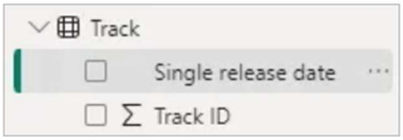
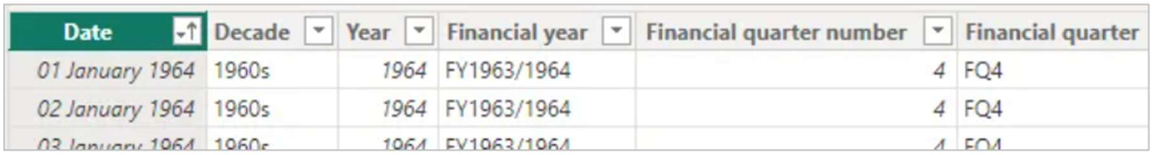
Question 1

Open the **Music Tours - Time Intelligence and Custom Calendars.pbix** file in the folder shown above. This report has the automatic calendar feature disabled.



Date fields in this report don't have an automatic calendar.

Import the **Track calendar** worksheet from the **Track calendar.xlsx** Excel file. This calendar assumes the financial year ends on the 31st of March.



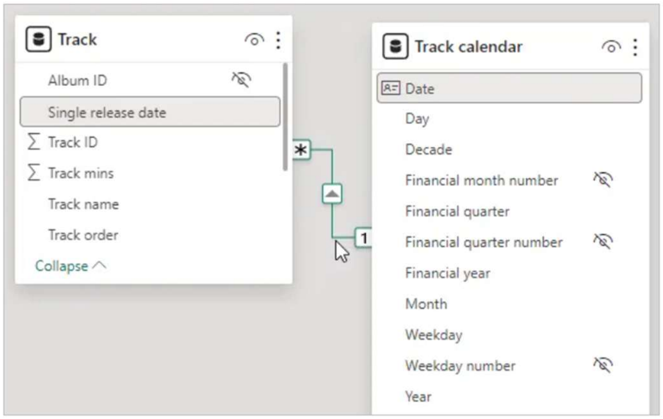
Date	Decade	Year	Financial year	Financial quarter number	Financial quarter
01 January 1964	1960s	1964	FY1963/1964	4	FQ4
02 January 1964	1960s	1964	FY1963/1964	4	FQ4
03 January 1964	1960s	1964	FY1963/1964	4	FQ4

The table has several columns related to financial periods.

Configure this table so that you can use it as a calendar table. Here are some things you might need to do:

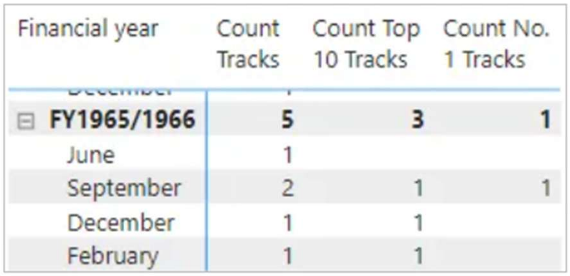
- Mark the table as a date table.
- Disable any automatic summing of numeric columns.
- Sort the **Month** and **Weekday** columns by appropriate columns.
- Hide columns from the report view.
- Create a relationship between the **Track** and **Track calendar** tables.

After completing the above tasks your table should resemble the one shown below:



The finished calendar table.

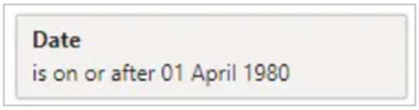
Use the **Track calendar** to create the following matrix visual:



Financial year	Count Tracks	Count Top 10 Tracks	Count No. 1 Tracks
⊟ **FY1965/1966**	**5**	**3**	**1**
June	1		
September	2	1	1
December	1	1	
February	1	1	

*You'll find measures in the **Music Measures** table to help you with this.*

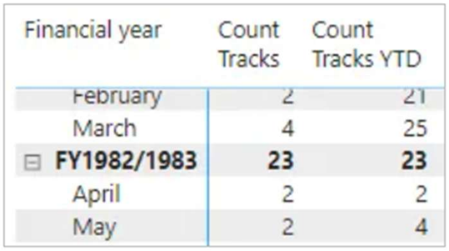
Apply a filter to the matrix visual so that it only shows dates from **1 April 1980**.



Date
is on or after 01 April 1980

*You'll need to add a field to the **Filters on this visual** section of the **Filter** pane to do this.*

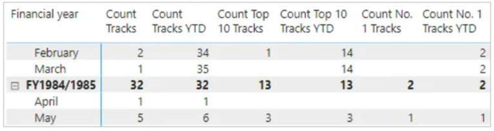
Create a measure which shows a year-to-date count of tracks and add this to the matrix.



Financial year	Count Tracks	Count Tracks YTD
February	2	21
March	4	25
⊟ **FY1982/1983**	**23**	**23**
April	2	2
May	2	4

Don't forget to reset the count on the end date of the financial year.

Calculate year-to-date counts for the other two measures in the matrix.



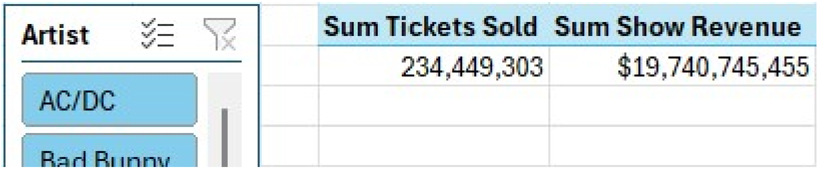
Financial year	Count Tracks	Count Tracks YTD	Count Top 10 Tracks	Count Top 10 Tracks YTD	Count No. 1 Tracks	Count No. 1 Tracks YTD
February	2	34	1	14		2
March	1	35		14		2
⊟ **FY1984/1985**	**32**	**32**	**13**	**13**	**2**	**2**
April	1	1				
May	5	6	3	3	1	1

The final matrix should resemble this.

Save and close the report.

Question 2

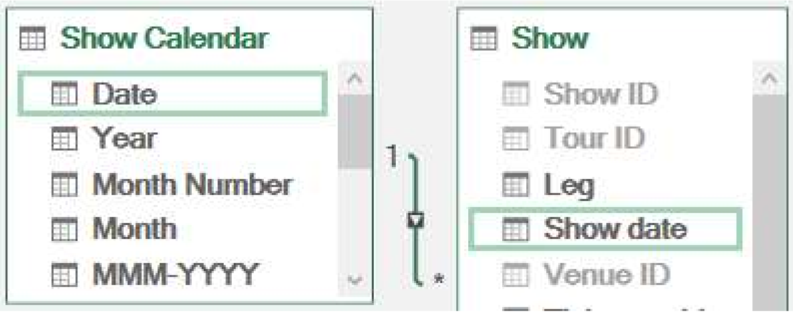
Open the **Music Tours Basic Time Intelligence.xlsx** file in the folder shown above.
On **Sheet1** you'll find a pivot table displaying a couple of measures.



Sum Tickets Sold	Sum Show Revenue
234,449,303	$19,740,745,455

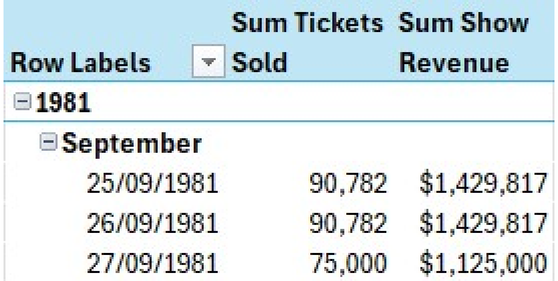
We'd like to analyse these values by year and month.

In Power Pivot, choose **Design | Date Table | New** to create a new calendar table. Rename this table as **Show Calendar** and create a relationship to the **Show** table.



*Connect the **Date** column to the **Show date** column.*

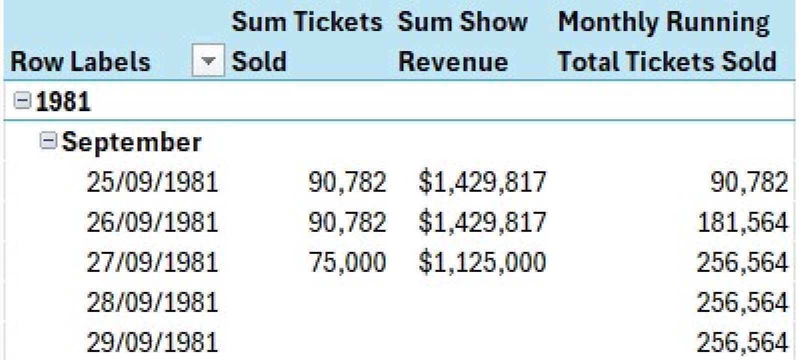
Add the **Year**, **Month** and **Date** fields to the pivot table on **Sheet1**.



Row Labels	Sum Tickets Sold	Sum Show Revenue
⊟1981		
⊟September		
25/09/1981	90,782	$1,429,817
26/09/1981	90,782	$1,429,817
27/09/1981	75,000	$1,125,000

The table is now divided by dates.

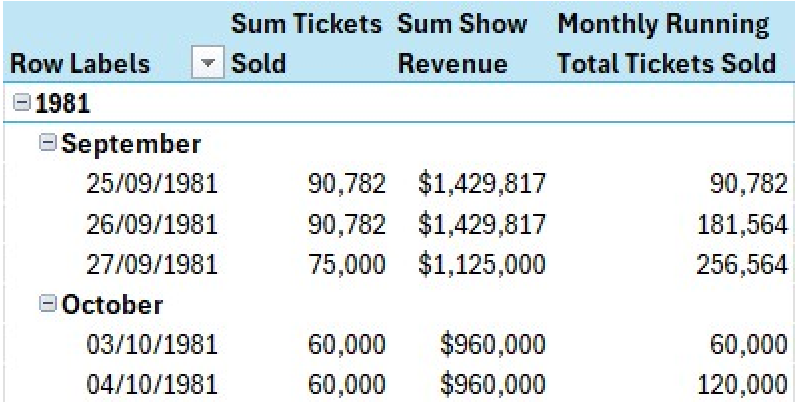
Create a measure using the **TOTALMTD** function to calculate a monthly running total of tickets sold. Add the measure to the pivot table.



Row Labels	Sum Tickets Sold	Sum Show Revenue	Monthly Running Total Tickets Sold
⊟1981			
⊟September			
25/09/1981	90,782	$1,429,817	90,782
26/09/1981	90,782	$1,429,817	181,564
27/09/1981	75,000	$1,125,000	256,564
28/09/1981			256,564
29/09/1981			256,564

Annoyingly, the running total makes dates on which no tickets were sold appear in the table.

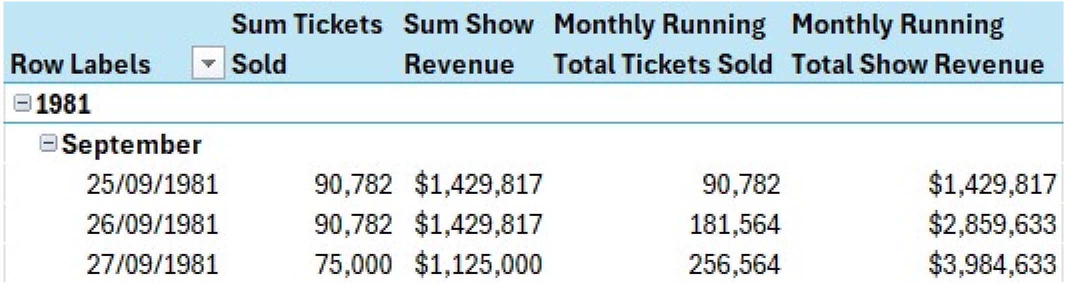
To hide the dates on which no tickets were sold, update the measure to include an **IF** function which checks if the sum of tickets sold is blank.



Row Labels ▾	Sum Tickets Sold	Sum Show Revenue	Monthly Running Total Tickets Sold
⊟1981			
⊟September			
25/09/1981	90,782	$1,429,817	90,782
26/09/1981	90,782	$1,429,817	181,564
27/09/1981	75,000	$1,125,000	256,564
⊟October			
03/10/1981	60,000	$960,000	60,000
04/10/1981	60,000	$960,000	120,000

The table should now hide any dates with no ticket sales.

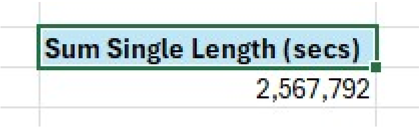
Add another measure to calculate a monthly running total of show revenue. Add this measure to the pivot table and make sure that it doesn't cause dates with no ticket sales to appear.



Row Labels ▾	Sum Tickets Sold	Sum Show Revenue	Monthly Running Total Tickets Sold	Monthly Running Total Show Revenue
⊟1981				
⊟September				
25/09/1981	90,782	$1,429,817	90,782	$1,429,817
26/09/1981	90,782	$1,429,817	181,564	$2,859,633
27/09/1981	75,000	$1,125,000	256,564	$3,984,633

Apply some formatting to the measure.

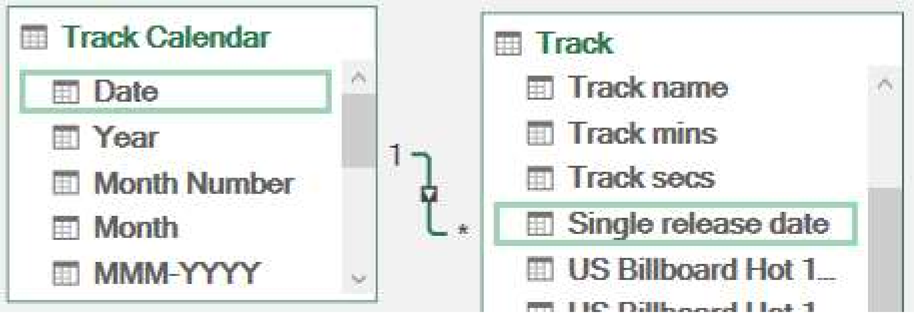
On **Sheet2** you'll find a pivot table showing the total length of tracks released as singles.



Sum Single Length (secs)
2,567,792

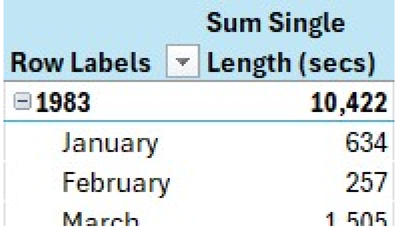
We'd like to group this measure by date.

Create a new calendar table called **Track calendar** and connect it to the **Track** table.



*Connect the **Date** column to the **Single release date** column.*

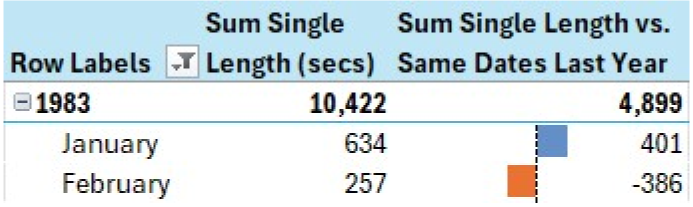
Add the **Year** and **Month** columns from the **Track Calendar** table to the pivot table on **Sheet2**. Use the **Row Labels** filter to exclude blanks and choose to show subtotals at the top of a group.



Row Labels	Sum Single Length (secs)
⊟1983	10,422
January	634
February	257
March	1,505

Each year should show a subtotal for the measure.

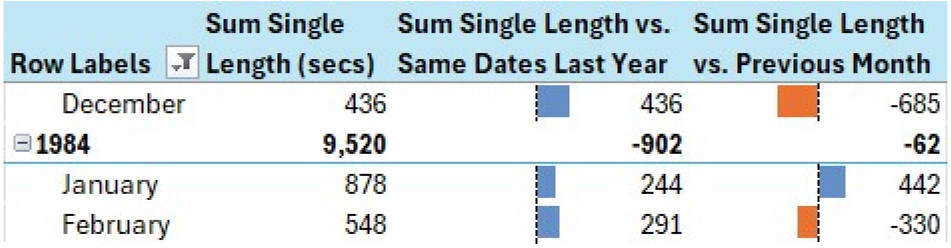
Create a measure which compares the sum of single length with the same period in the previous year. You can use a combination of the **CALCULATE** and **SAMEPERIODLASTYEAR** functions to do this. Add the measure to the pivot table.



Row Labels	Sum Single Length (secs)	Sum Single Length vs. Same Dates Last Year
⊟1983	10,422	4,899
January	634	401
February	257	-386

Add conditional formatting to highlight positive and negative numbers.

Add a measure which compares the sum of single length with the same value from one month ago. You can use the **CALCULATE** and **DATEADD** functions to do this. Add the measure to the pivot table.



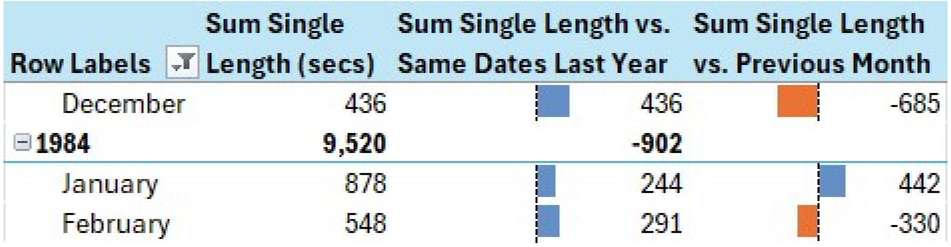
Row Labels	Sum Single Length (secs)	Sum Single Length vs. Same Dates Last Year	Sum Single Length vs. Previous Month
December	436	436	-685
⊟1984	**9,520**	**-902**	**-62**
January	878	244	442
February	548	291	-330

Conditional formatting might make it easier to read the results.

When comparing with the previous month, it doesn't make sense to display a value for the year. Try modifying the measure so that it only shows a result when the **Month** column is in scope.

*You can use the **ISINSCOPE** function to check if the **Month** column of the **Track Calendar** table is in scope.*

Check that the subtotals don't appear for years for this measure.



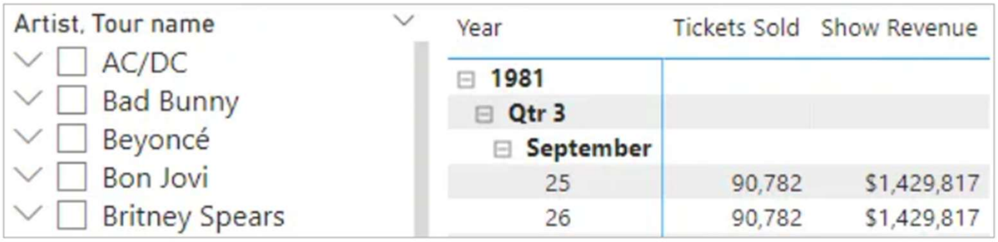
Row Labels	Sum Single Length (secs)	Sum Single Length vs. Same Dates Last Year	Sum Single Length vs. Previous Month
December	436	436	-685
⊟1984	**9,520**	**-902**	
January	878	244	442
February	548	291	-330

You should still see subtotals for the first two measures.

Save and close the file.

Question 3

Open the **Music Tours - Basic Time Intelligence.pbix** file in the folder shown above. On **Page 1** you'll find a matrix containing the dates of shows and the tickets sold and revenue generated.



Artist, Tour name

- □ AC/DC
- □ Bad Bunny
- □ Beyoncé
- □ Bon Jovi
- □ Britney Spears

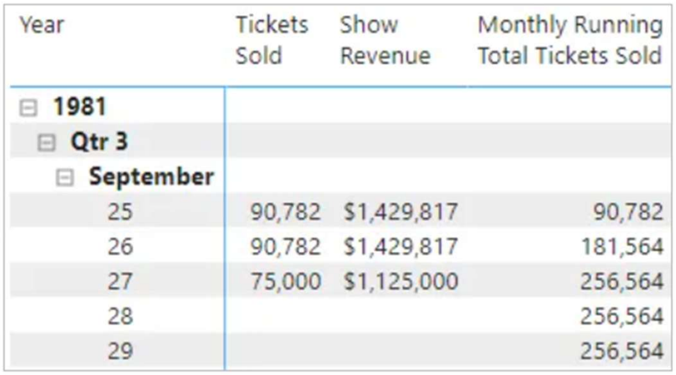
Year	Tickets Sold	Show Revenue
⊟ **1981**		
⊟ **Qtr 3**		
⊟ **September**		
25	90,782	$1,429,817
26	90,782	$1,429,817

You can use the slicer to choose different artists and tours.

Add a measure to create a monthly running total of tickets sold using the **TOTALMTD** function.

Remember to reference the ***Date*** *column of the automatic calendar table that is related to the* ***Show date*** *field, rather than just the* ***Show date*** *field itself.*

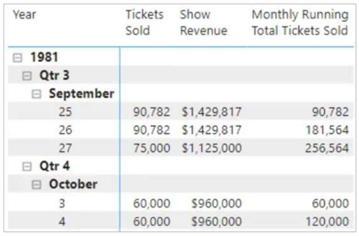
Add the measure to the matrix.



Year	Tickets Sold	Show Revenue	Monthly Running Total Tickets Sold
⊟ **1981**			
⊟ **Qtr 3**			
⊟ **September**			
25	90,782	$1,429,817	90,782
26	90,782	$1,429,817	181,564
27	75,000	$1,125,000	256,564
28			256,564
29			256,564

Annoyingly, the running total makes dates on which no tickets were sold appear in the matrix.

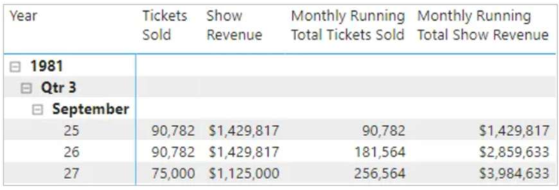
To hide the dates on which no tickets were sold, update the measure to include an **IF** function which checks if the sum of tickets sold is blank.



Year	Tickets Sold	Show Revenue	Monthly Running Total Tickets Sold
⊟ **1981**			
⊟ **Qtr 3**			
⊟ **September**			
25	90,782	$1,429,817	90,782
26	90,782	$1,429,817	181,564
27	75,000	$1,125,000	256,564
⊟ **Qtr 4**			
⊟ **October**			
3	60,000	$960,000	60,000
4	60,000	$960,000	120,000

The matrix should now hide any dates with no ticket sales.

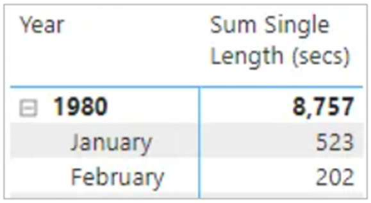
Add another measure to calculate a monthly running total of show revenue. Add this measure to the matrix and make sure that it doesn't cause dates with no ticket sales to appear.



Year	Tickets Sold	Show Revenue	Monthly Running Total Tickets Sold	Monthly Running Total Show Revenue
⊟ **1981**				
⊟ **Qtr 3**				
⊟ **September**				
25	90,782	$1,429,817	90,782	$1,429,817
26	90,782	$1,429,817	181,564	$2,859,633
27	75,000	$1,125,000	256,564	$3,984,633

You could apply some formatting to the measure.

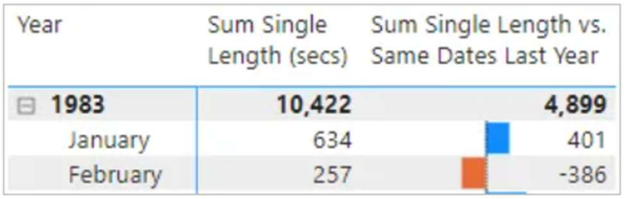
On **Page 2** you'll find a matrix showing the total length of tracks released as singles, grouped by year and month.



Year	Sum Single Length (secs)
⊟ **1980**	**8,757**
January	523
February	202

The matrix has been filtered to show singles released since 1980.

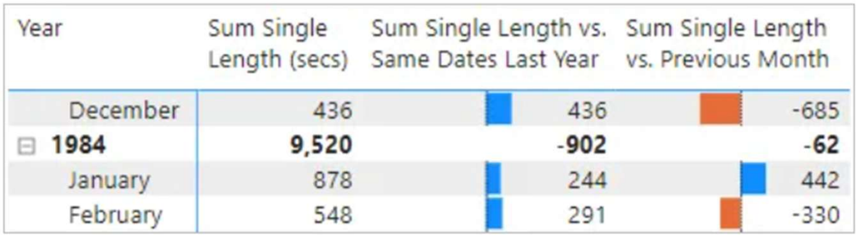
Create a measure which compares the sum of single length with the same period in the previous year. You can use a combination of the **CALCULATE** and **SAMEPERIODLASTYEAR** functions to do this. Add the measure to the matrix.



Year	Sum Single Length (secs)	Sum Single Length vs. Same Dates Last Year
⊟ **1983**	**10,422**	**4,899**
January	634	401
February	257	-386

You could add conditional formatting to highlight positive and negative numbers.

Add a measure which compares the sum of single length with the same value from one month ago. You can use the **CALCULATE** and **DATEADD** functions to do this. Add the measure to the matrix.



Year	Sum Single Length (secs)	Sum Single Length vs. Same Dates Last Year	Sum Single Length vs. Previous Month
December	436	436	-685
⊟ **1984**	**9,520**	**-902**	**-62**
January	878	244	442
February	548	291	-330

Again, conditional formatting might make it easier to read the results.

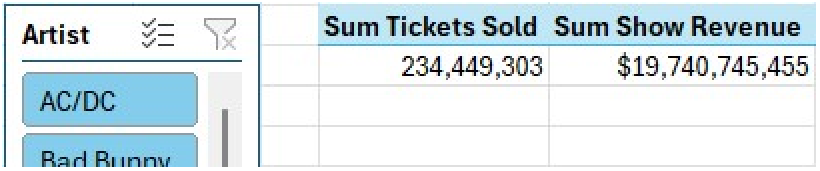
When comparing with the previous month, it doesn't make sense to display a value for the year. Try modifying the measure so that it only shows a result when the **Month** column is in scope.

*You can use the **ISINSCOPE** function to check if the **Month** column of the **Single release date** field is in scope.*

Save and close the report.

Question 2

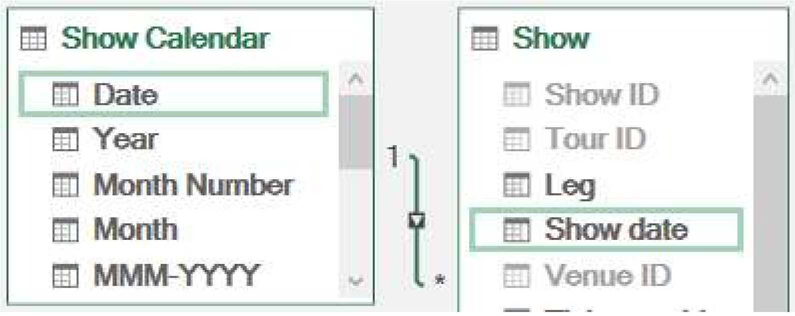
Open the **Music Tours Basic Time Intelligence.xlsx** file in the folder shown above. On **Sheet1** you'll find a pivot table displaying a couple of measures.



Sum Tickets Sold	Sum Show Revenue
234,449,303	$19,740,745,455

We'd like to analyse these values by year and month.

In Power Pivot, choose **Design | Date Table | New** to create a new calendar table. Rename this table as **Show Calendar** and create a relationship to the **Show** table.



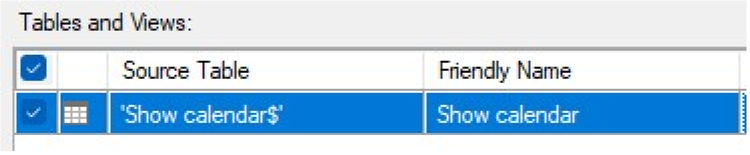
*Connect the **Date** column to the **Show date** column.*

Add the **Year**, **Month** and **Date** fields to the pivot table on **Sheet1**.

Question 4

Open the **Music Tours Custom Calendars.xlsx** file in the folder shown above.

Import the **Show calendar** worksheet from the Excel file called **Show calendar.xlsx** into the Power Pivot data model.

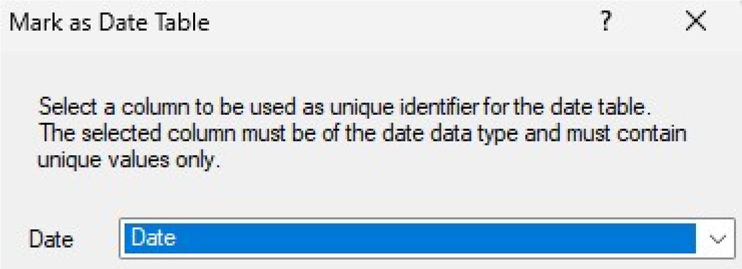


Tables and Views:

☑		Source Table	Friendly Name
☑	▦	'Show calendar$'	Show calendar

Import this worksheet.

Mark the newly imported table as a date table, using the **Date** column as the key.



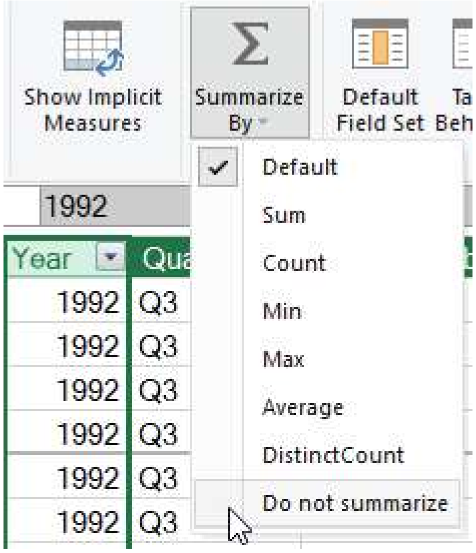
Mark as Date Table

Select a column to be used as unique identifier for the date table. The selected column must be of the date data type and must contain unique values only.

Date: Date

*Use the **Date** column as the unique identifier for the table.*

Set the default summarization of any numeric columns in the table to **Do not summarize**.



*Power Pivot tries to sum values in numeric columns, like the **Year** column shown here. We don't want this to happen!*

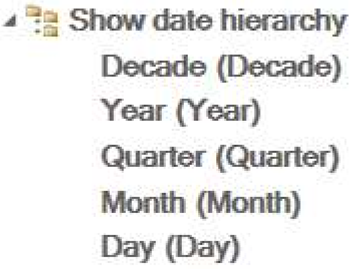
Change the **Sort by Column** setting of the columns shown in the table below:

Column	Column to sort by

Month	**Month number**
Weekday	**Weekday number**

Hide the **Month number** and **Weekday number** fields from client tools so that they won't appear when you're building pivot tables.

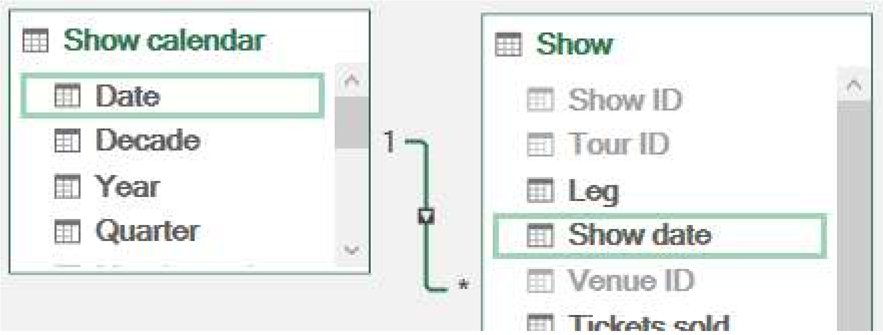
Create a hierarchy starting with the **Decade** column. Rename the hierarchy as **Show date hierarchy** and add the columns shown below to it:



Show date hierarchy
Decade (Decade)
Year (Year)
Quarter (Quarter)
Month (Month)
Day (Day)

*You can create a hierarchy in the **Diagram view** of Power Pivot.*

Create a relationship between the **Show calendar** and **Show** tables.



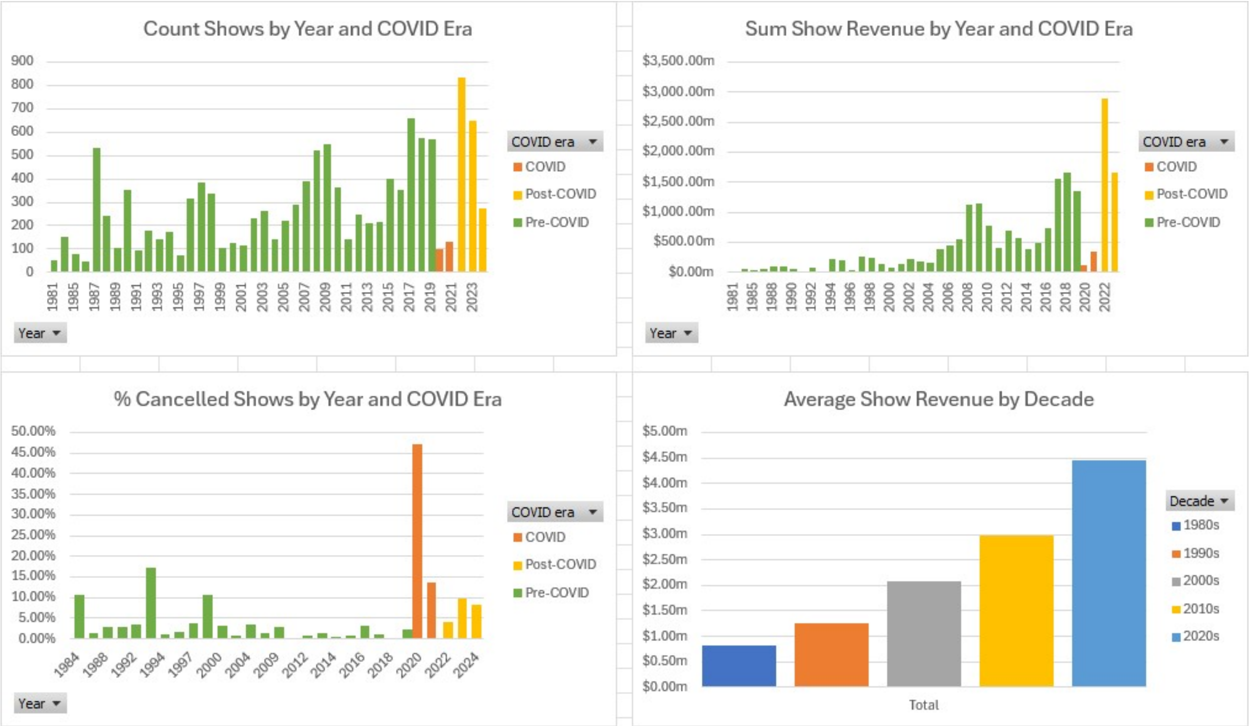
Show calendar
Date
Decade
Year
Quarter

1
*

Show
Show ID
Tour ID
Leg
Show date
Venue ID
Tickets sold

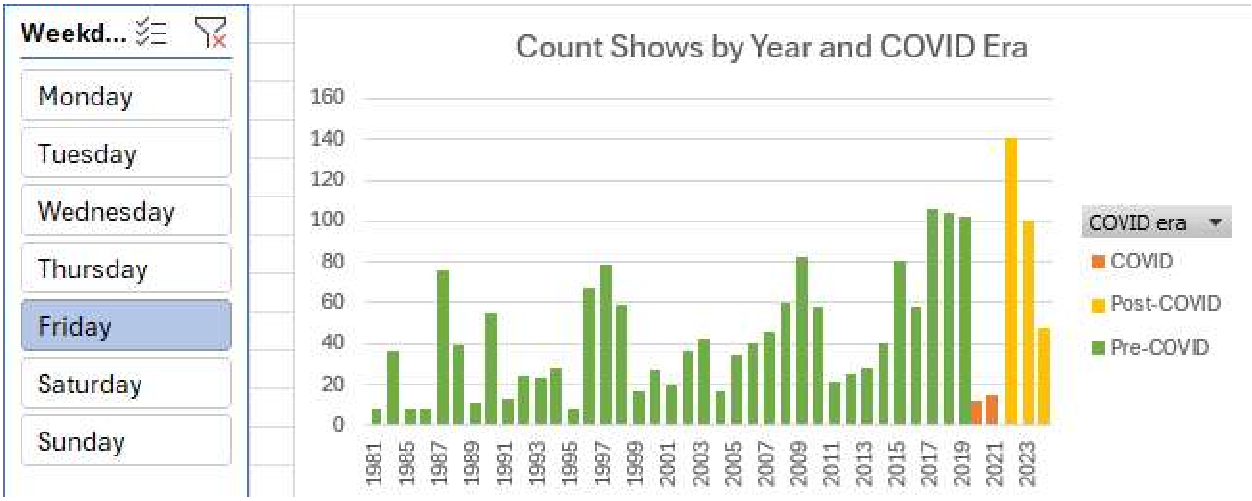
Create the relationship between the columns highlighted here.

Use your new calendar table to create the following pivot charts (you'll find some measures in the **Show** table to help you):



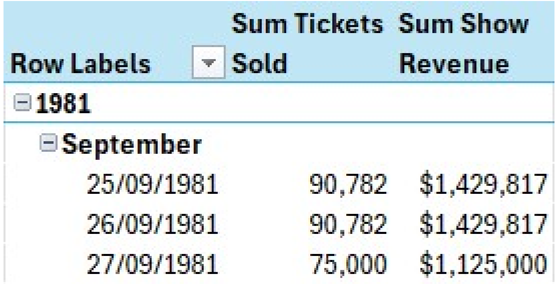
Change the formatting to suit your preferences.

Add a slicer which you can use to filter all the charts by the **Weekday** column.



Connect the slicer to each chart.

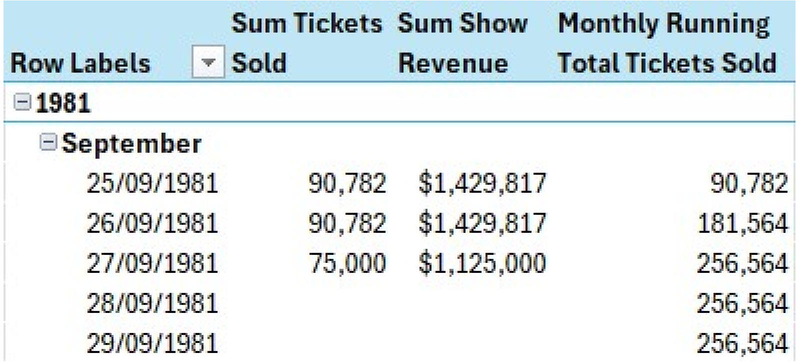
Save and close the file.



Row Labels	Sum Tickets Sold	Sum Show Revenue
⊟1981		
⊟September		
25/09/1981	90,782	$1,429,817
26/09/1981	90,782	$1,429,817
27/09/1981	75,000	$1,125,000

The table is now divided by dates.

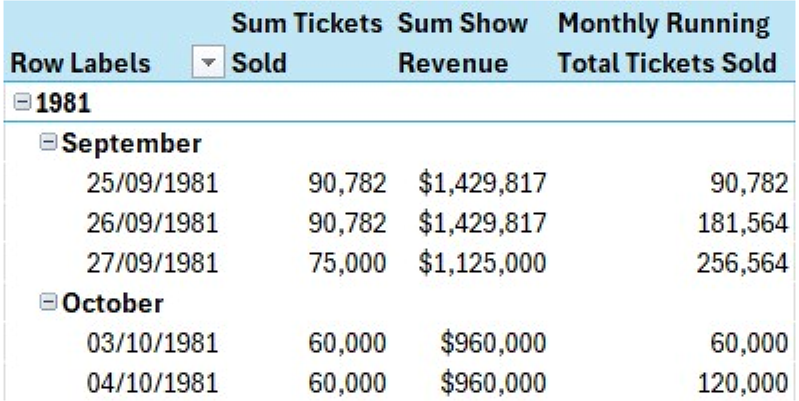
Create a measure using the **TOTALMTD** function to calculate a monthly running total of tickets sold. Add the measure to the pivot table.



Row Labels	Sum Tickets Sold	Sum Show Revenue	Monthly Running Total Tickets Sold
⊟1981			
⊟September			
25/09/1981	90,782	$1,429,817	90,782
26/09/1981	90,782	$1,429,817	181,564
27/09/1981	75,000	$1,125,000	256,564
28/09/1981			256,564
29/09/1981			256,564

Annoyingly, the running total makes dates on which no tickets were sold appear in the table.

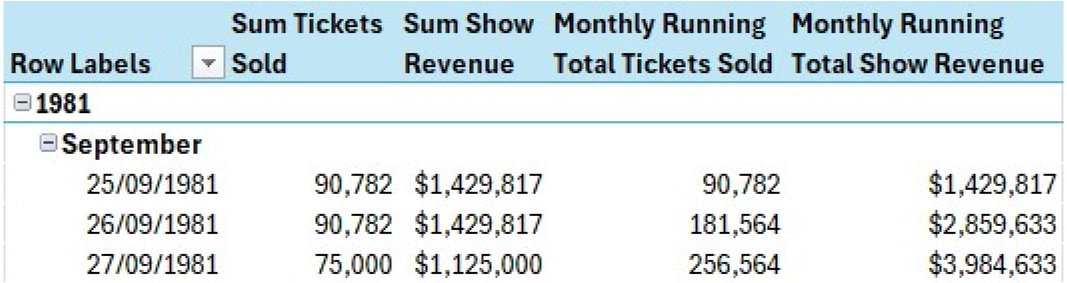
To hide the dates on which no tickets were sold, update the measure to include an **IF** function which checks if the sum of tickets sold is blank.



Row Labels	Sum Tickets Sold	Sum Show Revenue	Monthly Running Total Tickets Sold
⊟1981			
⊟September			
25/09/1981	90,782	$1,429,817	90,782
26/09/1981	90,782	$1,429,817	181,564
27/09/1981	75,000	$1,125,000	256,564
⊟October			
03/10/1981	60,000	$960,000	60,000
04/10/1981	60,000	$960,000	120,000

The table should now hide any dates with no ticket sales.

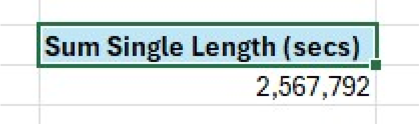
Add another measure to calculate a monthly running total of show revenue. Add this measure to the pivot table and make sure that it doesn't cause dates with no ticket sales to appear.



Row Labels	Sum Tickets Sold	Sum Show Revenue	Monthly Running Total Tickets Sold	Monthly Running Total Show Revenue
⊟1981				
⊟September				
25/09/1981	90,782	$1,429,817	90,782	$1,429,817
26/09/1981	90,782	$1,429,817	181,564	$2,859,633
27/09/1981	75,000	$1,125,000	256,564	$3,984,633

Apply some formatting to the measure.

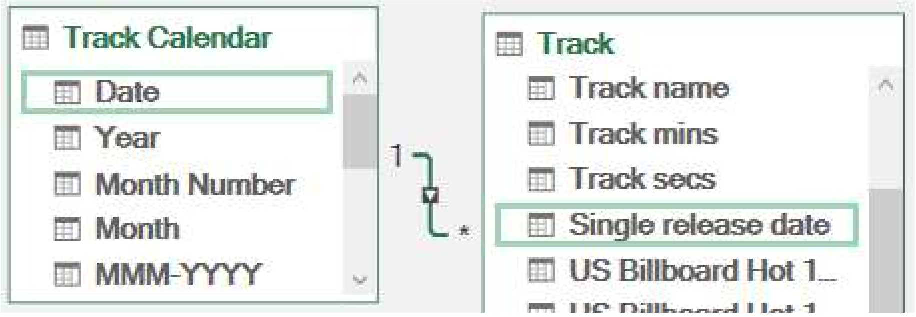
On **Sheet2** you'll find a pivot table showing the total length of tracks released as singles.



Sum Single Length (secs)
2,567,792

We'd like to group this measure by date.

Create a new calendar table called **Track calendar** and connect it to the **Track** table.

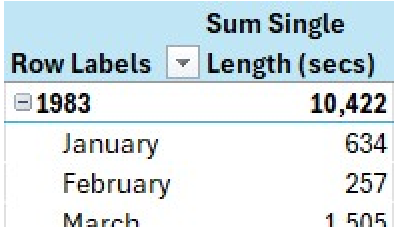


Track Calendar
Date
Year
Month Number
Month
MMM-YYYY

Track
Track name
Track mins
Track secs
Single release date
US Billboard Hot 1...

*Connect the **Date** column to the **Single release date** column.*

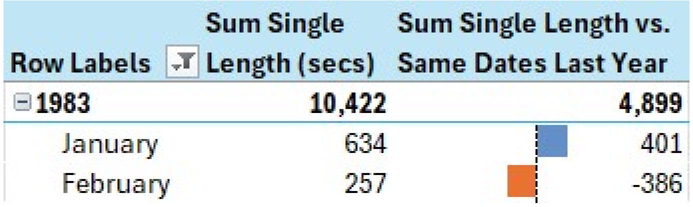
Add the **Year** and **Month** columns from the **Track Calendar** table to the pivot table on **Sheet2**. Use the **Row Labels** filter to exclude blanks and choose to show subtotals at the top of a group.



Row Labels	Sum Single Length (secs)
⊟1983	10,422
January	634
February	257
March	1,505

Each year should show a subtotal for the measure.

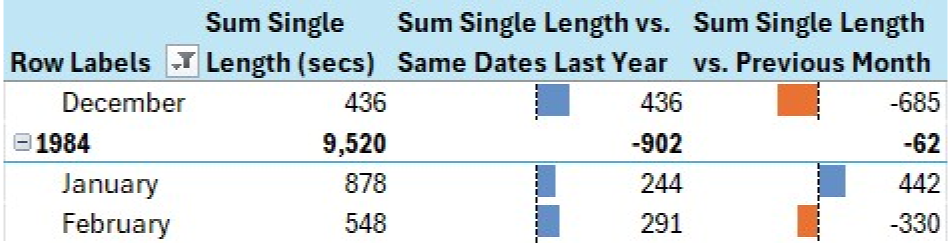
Create a measure which compares the sum of single length with the same period in the previous year. You can use a combination of the **CALCULATE** and **SAMEPERIODLASTYEAR** functions to do this. Add the measure to the pivot table.



Row Labels	Sum Single Length (secs)	Sum Single Length vs. Same Dates Last Year
⊟**1983**	**10,422**	**4,899**
January	634	401
February	257	-386

Add conditional formatting to highlight positive and negative numbers.

Add a measure which compares the sum of single length with the same value from one month ago. You can use the **CALCULATE** and **DATEADD** functions to do this. Add the measure to the pivot table.



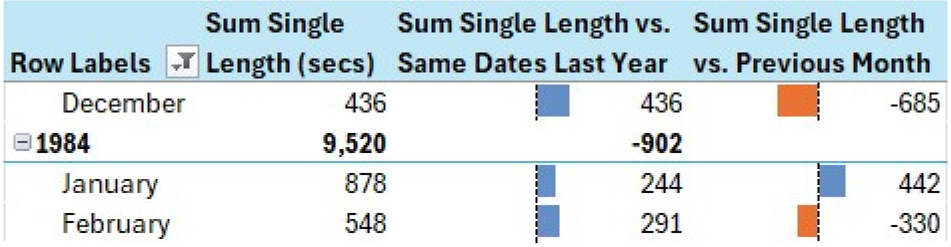
Row Labels	Sum Single Length (secs)	Sum Single Length vs. Same Dates Last Year	Sum Single Length vs. Previous Month
December	436	436	-685
⊟**1984**	**9,520**	**-902**	**-62**
January	878	244	442
February	548	291	-330

Conditional formatting might make it easier to read the results.

When comparing with the previous month, it doesn't make sense to display a value for the year. Try modifying the measure so that it only shows a result when the **Month** column is in scope.

*You can use the **ISINSCOPE** function to check if the **Month** column of the **Track Calendar** table is in scope.*

Check that the subtotals don't appear for years for this measure.



Row Labels	Sum Single Length (secs)	Sum Single Length vs. Same Dates Last Year	Sum Single Length vs. Previous Month
December	436	436	-685
⊟**1984**	**9,520**	**-902**	
January	878	244	442
February	548	291	-330

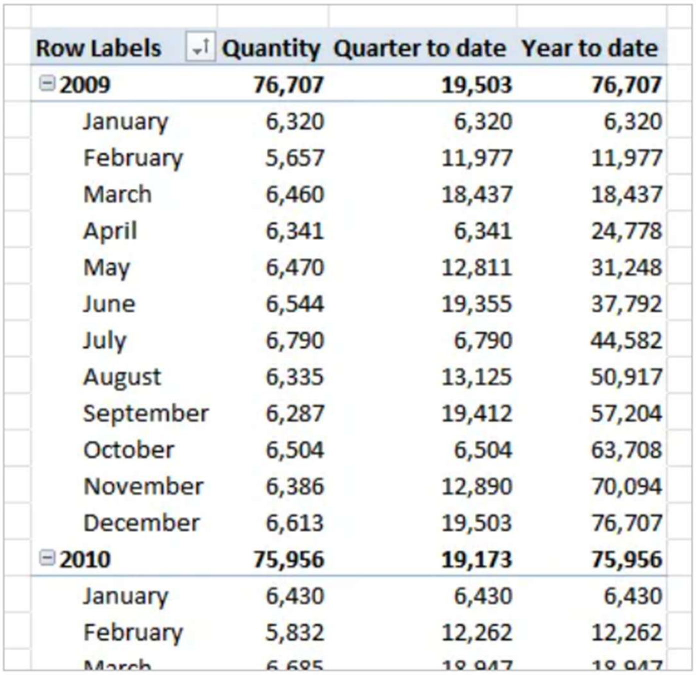
You should still see subtotals for the first two measures.

Save and close the file.

Question 5

o start, if you haven't already done so run the script in the above folder to generate the **MAM** database (not for commercial use or copying).

Create quarter-to-date and year-to-date total quantity sold measures using the **TOTALQTD** and **TOTALYTD** functions:



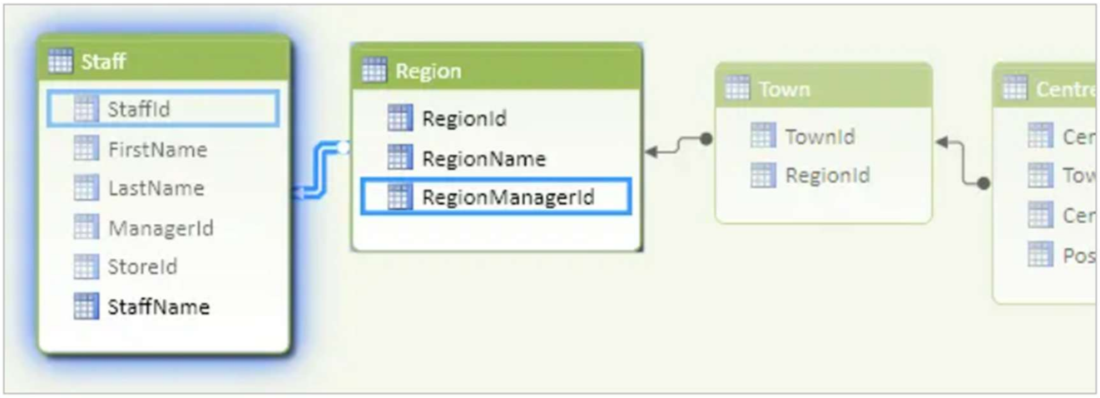
Row Labels	Quantity	Quarter to date	Year to date
⊟2009	**76,707**	**19,503**	**76,707**
January	6,320	6,320	6,320
February	5,657	11,977	11,977
March	6,460	18,437	18,437
April	6,341	6,341	24,778
May	6,470	12,811	31,248
June	6,544	19,355	37,792
July	6,790	6,790	44,582
August	6,335	13,125	50,917
September	6,287	19,412	57,204
October	6,504	6,504	63,708
November	6,386	12,890	70,094
December	6,613	19,503	76,707
⊟2010	**75,956**	**19,173**	**75,956**
January	6,430	6,430	6,430
February	5,832	12,262	12,262
March	6,685	18,947	18,947

*You'll need to import the **tblPos**, **tblTransaction** and **tblCalendar** tables.*

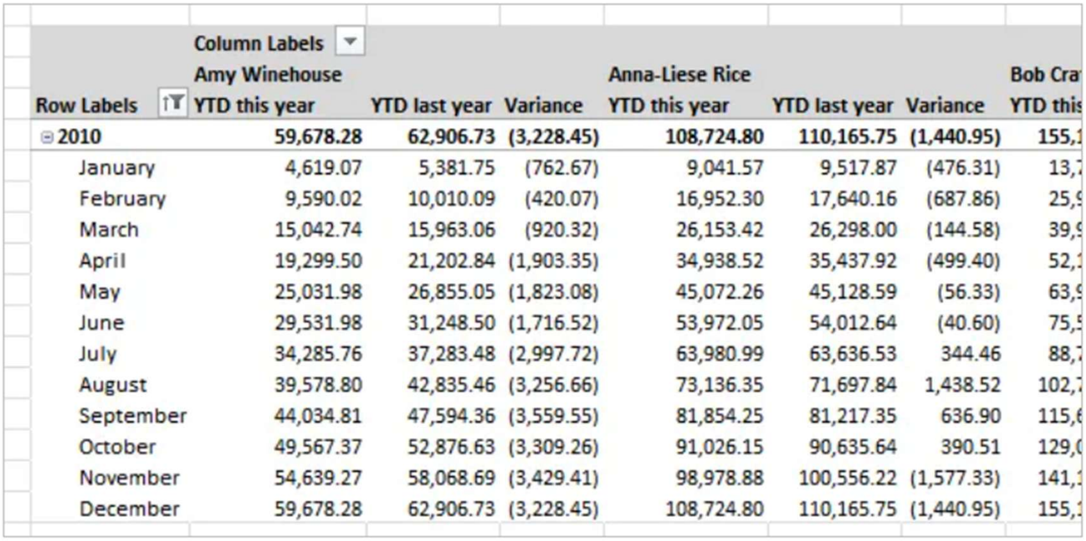
Save this workbook as **Years and quarters**, then close it down.

Question 6

Good luck! To start, if you haven't already done so run the script in the above folder to generate the **MAM** database (not for commercial use or copying).



Connect to tables as follows:



	Column Labels						
	Amy Winehouse			Anna-Liese Rice			Bob Cra
Row Labels	YTD this year	YTD last year	Variance	YTD this year	YTD last year	Variance	YTD this
⊟ 2010	**59,678.28**	**62,906.73**	**(3,228.45)**	**108,724.80**	**110,165.75**	**(1,440.95)**	**155,1**
January	4,619.07	5,381.75	(762.67)	9,041.57	9,517.87	(476.31)	13,7
February	9,590.02	10,010.09	(420.07)	16,952.30	17,640.16	(687.86)	25,9
March	15,042.74	15,963.06	(920.32)	26,153.42	26,298.00	(144.58)	39,9
April	19,299.50	21,202.84	(1,903.35)	34,938.52	35,437.92	(499.40)	52,1
May	25,031.98	26,855.05	(1,823.08)	45,072.26	45,128.59	(56.33)	63,9
June	29,531.98	31,248.50	(1,716.52)	53,972.05	54,012.64	(40.60)	75,5
July	34,285.76	37,283.48	(2,997.72)	63,980.99	63,636.53	344.46	88,7
August	39,578.80	42,835.46	(3,256.66)	73,136.35	71,697.84	1,438.52	102,7
September	44,034.81	47,594.36	(3,559.55)	81,854.25	81,217.35	636.90	115,6
October	49,567.37	52,876.63	(3,309.26)	91,026.15	90,635.64	390.51	129,0
November	54,639.27	58,068.69	(3,429.41)	98,978.88	100,556.22	(1,577.33)	141,1
December	59,678.28	62,906.73	(3,228.45)	108,724.80	110,165.75	(1,440.95)	155,1

*Most of the tables and relationships are as normal, but each region's **RegionManagerId** must tie in to the value of the **StaffId** field in the **Staff** table. You'll also need the store, point-of-sale, transaction and calendar tables.*

The company manager wants to be able to compare year-to-date figures for 2010 for the various regional managers against the year-to-date figures for the corresponding previous period. Your task is to create these figures!

Here's what Wise Owl think the final answer should look like:

The figures for the first couple of regional managers ...

*You should find the **DATESYTD** function useful, as shown in the manual.*

Save this workbook as **The full Monty**, and close it down.