protect you from looking like a rookie (presenting wrong data or data wrong). Here after making a new column of derived data that holds the age of the employee, then you can use that new dimension field in the X-axis of an area chart. If we simply SUM the number of sick leave hours used by the employees of the company, you might get tricked into believing that there is a rash of illness for the 39, 47, 49, and 69 year olds. Right?

|  |  |
| --- | --- |
| USE [AdventureWorksDW2014]  SELECT  DATEDIFF(year, [BirthDate], GETDATE()) as [Age]  , SUM([SickLeaveHours]) as [# Hours Sick]  FROM [dbo].[DimEmployee]  GROUP BY DATEDIFF(year, [BirthDate], GETDATE())  ORDER BY [Age] | This is a typical starting place. This data has been displayed with the area chart below. |
| --Here we add some reality – count the # employees and generate an average  SELECT  DATEDIFF(year, [BirthDate], GETDATE()) as [Age]  , COUNT([EmployeeKey]) as [# Employees]  , SUM([SickLeaveHours]) as [# Hours Sick]  , SUM([SickLeaveHours]) / COUNT([EmployeeKey]) as [Avg.]  FROM [dbo].[DimEmployee]  GROUP BY DATEDIFF(year, [BirthDate], GETDATE())  ORDER BY [Age] | Here is the data that can create the chart below. |



The chart on the right shows totals (sums) of sick leave hours used by age.

Using this chart would be a rookie mistake. Before scrolling down generate some ideas how to better analyze this data.

You are the analyst, and therefore need to summon up equal parts common sense and technical skill

|  |  |
| --- | --- |
| Here the analyst uses an average but still cannot understand the problem, because the data is too granular  Next after further discussion go ahead and put the data into groups to reduce its granularity to better see any patterns. |  |

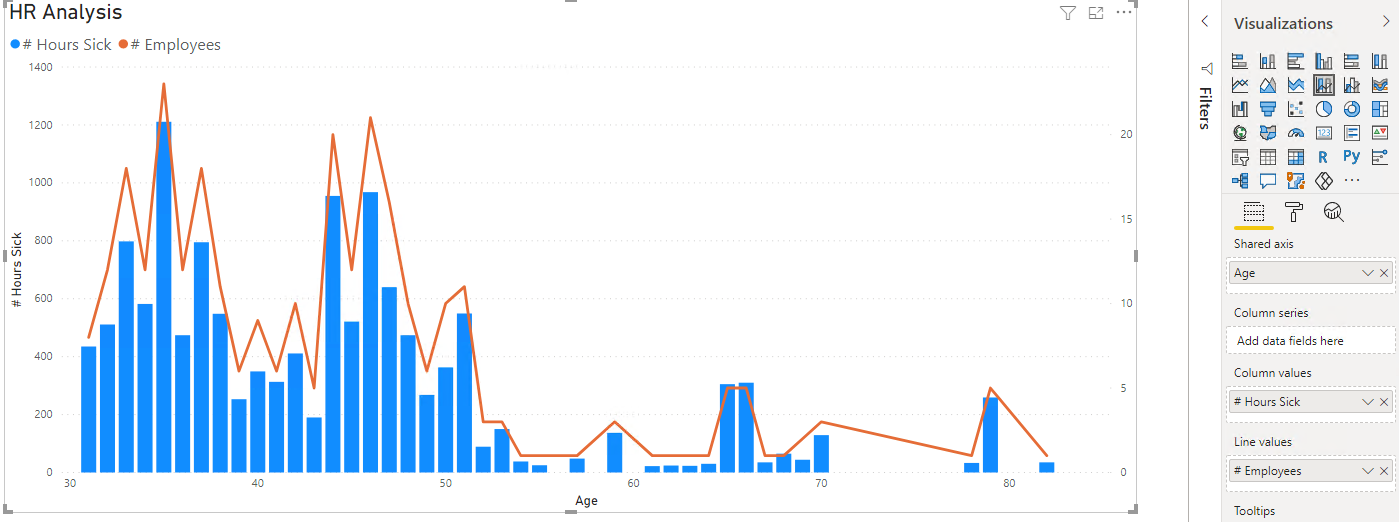
Below shows an improvement – a combo chart.

The orange line below shows the count of the number of employees. You can easily see now that the rise and fall in the number of sick leave hours used by employees of a certain again is very similar to the number of employees at each age.

See? So the rise and fall in the total number of sick leave hours used, is not based on age (not even a trend here), but rather just based on the number of employees in the company at each age.

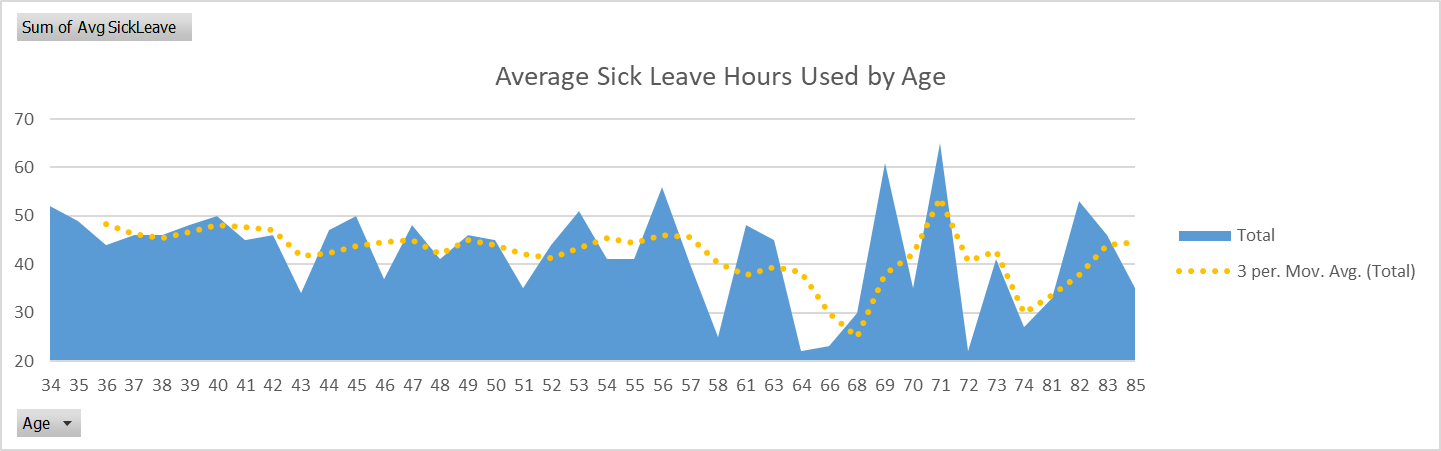
The analyst’s mantra ….. How can I best understand then explain the phenomenon using charts and analytics?

So if the number of hours sick leave are based on the number of employees in each category (here age) then you need to average the sick leave used by age.



Now you can see that the number of Total sick leave hours does spike to 1211 for 35 year olds but that’s because there are 24 employees aged 39. The average sick leave hours used by employee is at a steady amount 44-52 range.

|  |  |
| --- | --- |
| This final chart may show some insight.  Insight and recommendations for improvement is what you are getting paid to generate  How to improve this analysis folks? Reduce the granularity to perhaps 5 or 10 year age buckets.  Or add a moving average as shown below in Excel |  |



Now you can see that the average sick leave hours used is relatively stable for employees of each age group. So the learning here is that you can’t only focus on frequency of an event occurring or total amount of a dimension occurring. If you average out the data to take into effect sample size, then you can see a truer different picture.

*Similarly you could claim that a certain crime happens in California too much, however if you average out the frequency of the crime, with the population of 39 million (as of 2019) then a different story is told, the event is not that common if you take into account the size of the state.*

So try not to ever act like a rookie, even if you have 20 years experience. Look for discordant data, rather than self-conforming data.

SQL Code: The remaining content shows two examples of CASE statements. While one example is enough to get this skill transferred to you, it seems that to make the employee scenario more authentic both the age of the employee and how many years they worked at the company are used in analysis to look for trends to understand business patterns.

Each CASE statement creates one new column of data and inserts some words into that column ***for every row in the dataset produced. Later we will also transfer this newly generated table of data with its new columns of descriptors and measures to a new SQL database table.*** It’s as easy as the code in the box on the left. You must be an admin on the database to do this task.  
  
So using 2 CASE structures, next we create 2 new columns of text with categorical terms for   
a) how old the employee is chronologically in years, and   
b) how many years the employee has been working at Adventureworks. Different business performance outcomes can be examined to identify if there is systematic variance in the business performance based on the grouping made (Here employee age or years at the company). Forming groups and looking for group differences is very similar to statistical ANOVA (analysis of variance amongst groups means).

USE DatabaseName;

SELECT fieldnames

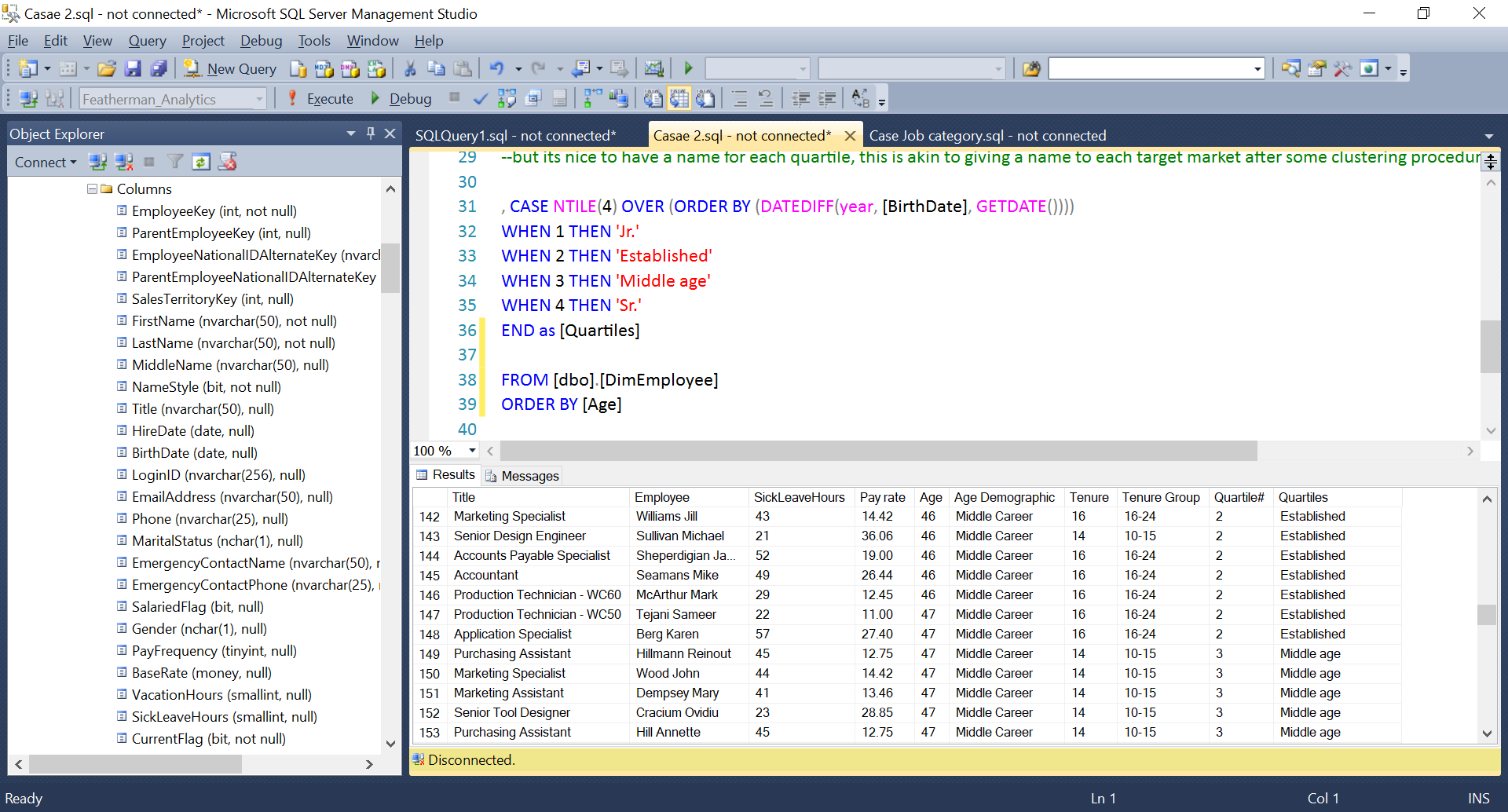
INTO DatabaseName.[dbo].[newTaleName]

FROM [dbo].[oldTablename] as O  
INNER JOIN [dob].[ReallyOldTableName] as R  
ON R.ID = O.ID

CASE statements can be used to put different types of data into useful groups, such as in the classic market segmentation based on behaviors, or demographics. NTILE is also shown below and is useful to put records into equal sized groups quickly such as generating quartiles and assigning sales representatives to quartiles (top 25%, bottom 25% etc.) based on performance.

|  |  |
| --- | --- |
| USE [Featherman\_Analytics]; | When starting a new query specify the database to work with by the USE statement |
| SELECT [Title], CONCAT( [LastName], ' ', [FirstName]) as [Employee] | Now select the title column and merge the first and last names using the CONCAT function. Set the column header to Employee |
| , [SickLeaveHours], FORMAT([BaseRate], 'N2') as [Pay rate] | Retrieve 2 fields of numbers, formatting one of then to only have 2 decimal places. |
| , DATEDIFF(year, [BirthDate], GETDATE()) as Age | Now this is the formula that calculates the age of the employee. DATEDIFF also works in Excel. We specify that we want the number of years between the value in the birthdate column, and today. |
| , DATEDIFF(year, [HireDate], GETDATE()) as Tenure | Calculate the # years employees worked for AW |
| , CASE  WHEN DATEDIFF(year, [HireDate], GETDATE()) < 10 THEN '<10'  WHEN DATEDIFF(year, [HireDate], GETDATE()) BETWEEN 10 and 15 THEN '10-15'  WHEN DATEDIFF(year, [HireDate], GETDATE()) BETWEEN 16 and 24 THEN '16-24'  WHEN DATEDIFF(year, [HireDate], GETDATE()) >= 25 THEN 'Silverback'  END AS [Tenure Group] | Here we put employees into groups based on their # years with Adventureworks. These are not equal sized groups. This tenure is just with AdventureWorks and can be quite different than the age of the employee, as often employees started their careers elsewhere. Managers in particular are often hired from other companies. |
| , NTILE(4) OVER  (ORDER BY (DATEDIFF(year, [BirthDate], GETDATE() )) ) as [Quartile#] | If you do not need custom ranges, then equally sized groups might be a good enough criteria to form group membership. Here 4 quartiles using NTILE |
| , CASE NTILE(4) OVER (ORDER BY (DATEDIFF(year, [BirthDate], GETDATE() )) )  WHEN 1 THEN 'Jr.'  SELECT [Quartiles], [Quartile#]  , COUNT([Quartile#]) as [#]  , AVG([Age]) as [Quartile Avg. Age]  FROM [dbo].[dimEmployee2]  GROUP BY [Quartiles], [Quartile#]  ORDER BY [Quartile#]  WHEN 2 THEN 'Established'  WHEN 3 THEN 'Middle age'  WHEN 4 THEN 'Sr.'  END  as [Quartiles] | Here we use a CASE statement to assign a name to each quartile (1,2,3,4).  The select statement to the left is used to verify that the quartiles are of equal size (74 employees each).  Look at the quartiles to see the average age of each quartile. They are not evenly spread out, which is just based on having less variance in age at one end of the spectrum (ie between Jr. and the established group) and more spread in the age of employees that are senior level. |

|  |  |
| --- | --- |
| INTO [yourDatabaseName].[dbo].[tableName] | If you are the admin on your database then you can create new tables of data using a term like this |
| FROM [dbo].[DimEmployee]  ORDER BY [Age] | Here is the table that the data was extracted from. We sort the records by age. |



Here are the results that are returned from the query.

One more example can give more ideas for the development of your analytics skills. Certainly your exposure to DATEDIFF, NTILE and CASE are substantial already but lets push the analytics to bring in a table of measures, so we can use the groups we made.

|  |  |
| --- | --- |
| USE [AdventureWorksDW2012];  SELECT DISTINCT([Title])  , COUNT([LastName]) as [Count for Age]  , Case  WHEN [Title] = 'Accountant' THEN 'Acc'  WHEN [Title] LIKE '%Fin%' THEN 'Fin'  WHEN [Title] LIKE '%Specialist%' THEN 'Admin'  WHEN [Title] LIKE '%Design%' THEN 'Design'  WHEN [Title] LIKE '%Data%' OR [Title] LIKE '%Info%' OR [Title] LIKE '%Network%' THEN 'IT'  WHEN [Title] Like '%Human%' OR [Title] LIKE '%Benefits%' OR [Title] LIKE '%Recruiter%' THEN 'HR'  WHEN [Title] LIKE '%Marketing%' THEN 'Mktg'  WHEN [Title] LIKE '%Schedul%' OR [Title] LIKE '%Buyer%' OR [Title] LIKE '%Purchasing%'   OR [Title] LIKE '%Control%'THEN 'Logistics'  WHEN [Title] LIKE '%Quality%' OR [Title] LIKE '%Document%' THEN 'Quality'  WHEN [Title] LIKE '%Sales%' OR [Title] LIKE '%Accounts Manager%' THEN 'Sales'  WHEN [Title] Like '%Jan%' OR [Title] LIKE '%Maint%' OR [Title] LIKE '%Tool' OR [Title] LIKE '%Facilities%' THEN 'HR'  WHEN [Title] LIKE '%Prod% Sup%' THEN 'Prod. Supervisor'  WHEN [Title] LIKE '%Prod% Tech%' THEN 'Prod. Technician'  WHEN [Title] LIKE '%Shipping%' OR [Title] Like '%Stock%' THEN 'Inventory'  WHEN [Title] LIKE '%Manager%' OR [Title] LIKE '%Vice%' OR [Title] LIKE '%Chief%' THEN 'Mgmt'  WHEN [Title] LIKE '%Research%' THEN 'Research'  END  AS [Job Category]  , FORMAT(AVG([BaseRate]), 'N2') as [Avg. Rate]  INTO [Featherman\_Analytics].[dbo].[dimEmployeeCondensedJobCategory]  FROM [dbo].[DimEmployee]  GROUP BY [Title] | You can copy this code into SSMS query and recreate this data condensing project, creating new labor categories.  There are many different job titles that contain the word Financial. These can be condensed into one department (option).  Here we calculate the average payrate for each of the job categories we just created.  Also nice to know we have 9 buyers. Notice that different job titles can have the same Job Category.  The average hourly rate by job category is an important HR metric.  The above query was used to make a new database table. If we query this new condensed data, then we can use a GROUP BY query to change the granularity of the report.  The granularity has changed from employee level (N= 300) to job title level, (N=187) to job category (N=15). The code isn’t really hard, it just takes time and attention to spelling. |

|  |  |
| --- | --- |
| USE [AdventureWorksDW2012];  SELECT [Title], [SalesTerritoryCountry] as [Country]  , [SalesTerritoryRegion] as [Region], CONCAT( [LastName], ' ', [FirstName]) as [Employee]  /\*we build a personel file for the sales employees. Putting them into quartiles for sales performance based  on a) # Units moved, b) total revenue. \*/  , case --here we put the employees into an HR age group.  WHEN DATEDIFF(year, [BirthDate], GETDATE()) < 28 THEN '1. Jr. Associate'  WHEN DATEDIFF(year, [BirthDate], GETDATE()) BETWEEN 29 and 39 THEN '2. Emerging Leader'  WHEN DATEDIFF(year, [BirthDate], GETDATE()) BETWEEN 40 and 50 THEN '3. Mid-Career'  WHEN DATEDIFF(year, [BirthDate], GETDATE()) BETWEEN 51 and 59 THEN '4. Sr. Associate'  WHEN DATEDIFF(year, [BirthDate], GETDATE()) > 59 THEN '5. Partner'  END AS [Age Demographic]  -- Now lets see how many years the employees has worked for AW, and their age.  , DATEDIFF(year, [HireDate], GETDATE()) as Tenure  , DATEDIFF(year, [BirthDate], GETDATE()) as Age  --here are the metrics derived from the sales table, # orders, # sku's, # Units, and Revenue  , COUNT(DISTINCT([SalesOrderNumber])) as [# Orders]  , COUNT([SalesOrderLineNumber]) as [# SKU's]  , SUM([OrderQuantity]) as [Total Units]  , NTILE(4) OVER (ORDER BY (SUM([OrderQuantity]) )) as [Units quartile]  , FORMAT(SUM([SalesAmount])/ COUNT(DISTINCT([SalesOrderNumber])), 'N0') as [Avg. $ale]  *Sorry this query NEVER should have used the DimSalesTerritory table.  Better to use dimGeography and dimReller connection*  , FORMAT(SUM([SalesAmount]),'N0') as [Total Revenue]  , NTILE(4) OVER (ORDER BY (SUM([OrderQuantity]) )) as [Revenue quartile]  FROM [dbo].[DimEmployee] as e  INNER JOIN [dbo].[FactResellerSales] as s ON e.[EmployeeKey] =s.[EmployeeKey]  INNER JOIN [dbo].[DimSalesTerritory] as t ON t.[SalesTerritoryKey] = e.[SalesTerritoryKey]  WHERE [Title] LIKE '%sales %'  --here are all the dimensions we are using to specify the unit of analysis. The column with the highest granularity such as Employee  --below will have the most variety of different data values in teh column. That dictates the number of rows generated in the GROUP BY  GROUP BY [Title], CONCAT( [LastName], ' ', [FirstName]), [SalesTerritoryCountry], [SalesTerritoryRegion]  , DATEDIFF(year, [BirthDate], GETDATE())  , DATEDIFF(year, [HireDate], GETDATE())  ORDER BY [Title], [SalesTerritoryCountry], [SalesTerritoryRegion] | This is the final demonstration in this module. By joining to the ResellerSales table and the Saleserritory Table, we can make a very detailed and insightful table because we can add some colunms of metrics.  Quartile categorization is useful for job evaluations. Its non-biased, but is only one metric of performance.  The dimensions must be in the GROUP BY line of code. |
| This is the output from the query. Go ahead and run the same query to recreate the results or similar. It’s not really that hard, after you understand where the columns come from. Just be careful to look for typos and red squiggly lines. | You have to admit this is an impressive report, just using the tools now in your SQL toolbox:  NTILE() GROUP BY()  INNER JOIN ON DATEDIFF() ORDER BY  COUNT () SUM () FORMAT () CONCAT () |
|  | This pivot chart becomes easy to generate. |
| The sales reps can be evaluated by the number of units they push and the average sale amount. | By coincidence there are 5 sales reps that are 50 years old.  Its good to check for root cause of variance in the data, and to continue to create new columns and aggregations until you get it right. |
| Looks like the UK sales rep sells a lot of units but the average invoice amount is low, similar to the central and nowrtheast regions of the US.  In the Northwest, and to a less extent Germany and Australia, the average sale amount s much higher indicating more expensive product line is being sold.  Below we have to add a PowerBI map since we have country as a field in our dataset. | There are so many business forces that effect the sales performance of any company in a region. If you have knowledgeable sales reps, then look to other factors such as macro-economic and price-point of competitors rather than focus on salesrep performance. |
|  | Before the granularity of analysis was region.  Now the granularity is changes to Country, and you can see that the US market is much larger than the others.  I hope you enjoy the content and it inspires you to experiment more!  Mauricio |