Data Technician

Name: Athiramol Padinjarekudiyil Sukumaran

Course Date: Monday 11 August 2025 - Thursday 14 August 2025

Table of contents

Day 1: Task 1

Please research the different versions of Tableau, compare and contrast them below and explain the limited functionality on 'Tableau Public'.

Different Tableau versions

Version	Description	Key Features	Target Users	Cost
Tableau Desktop	Full-featured authoring tool	- Create advanced visualizations and dashboards - Connect to various data sources (local and cloud) - Create calculated fields, parameters, and complex analytics - Publish to Tableau Server or Tableau Online	Data analysts, business users, developers	Paid (subscription/license)
Tableau Server	On-premises server for sharing and collaboration	- Host dashboards within the organization - Manage users and permissions - Schedule data refreshes - Supports large	IT teams, enterprises	Paid (subscription/license)

		enterprise deployment		
Tableau Online	Cloud-based Tableau Server	- Similar functionality as Tableau Server - Hosted by Tableau, no infrastructure setup needed - Easy collaboration and sharing	Businesses wanting cloud-hosted analytics	Paid (subscription/license)
Tableau Public	Free version for public data sharing	- Create visualizations and dashboards - Connect mainly to Excel, text files, Google Sheets - Publish visualizations publicly online (no privacy)	Students, hobbyists, journalists, bloggers	Free
Tableau Prep	Data preparation and cleaning tool	- Drag and drop interface to clean and shape data - Connect multiple data sources - Output prepared data for Tableau Desktop or other tools	Data engineers, analysts	Paid (license/subscription)

Key Differences

- ➔ **Data Sources:** Desktop, Server, and Online connect to many databases (SQL, Oracle, cloud sources), while Public mostly supports flat files like Excel, CSV, Google Sheets.
- ➔ **Data Privacy:** Tableau Public **only allows publishing workbooks publicly** on Tableau's public server — you cannot keep your work private.
- ➔ **Functionality:** Tableau Public lacks advanced analytics features, automation, and enterprise-level sharing and security.
- ➔ **Cost:** Tableau Public is free; other versions require paid licenses or subscriptions.
- ➔ **Collaboration:** Server and Online support team collaboration with user roles, permissions, and scheduled refreshes, which Public doesn't.

Limitations of Tableau Public

- ➔ **No Private Workbooks:** All dashboards published via Tableau Public are accessible to anyone online. There is no option to save dashboards privately or restrict access.
- ➔ **Limited Data Connectivity:** It supports only basic file types (Excel, CSV, Google Sheets) and no direct connection to databases or live data sources.
- ➔ **No Data Refresh Automation:** You cannot schedule data refreshes; any data updates must be manually uploaded.
- ➔ **Limited Advanced Features:** Some calculation types, advanced analytics, and integration options are restricted.
- ➔ **No Enterprise Features:** Lacks administrative controls, user management, and other security features necessary for businesses.

Day 1: Task 2

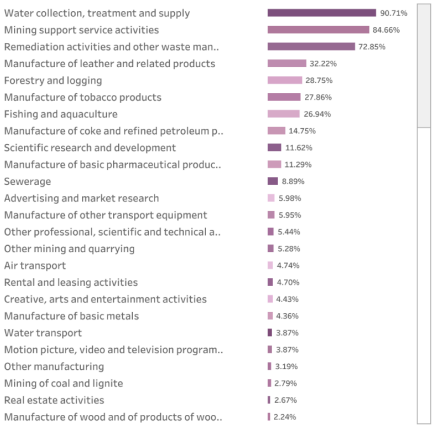
Using the *EMSI_JobChange_UK* dataset, create your own dashboard, I want to see a bar chart showing percentage change and a UK based map showing the key city locations impacted.

Paste your print screen here

UK Job Market Dynamics: Industry Changes & Regional Impact (2011-2014)

This dashboard visualizes employment trends across key UK cities from 2011 to 2014.
It highlights industries with significant job growth or decline, and geographically maps how different regions have been impacted by these changes.
Understanding these patterns helps policymakers and businesses make informed decisions about economic development and workforce planning.

Percentage Change in Jobs by Sub - Industry



Top growth
Water collection, treatment & supply (+90.71%)
Mining support services (+84.66%)
Waste management & remediation (+72.85%)

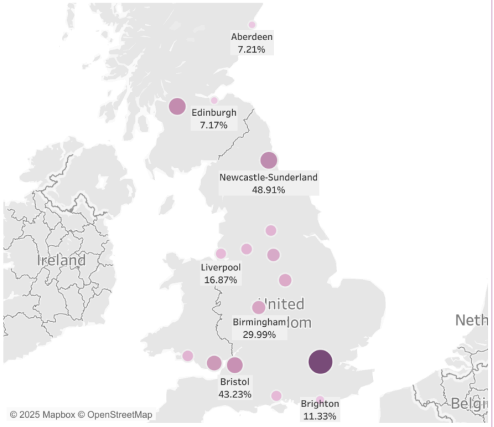
Moderate growth
Leather products (+32.22%), Forestry (+28.75%), Tobacco products (+27.86%), Fishing (+26.94%)

Science & health
R&D (+11.62%), Pharmaceuticals (+11.29%)

Steady but low growth
Advertising (+5.98%), Air transport (+4.74%), Creative arts (+4.43%)

Growth is strongest in **environmental, resource, and niche manufacturing sectors**; slower in creative and transport industries.

Geographic Impact of Job Changes Across UK Cities



London stands out as the most impacted city with the **largest positive job growth**, indicated by the biggest and darkest blue bubble, suggesting a **strong regional economy and job creation hub.**

Newcastle-Sunderland shows the next highest growth rate (48.91%), signaling significant employment opportunities emerging in the northeast of England.
Other notable cities with substantial job growth include:
Birmingham (29.99%)
Bristol (43.23%)
Liverpool (16.87%)
Brighton (11.33%)

Cities in Scotland such as Edinburgh (7.17%) and Aberdeen (7.21%) show moderate growth, reflecting steady but less rapid employment increases.

The distribution suggests that job **growth is concentrated mainly in southern and central urban areas,** with some northern cities also experiencing notable increases.

Between 2011 and 2014, UK job growth was driven by both regional and sectoral strengths.

London led as the primary job creation hub, with notable surges in Newcastle-Sunderland, Bristol, and Birmingham, indicating strong urban economic activity. Growth was geographically concentrated in southern and central cities, with moderate gains in Scotland.

By industry, the fastest expansions occurred in environmental services, mining support, and waste management, supported by solid growth in niche manufacturing and resource sectors. Science and healthcare fields showed steady gains, while creative and transport industries grew more slowly.

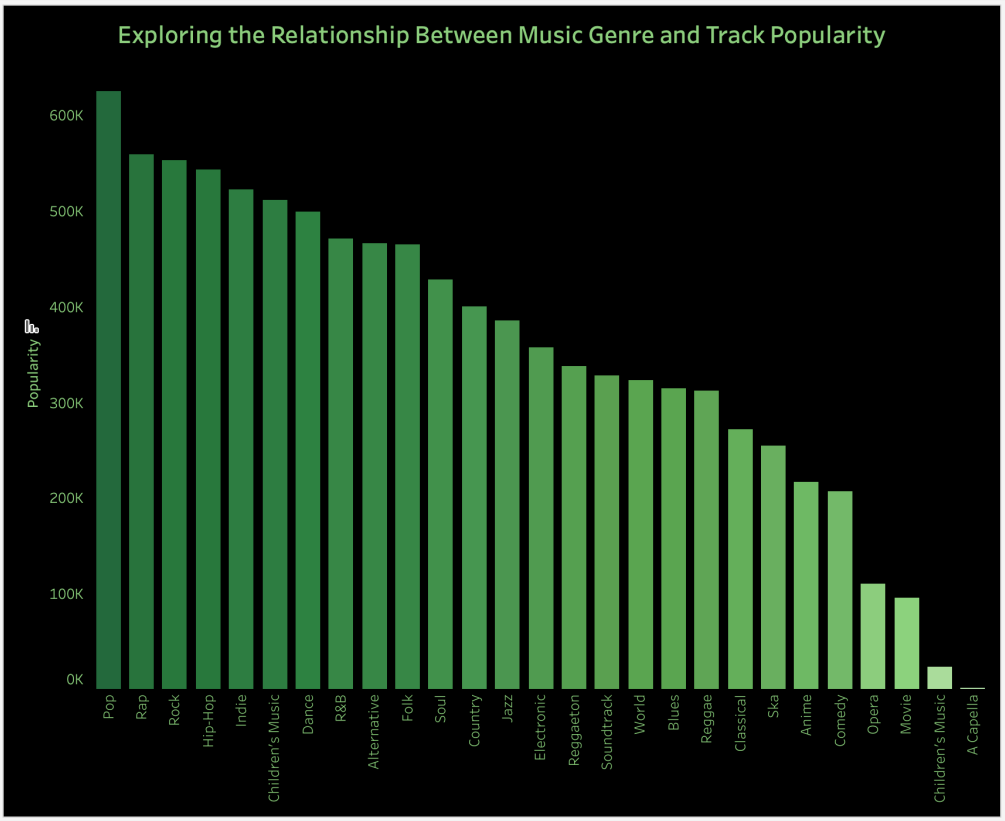
Overall, the data highlights a dual trend: robust urban growth centres and rapid expansion in environmentally and resource-focused industries.

Day 2: Task 1

Using the Spotify data set, conduct an analysis to find trends and key information that could be used by an organisation for future projects.

There is no set scope for the analysis, simply to find trends and document them below:

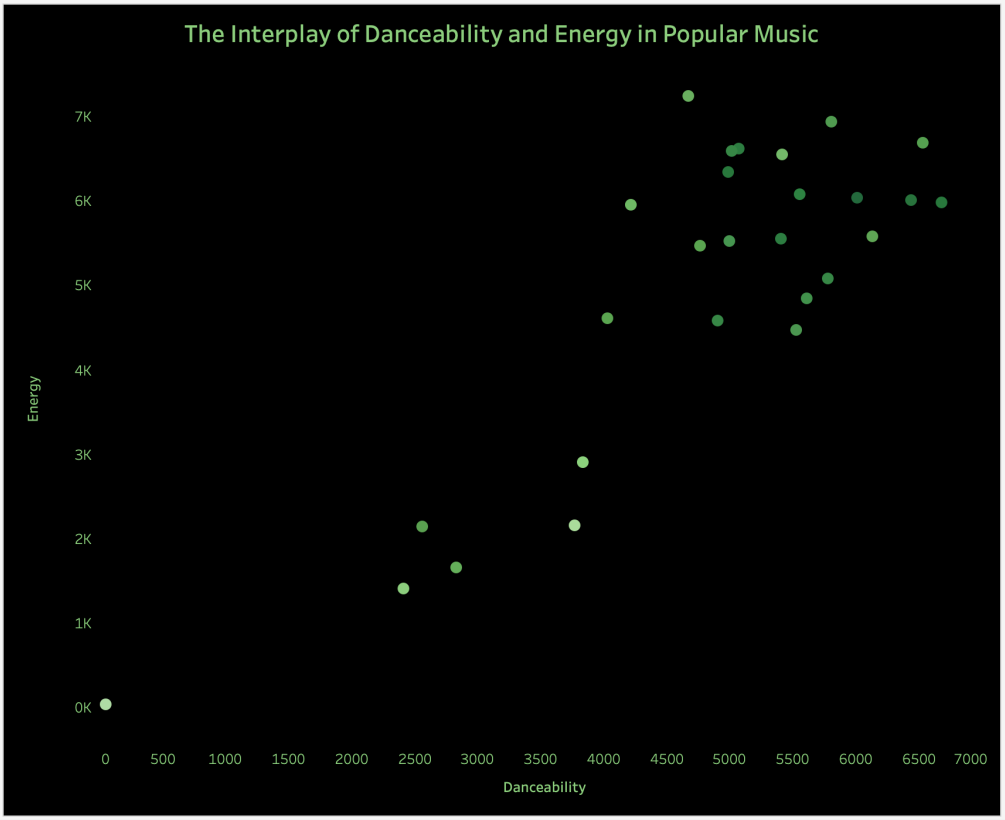
Paste your print screens here-1



What did you find?

- Pop is the most popular genre, with 625,020.
- Rap, Rock, Hip-Hop, and Indie are also highly popular, all exceeding 500,000.
- A Capella is the least popular genre, with a score of 1,107.
- There's a significant drop in popularity after the top four genres.
- Genres like Comedy, Movie, and Children's Music are among the least popular.

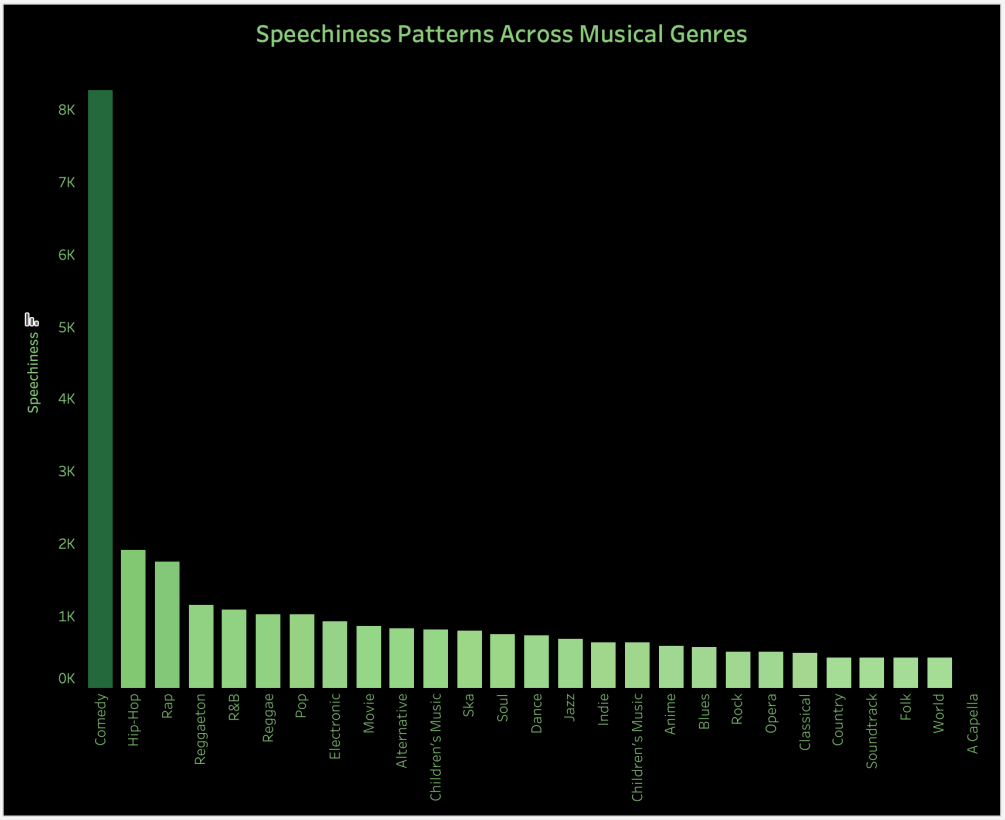
Paste your print screens here-2



What did you find?

- Danceability and energy have a positive correlation.
- There's a main cluster of songs with high danceability and high energy.
- A smaller group of songs have low danceability and low energy.
- No songs have high danceability and low energy, or vice versa.

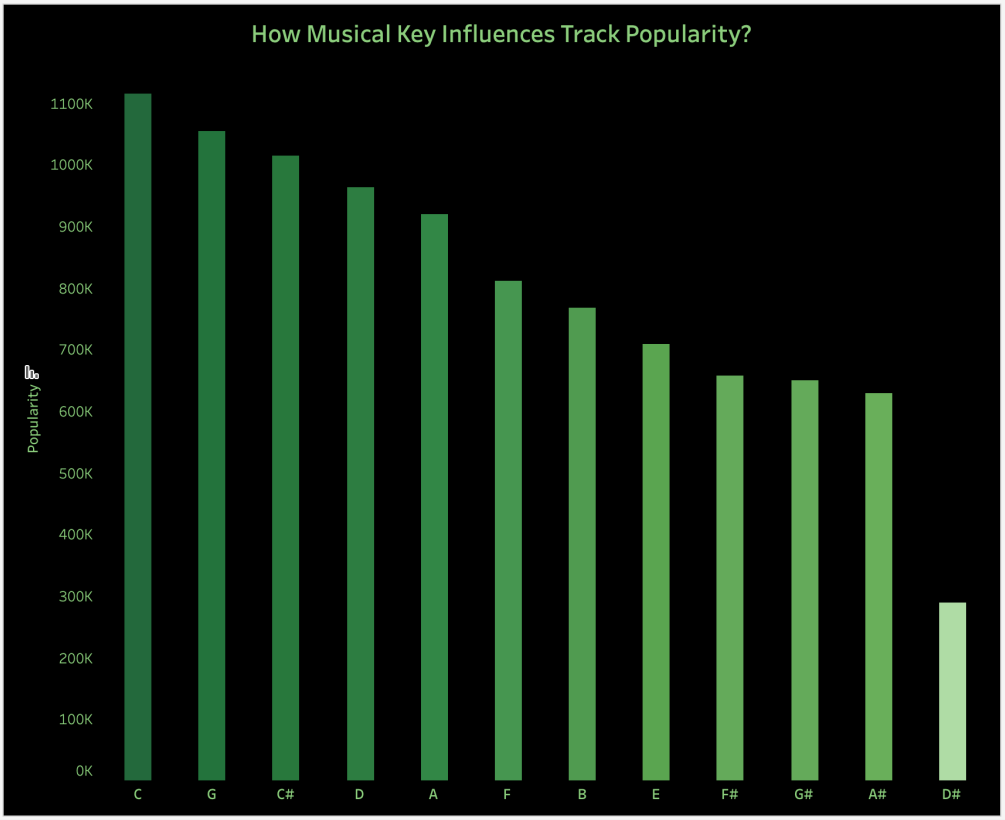
Paste your print screens here-3



What did you find?

- Comedy and Hip-Hop have the highest speechiness.
- Rap and Reggaeton follow, with lower but still high speechiness.
- A Capella, World, Folk, and Country have the lowest speechiness.
- There's a significant drop in speechiness between the top two genres and the rest.

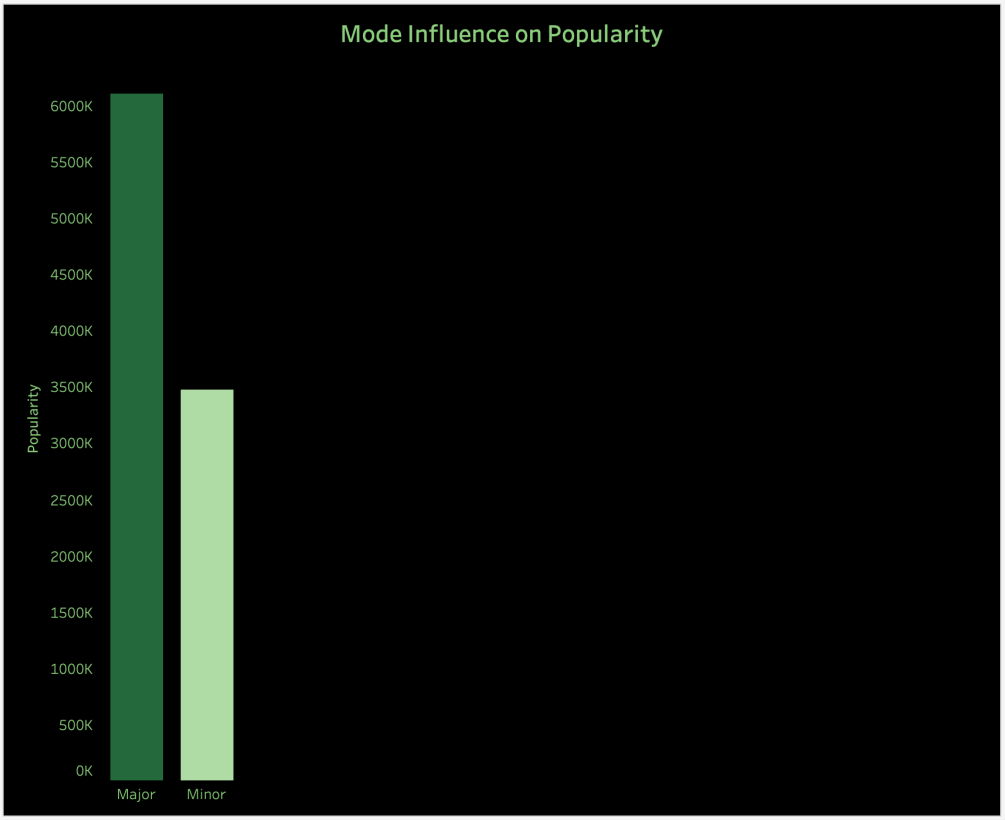
Paste your print screens here-4



What did you find?

- C major is the most popular key, with 1,114,020.
- A, C#, D, F, and G are also highly popular keys, all exceeding 800,000.
- D# is the least popular key, with a score of 288,588.
- The popularity of keys varies significantly, with a clear low point at D#.

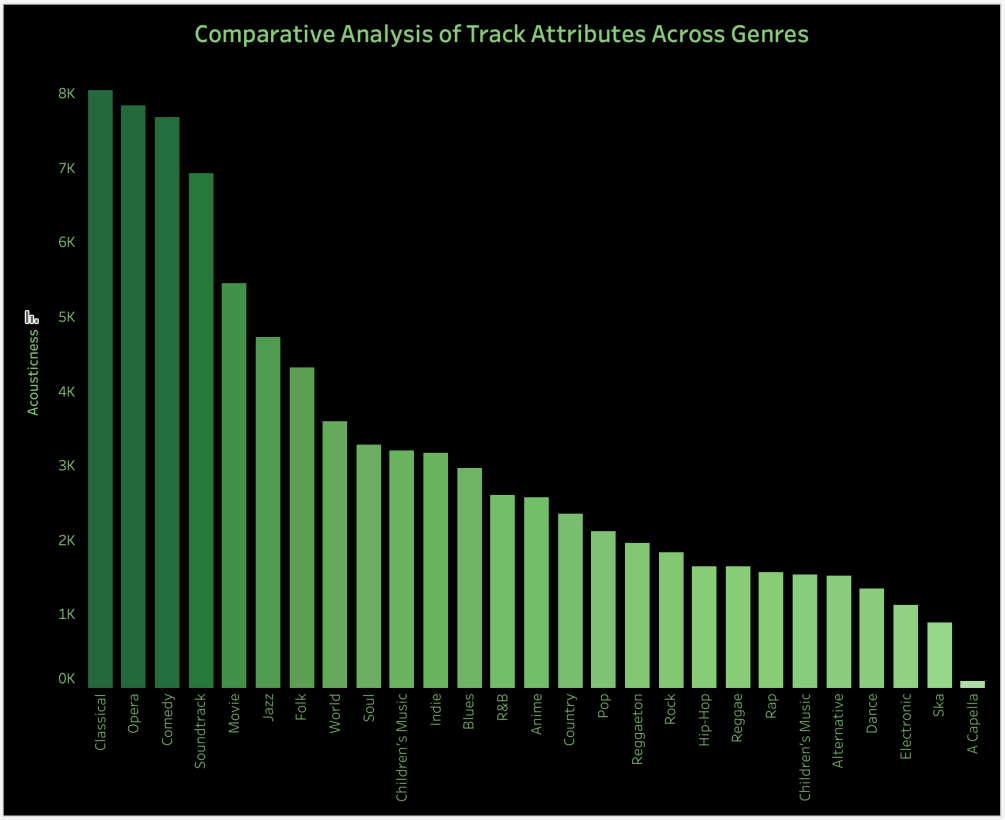
Paste your print screens here-5



What did you find?

- Major mode songs are more popular than minor mode songs.
- Major mode has a popularity of 6,098,594.
- Minor mode has a popularity of 3,472,804.
- Major mode is nearly double the popularity of minor mode.

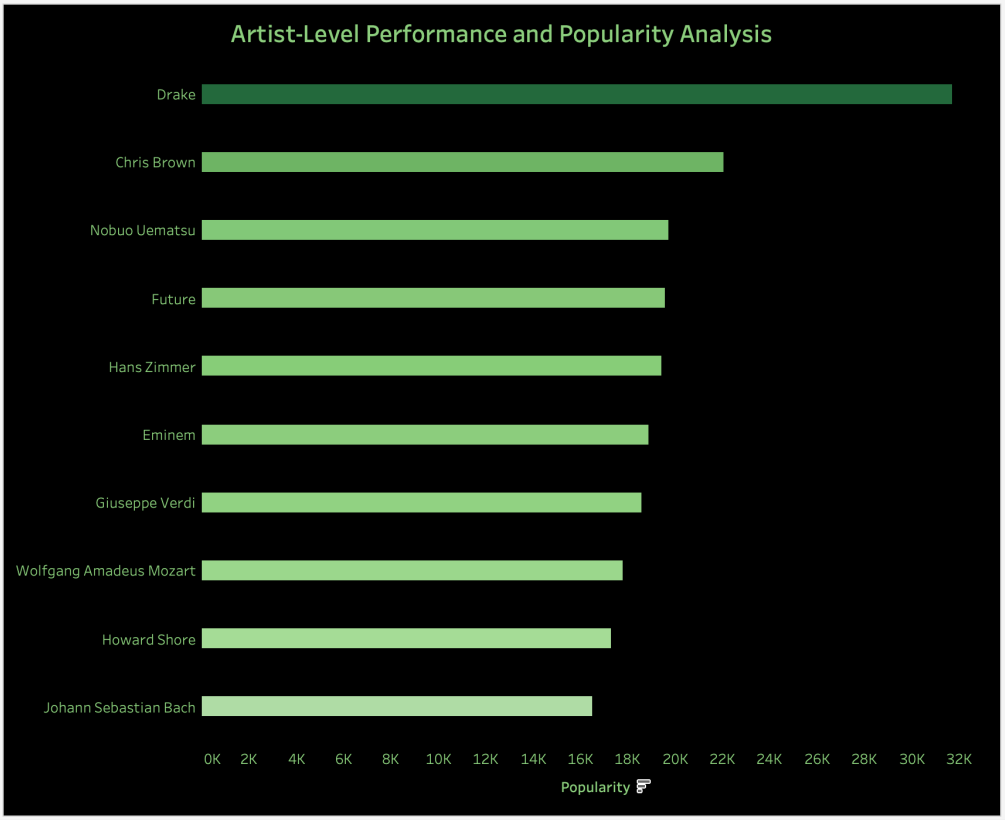
Paste your print screens here-6



What did you find?

- Classical and Opera have the highest acousticness.
- Comedy and Soundtrack are also very high in acousticness.
- A Capella, Ska, and Electronic have the lowest acousticness.
- Acousticness values range from over 8,000 to just under 1,000.

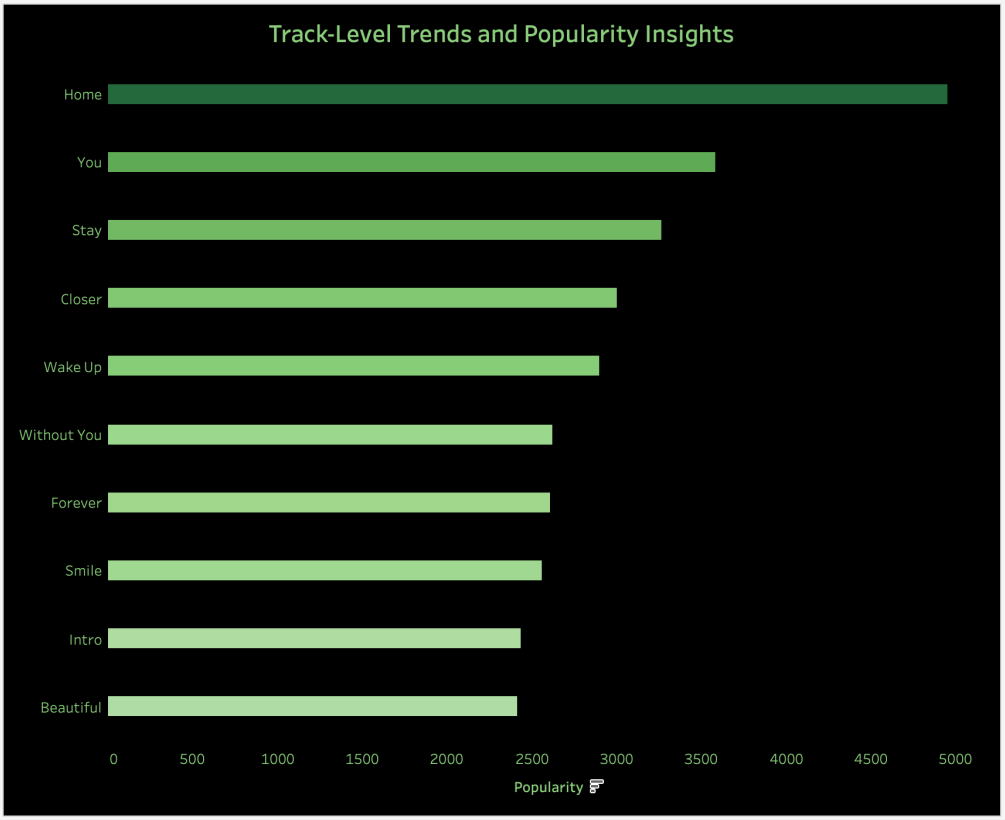
Paste your print screens here-7



What did you find?

- Drake leads significantly with the highest popularity score (31,703).
- Chris Brown ranks second, far behind Drake but ahead of others.

Paste your print screens here-8



What did you find?

- *Home* dominates with the highest popularity, far ahead of others.
- *You* and *Stay* hold strong second and third positions.
- Popularity steadily declines from *Closer* downwards.

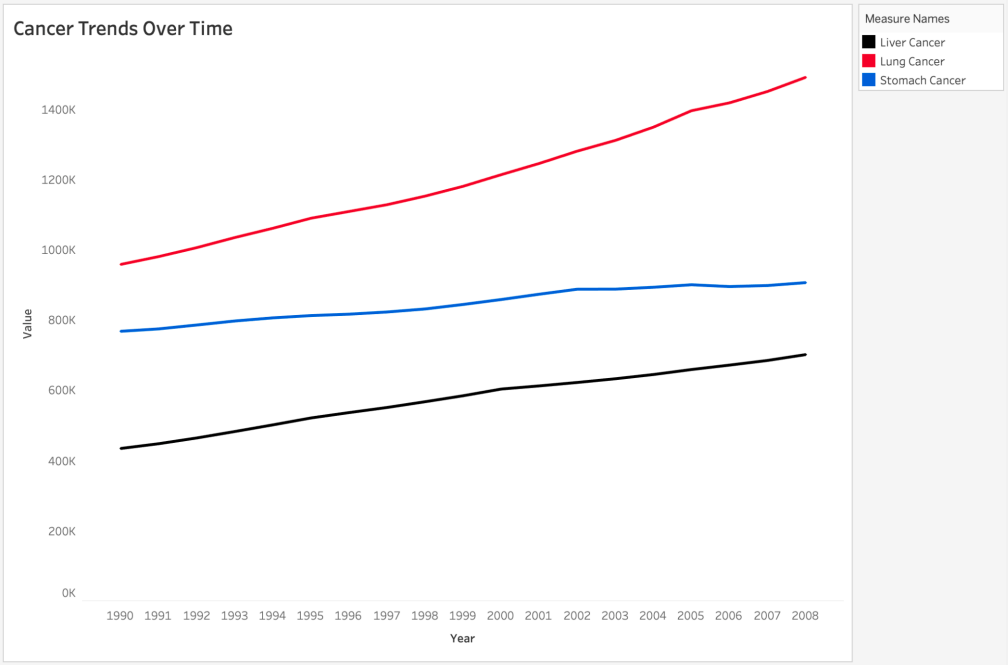
Day 2: Task 2

Using the Health data set, conduct an analysis to find trends and key information that could be used by an organisation for future support.

There is no set scope for the analysis, simply to find trends and document them below.

- Data can be lifesaving and is being used more within the NHS, reflecting on how this data could support decision making for the NHS.

Paste your print screens here-1

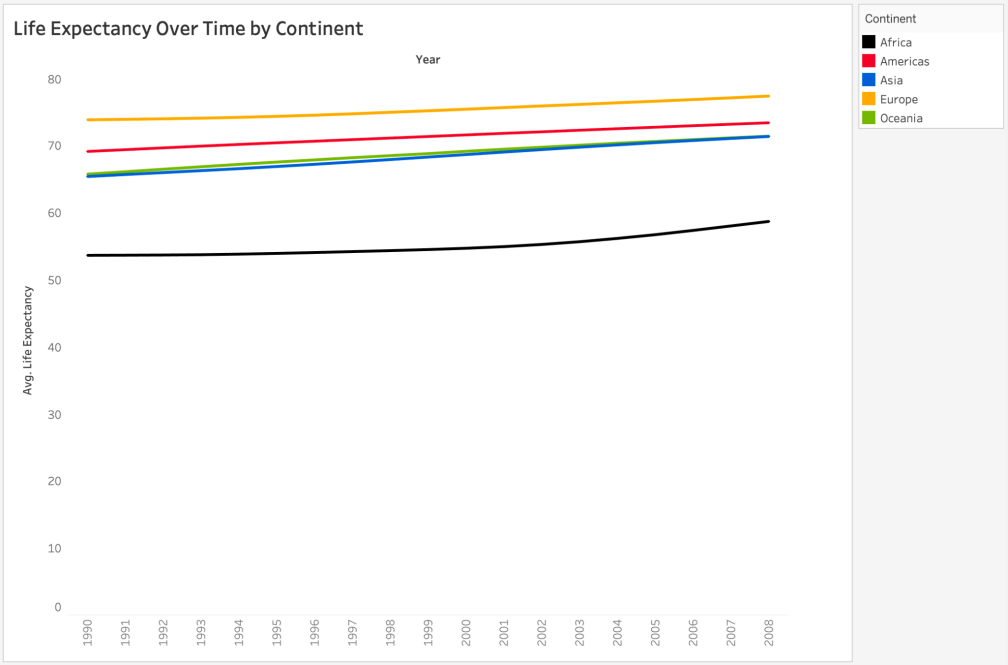


What did you find and any reflections on how the NHS could use this?

- All three cancer types—**Lung Cancer**, **Stomach Cancer**, and **Liver Cancer**—showed an upward trend from 1990 to 2008.
- **Lung Cancer** consistently had the highest number of cases, increasing from over 950,000 to nearly 1.5 million.
- **Liver Cancer** had the largest proportional increase, more than doubling its initial number of cases.
- **Stomach Cancer** showed the slowest growth rate over the period, with the trend flattening out towards the end of the data range.

The NHS can use this data for **strategic resource planning**, allocating more resources to the most prevalent and fastest-growing cancer types, particularly **lung** and **liver** cancers. The slow growth of **stomach cancer** could be a positive indicator of effective public health campaigns or improved screening, which the NHS could further investigate. This information is crucial for anticipating future demands on oncology services, guiding research priorities, and tailoring prevention efforts for specific cancer types.

Paste your print screens here-2

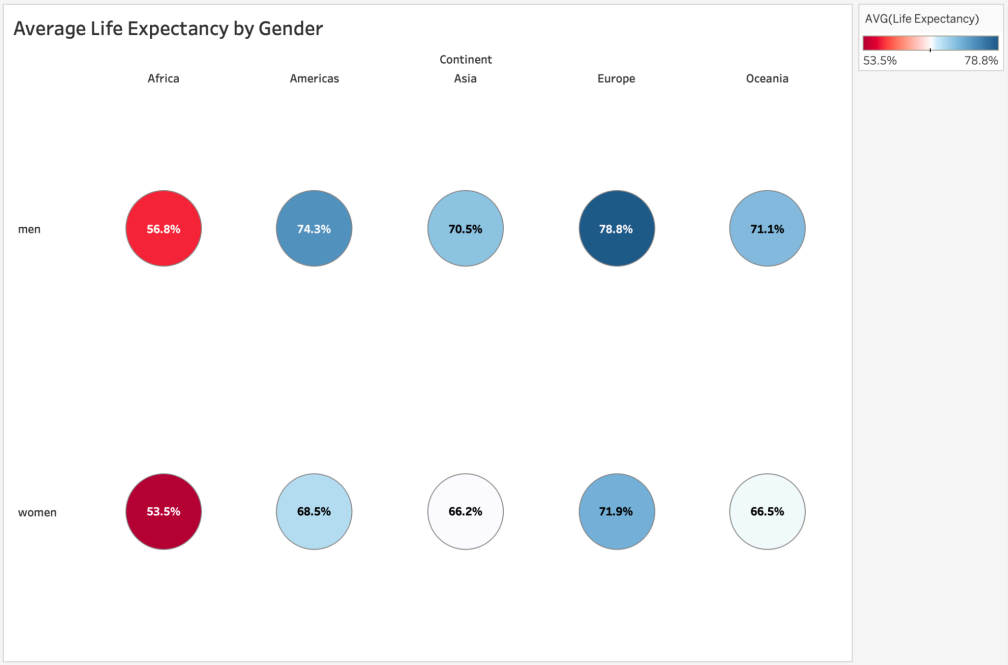


What did you find and any reflections on how the NHS could use this?

- Life expectancy increased across all continents from 1990 to 2008.
- **Europe** consistently had the highest average life expectancy.
- **Africa** consistently had the lowest average life expectancy.
- While all continents saw an increase, the gap between the highest and lowest life expectancies remained significant.

The NHS can use this data to benchmark the UK's life expectancy (part of Europe) against global averages. This can inform public health policies and strategies aimed at maintaining or improving high life expectancy rates. The data also highlights global health disparities, which could be relevant for the NHS in addressing health inequalities among the UK's diverse population.

Paste your print screens here-3

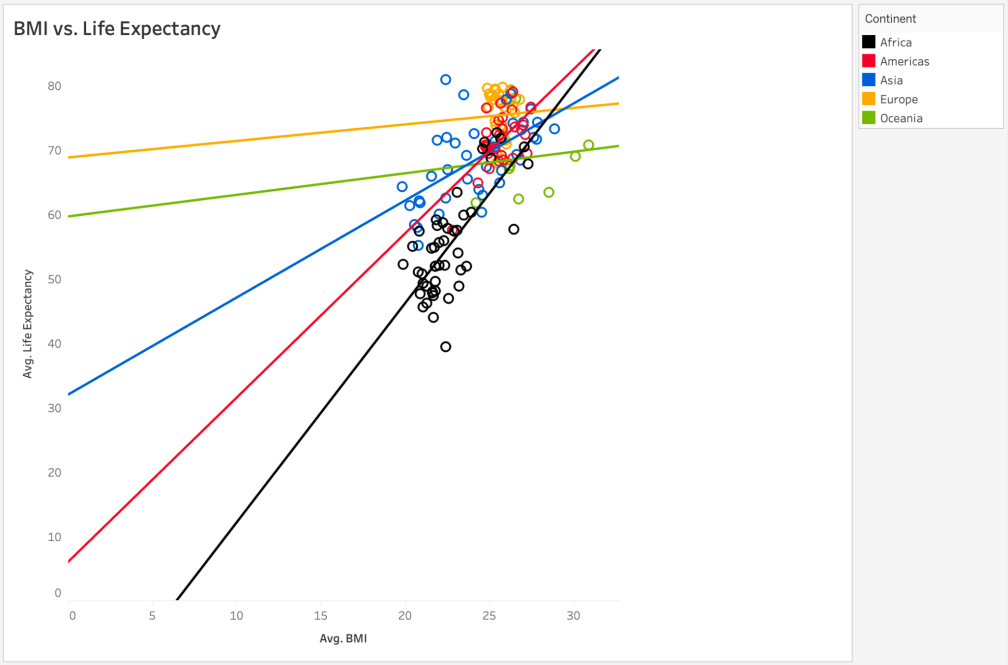


What did you find and any reflections on how the NHS could use this?

- Men generally have a higher life expectancy than women in all continents.
- Europe has the highest average life expectancy for both men and women. Africa has the lowest.
- The largest gender gap in life expectancy is in the Americas, where men live 5.8 years longer than women.

The NHS could use this data to investigate the reasons behind the gender disparity, particularly in Europe where women have a higher life expectancy. This could inform public health campaigns targeting specific health risks for men and women, aiming to close the gap and improve overall health outcomes.

Paste your print screens here-4

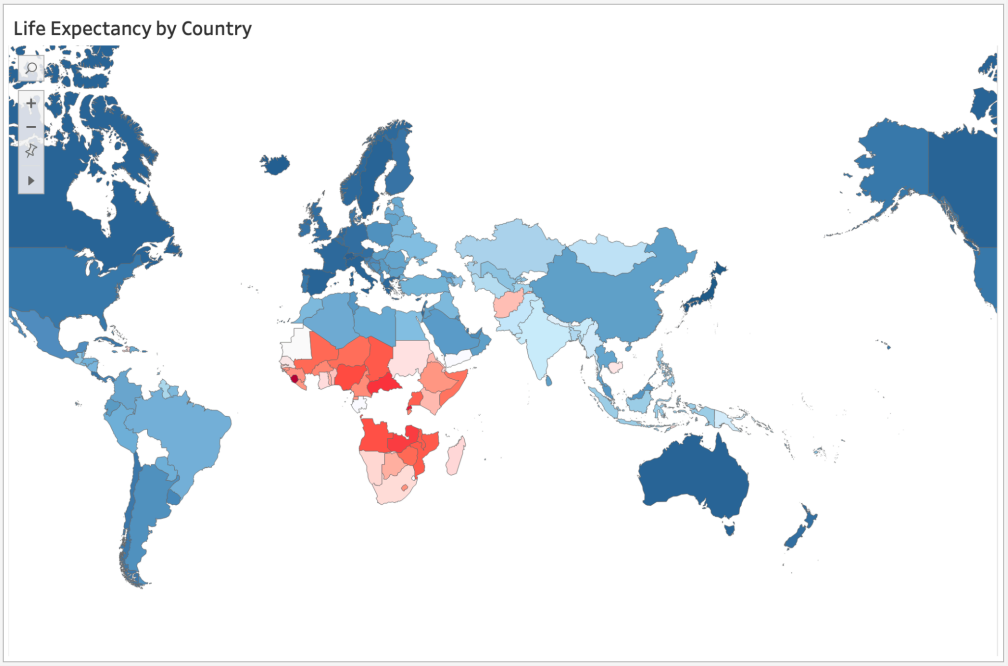


What did you find and any reflections on how the NHS could use this?

- There's a positive correlation between average BMI and average life expectancy for all continents. Generally, as BMI increases, so does life expectancy.
- **Europe** and **Oceania** have the highest average life expectancy across the BMI range.
- The trend lines for **Africa** and the **Americas** are the steepest, indicating that a rise in BMI is associated with the largest increase in life expectancy in these regions.
- **Asia** has the lowest life expectancy for a given BMI compared to other continents.

The NHS can use this data to understand the complex relationship between BMI and life expectancy. The positive correlation shown in the chart doesn't mean a higher BMI is healthy; rather, it could suggest that factors contributing to higher BMI (e.g., improved nutrition, economic development) also contribute to longer life. The NHS could investigate these underlying factors to improve public health initiatives. This data could inform strategies to address health disparities, recognizing that the link between BMI and life expectancy varies significantly by region.

Paste your print screens here-5



What did you find and any reflections on how the NHS could use this?

- Life expectancy varies significantly across the globe. North America, most of Europe, Australia, and Japan show high life expectancy, indicated by darker shades of blue.
- Most of the countries with the lowest life expectancy (indicated by red and orange shades) are in sub-Saharan Africa. The numbers are often in the 40s and 50s.
- There are clear regional patterns. For example, countries in Western Europe generally have higher life expectancy than those in Eastern Europe, and a strong contrast exists between countries in northern and southern Africa.

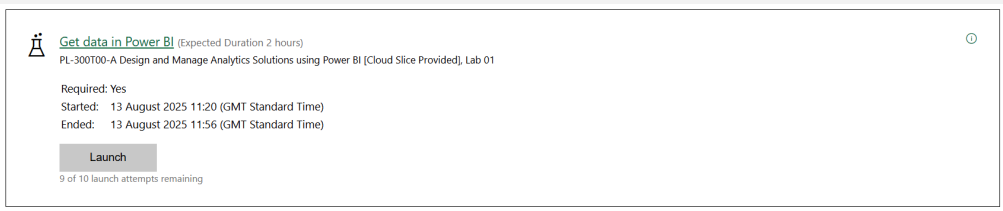
The NHS could use this data to benchmark the UK's life expectancy against other countries. The UK's position within a region of high life expectancy indicates a successful healthcare system and public health policies, but it can also be used to identify global best practices from countries with even higher numbers. Furthermore, this data is useful for understanding the health backgrounds and potential needs of diverse populations, especially for new immigrants, as their country of origin's average life expectancy can reflect a different set of health challenges and prevalent diseases.

Day 3: Task 1

Please complete Lab 1 'Get Data in Power Bi Desktop'. Once complete, paste a print screen below and in the collaboration board.

"Teaching is the best way to learn, so please listen out for support requests from the class and we'll work through the challenges together"

Paste your completed lab here

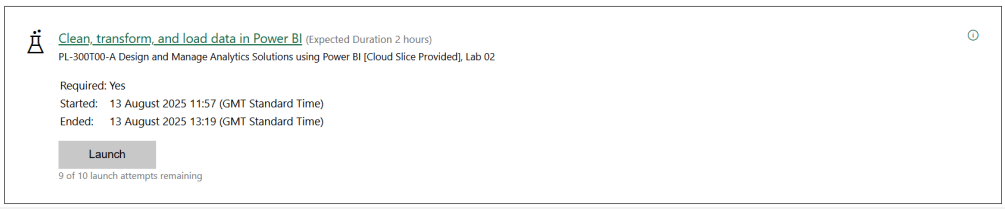


Day 3: Task 2

Please complete Lab 2 'Load Transformed Data in Power BI Desktop'. Once complete, paste a print screen below and in the collaboration board.

"Teaching is the best way to learn, so please listen out for support requests from the class and we'll work through the challenges together"

Paste your completed lab here

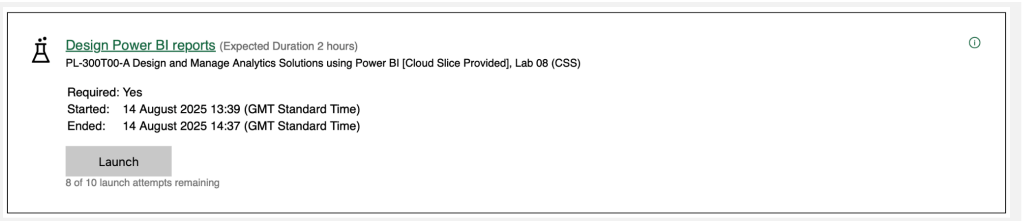


Day 4: Task 1

Please complete Lab 8 'Design a Report in Power BI Desktop'. Once complete, paste a print screen below and in the collaboration board.

"Teaching is the best way to learn, so please listen out for support requests from the class and we'll work through the challenges together"

Paste your completed lab here

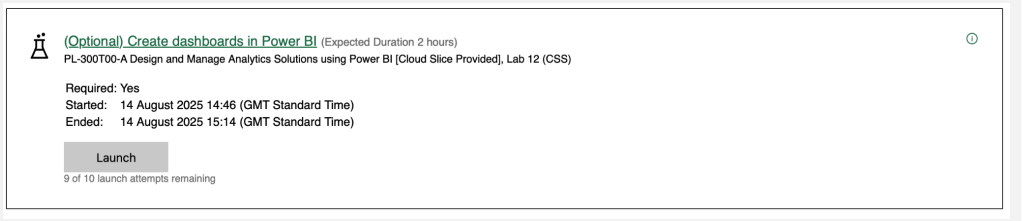


Day 4: Task 2

Please complete Lab 12 'Create a Power BI Dashboard'. Once complete, paste a print screen below and in the collaboration board.

"Teaching is the best way to learn, so please listen out for support requests from the class and we'll work through the challenges together"

Paste your completed lab here



All Lab works

Enrollment

Athiramol Padinjarekudiyil Sukumaran

PL-300T00-A Design and Manage Analytics Solutions Using Power BI [Cloud Slice Provided] (Sebastien)

Basic Information

Student: Athiramol Padinjarekudiyil Sukumaran Details
Event: PL-300T00-A Design and Manage Analytics Solutions Using Power BI [Cloud Slice Provided] (Sebastien) Details
Enrollment Status: Enrolled
Completion Status: Unknown
Expires: 18 February 2026
Is Retake: No
Enable Labs: Yes

Get Ready

Software Check	✓ OK	
✓ **1. Microsoft End User License Agreement**	View Agreement	You agreed on 13/08/2025 11:08.

Microsoft Learn Courseware

Activities

Access to your labs will expire on 18 February 2026 16:00 (GMT Standard Time)

100%

12 of 12 required activities complete

Interactive Labs

1 Get data in Power BI (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 01

Required: Yes
Started: 13 August 2025 11:20 (GMT Standard Time)
Ended: 13 August 2025 11:56 (GMT Standard Time)

Launch
9 of 10 launch attempts remaining

2 Clean, transform, and load data in Power BI (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 02

Required: Yes
Started: 13 August 2025 11:57 (GMT Standard Time)
Ended: 13 August 2025 13:19 (GMT Standard Time)

Launch
9 of 10 launch attempts remaining

3 Configure a semantic model in Power BI (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 03

Required: Yes
Started: 13 August 2025 14:37 (GMT Standard Time)
Ended: 13 August 2025 16:22 (GMT Standard Time)

Launch
7 of 10 launch attempts remaining

4 Create DAX calculations in semantic models (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 04

Required: Yes
Started: 14 August 2025 19:25 (GMT Standard Time)
Ended: 14 August 2025 19:50 (GMT Standard Time)

Launch
9 of 10 launch attempts remaining

5 Modify DAX filter context in Power BI (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 05

Required: Yes
Started: 14 August 2025 17:55 (GMT Standard Time)
Ended: 14 August 2025 18:10 (GMT Standard Time)

Launch
9 of 10 launch attempts remaining


6
Use DAX time intelligence functions in Power BI (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 06
Required: Yes
Started: 14 August 2025 18:11 (GMT Standard Time)
Ended: 14 August 2025 18:18 (GMT Standard Time)
Launch
9 of 10 launch attempts remaining
7
Create visual calculations in Power BI Desktop (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 07
Required: Yes
Started: 14 August 2025 19:16 (GMT Standard Time)
Ended: 14 August 2025 19:24 (GMT Standard Time)
Launch
9 of 10 launch attempts remaining
8
Design Power BI reports (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 08 (CSS)
Required: Yes
Started: 14 August 2025 13:39 (GMT Standard Time)
Ended: 14 August 2025 14:37 (GMT Standard Time)
Launch
8 of 10 launch attempts remaining
9
Enhance Power BI report designs (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 09 (CSS)
Required: Yes
Started: 14 August 2025 18:36 (GMT Standard Time)
Ended: 14 August 2025 18:51 (GMT Standard Time)
Launch
8 of 10 launch attempts remaining
10
Perform analytics in Power BI (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 10
Required: Yes
Started: 14 August 2025 18:52 (GMT Standard Time)
Ended: 14 August 2025 19:00 (GMT Standard Time)
Launch
9 of 10 launch attempts remaining
11
Secure data access in Power BI (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 11
Required: Yes
Started: 14 August 2025 19:00 (GMT Standard Time)
Ended: 14 August 2025 19:15 (GMT Standard Time)
Launch
9 of 10 launch attempts remaining
12
(Optional) Create dashboards in Power BI (Expected Duration 2 hours)
PL-300T00-A Design and Manage Analytics Solutions using Power BI [Cloud Slice Provided], Lab 12 (CSS)
Required: Yes
Started: 14 August 2025 14:46 (GMT Standard Time)
Ended: 14 August 2025 15:14 (GMT Standard Time)
Launch
9 of 10 launch attempts remaining
13
Practice Assessment: PL-300T00-A Microsoft Power BI Data Analyst Details
Required: No
Started: 14 August 2025 19:51 (GMT Standard Time)
Launch

Course Notes

It is recommended to take notes from the course, use the space below to do so, or use the revision guide shared with the class:

Day 1 Notes: Introduction to Visualization Tools (Tableau & Power BI)

Data Types

- **Discrete Data:** Can be counted, like age and shoe size.
- **Continuous Data:** Can be measured, like temperature and distance

Dimensions & Measures (Blue & Green)

- **Blue (Discrete):** Creates categories or distinct groups, like separate regions in a bar chart.
- **Green (Continuous):** Creates smooth axes or ranges, like a line chart tracking monthly sales

Day 2 Notes: Advanced Tableau & Introduction to Power BI

- Duration in Minutes formula in Tableau

[Duration Ms]/60000

➢ What are Parameters in Tableau?
In Tableau, **parameters** are dynamic workbook variables that allow users to replace a constant value in a calculation, filter, or reference line.

Day 3 Notes: Power BI and Data Transformation

Data Transformation and Shaping

- You can use the Power Query Editor to perform common transformations.

→ Power BI Data Types:

Decimal Currency Whole number percentage date/time Date Time Data/Time/Zone, Duration, Text, True/False, Binary.

- Binary - data type is used to store raw binary objects such as files images, PDFs, Excel sheets or any other non-text non numeric content.
  - The content column is a binary, it holds the actual file data.

→ Types of power BI

- Power BI Service. - Single page dashboards
- Storage Modes - Import -
  - Direct Query - eg streaming data

Benefits:
- frequent change
- large volume.
- multi dimensional data

Limitations:
- Data source performance
- Security btwn source & desti
- Ltd modelling capabilities
- Ltd transformation features

→ Data Profiling

Distinct - If I see the same thing more than once I'll keep 1 copy

Unique - If I see the same thing more than once, I know I have a

→ Data Import Errors:
- Query Timeout
- Couldn't find data formatted as a table.
- Couldn't find file.
- Data type errors.

→ Common Data Transformations

→ Data Types Changing
→ Rename Columns
→ Replace Values
→ Delete Columns
→ Split Columns
→ Add a New Column
→ Add Conditional Column. eg: Age Range defining
→ Shaping Table Structure

→ Remove Columns.
→ Remove Other Columns.
→ Remove Top Rows Bottom Rows Alternate Rows
→ Remove Duplicates Remove Blank Rows, Remove Errors

→ Change Column Data Type.

Change Simply by clicking On the Data Type Menu or the Icon On the column header.

→ Combining Queries

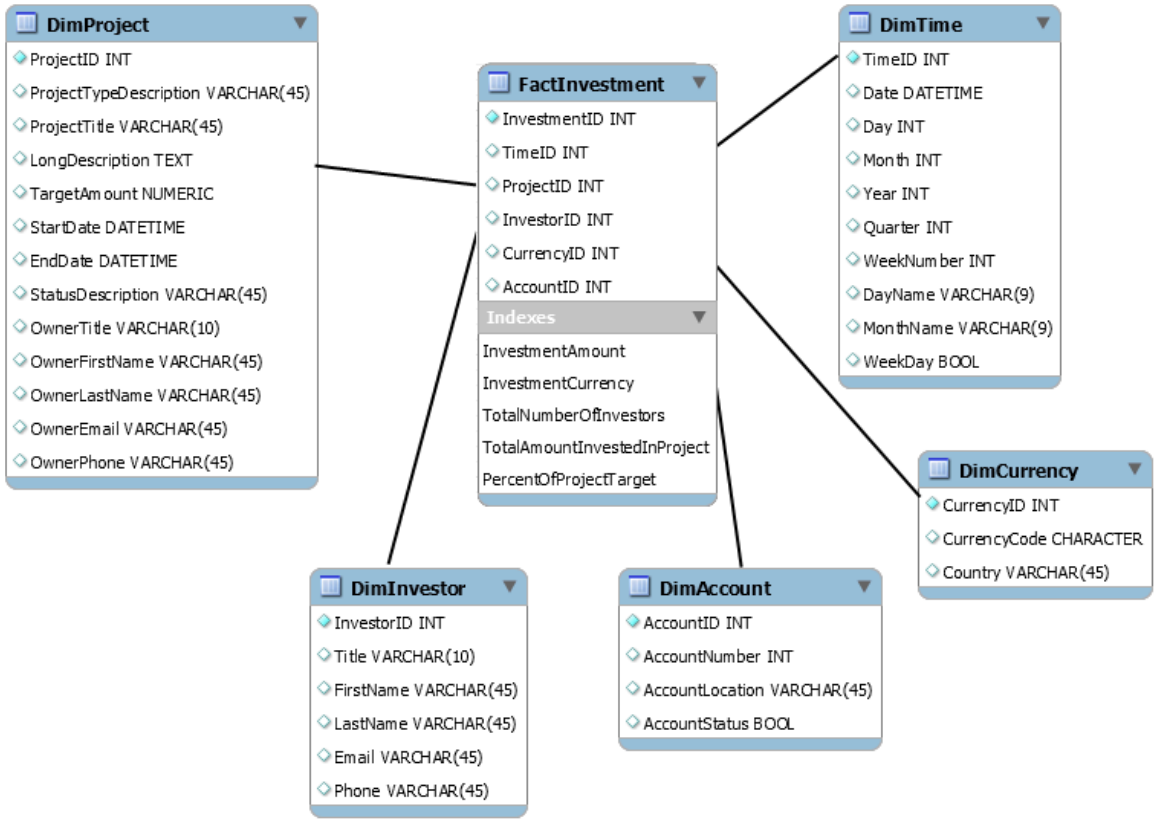
→ Append - Stacking data vertically
- Combine rows from two or more tables that have same fields
→ Merge - Joining data horizontally
- Combine columns from two tables based on a matching key column(s)

Data Import and Query

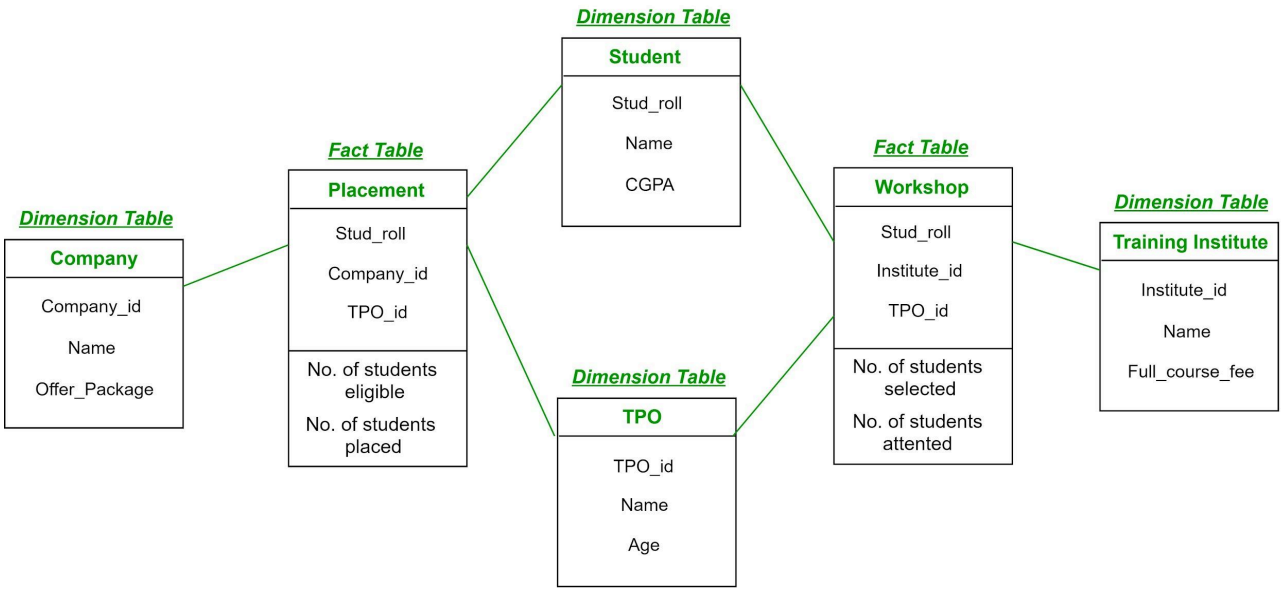
- **Storage Modes:** The storage mode you choose affects available transformations and report performance. Not all sources support all modes.
- **Direct Query:** This mode is beneficial for frequently changing data, large data volumes, and when you need near real-time updates. However, it has limitations, such as a dependence on the data source's performance and limited modeling capabilities.
- **Query Folding:** This feature pushes data transformations to the data source for better performance and efficiency.
- **Data Import Errors:** Possible errors include Query Timeout, not finding a file, or data type errors.

Data Modeling and Relationships

- **Fact Tables:** These tables represent activities or events.
- **Dimension Tables:** These tables provide details about the data.
- The document also mentions Star Schemas and managing relationships between tables.
- **Star Schema**



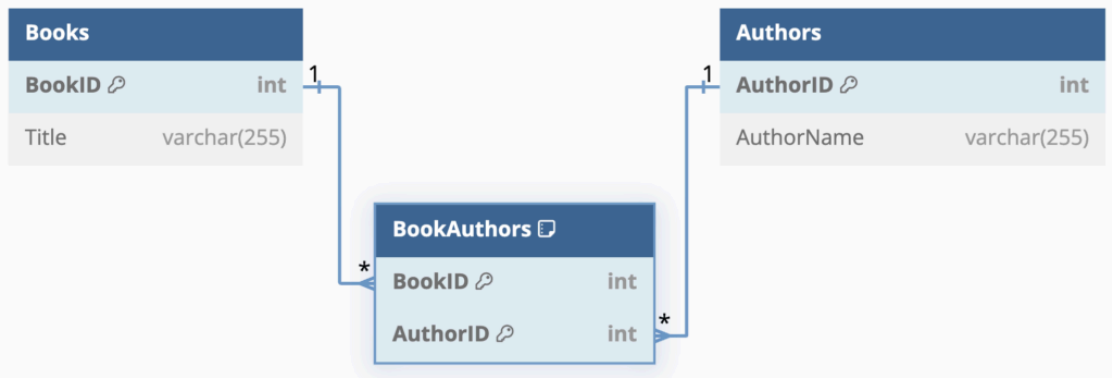
- **Galaxy Schema**



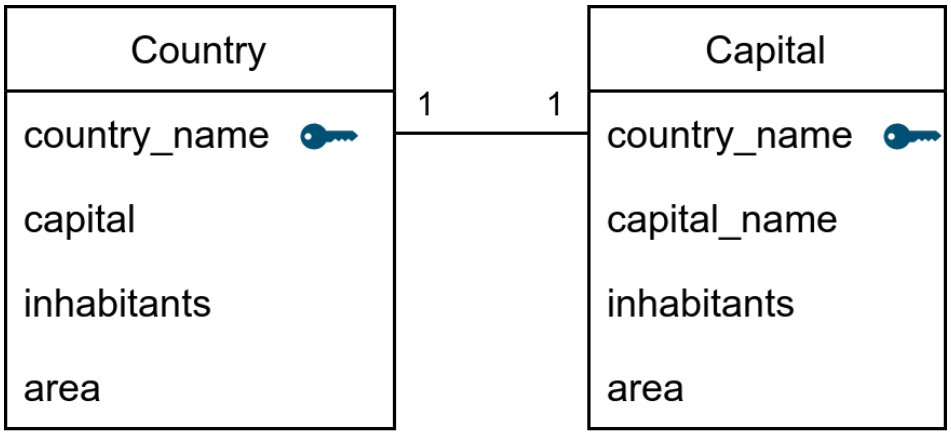
❖ Flat Table Schema

Id	Name	Email	Phone

❖ Many to many Schema



❖ One to one Schema



DAX (Data Analysis Expressions)

- DAX is a library of functions and operators used to build formulas and expressions.
- It is used to create calculated measures, columns, and tables.
- **Calculated Columns vs. Measures:**
  - **Calculated Columns:** Create a new value for each row in a table. They are stored in the data model and can increase its size.
  - **Measures:** Calculate on demand based on filters. They don't affect the data model size.

Data Best Practices

- Only keep necessary data.
- Check data types.
- Reduce cardinality.
- Use parameters

Data Modeling and Relationships

- **Fact Tables:** These tables represent activities or events, such as sales transactions.
- **Dimension Tables:** These tables provide details about the data in the fact tables. For example, a dimension table might contain details about products, customers, or dates.
- The notes also mention **Star Schemas** and how to manage the **relationships** between these different tables.

Day 4 Notes: Power BI Visuals, Reports & Advanced Features

- **Identifying Outliers: Outliers** are unexpected deviations in data. They can be detected using scatter charts and analyzed with DAX.
- **Data Protection:** You can control how users can export data and set data sensitivity.
- **Usage Metrics:** Power BI provides metrics on how your reports and dashboards are being used.

LAB Notes

DAX time intelligence functions in Power BI

Create a YTD measure

```
Sales YTD =
TOTALYTD(
   SUM(Sales[Sales]),
   'Date'[Date],
   "6-30"
)
```

The `TOTALYTD` function performs filter manipulation, specifically time filter manipulation. For example, to compute YTD sales for September 2017 (the third month of the fiscal year), all filters on the `Date` table are removed and replaced with a new filter of dates commencing at the beginning of the year (July 1, 2017) and extending through to the last date of the in-context date period (September 30, 2017).

Create a YoY growth measure

```
Sales YoY Growth =
VAR SalesPriorYear =
   CALCULATE(
      SUM(Sales[Sales]),
      PARALLELPERIOD(
         'Date'[Date],
         -12,
         MONTH
      )
   )
RETURN
   SalesPriorYear
```

The `SalesPriorYear` variable is assigned an expression that calculates the sum of the `Sales` column in a modified context. That context uses the `PARALLELPERIOD` function to shift 12 months back from each date in filter context.

Calculations

Profit = [Sum of Sales] - [Sum of Cost]

Versus Previous compares a value to a preceding value, so we see the Profit compared to the previous value for Year.
Running sum calculates the sum of values, adding the current value to the preceding values, so we see the total of current and previous years.
Moving average calculates an average of a set of values in a given window by dividing the sum of the values by the size of the window. By setting the window size to 2, we are calculating the average of two consecutive values. In this example, the values are yearly profits, so we see the moving average for FY2019 is the average of the profits for FY2018 and FY2019.

Running sum = RUNNINGSUM([Sum of Sales], HIGHESTPARENT)

Create the Date table

```
Date =
CALENDARAUTO(6)
```

The `CALENDARAUTO` function returns a single-column table comprising date values. The "auto" behavior scans all data model date columns to determine the earliest and latest date values stored in the data model. It then creates one row for each date within this range, extending the range in either direction to ensure full years of data is stored.

This function can take a single optional argument that is the last month number of a year. When omitted, the value is 12, meaning that December is the last month of the year. In this case, 6 is entered, meaning that June is the last month of the year.

Create calculated columns

```
Year =
"FY" & YEAR('Date'[Date]) + IF(MONTH('Date'[Date]) > 6, 1)
```

The formula uses the date's year value but adds one to the year value when the month is after June. That's how fiscal years at Adventure Works are calculated.

Formulas and Functions for Future Reference

Tableau

1.Tableau Number Functions

Function	Category	Formula / Syntax	Description / Use
ABS	Number	`ABS(number)`	Returns the absolute value of a number.
CEILING	Number	`CEILING(number [, significance])`	Rounds a number up to the nearest integer or specified significance.
EXP	Number	`EXP(number)`	Returns e raised to the power of a number.

FLOOR	Number	`FLOOR(number [, significance])`	Rounds a number down to the nearest integer or specified significance.
LN	Number	`LN(number)`	Returns the natural logarithm of a number.
LOG	Number	`LOG(number [, base])`	Returns the logarithm of a number; optionally specify base.
MOD	Number	`MOD(number, divisor)`	Returns the remainder after division.
POWER	Number	`POWER(number, exponent)`	Raises a number to a specified power.
ROUND	Number	`ROUND(number [, decimals])`	Rounds a number to a specified number of decimal places.
SQRT	Number	`SQRT(number)`	Returns the square root of a number.
INT	Number	`INT(number)`	Returns the integer part of a number (truncates decimals).
RANK	Number	`RANK(expression [, 'asc'	'desc'])`
RANK_DENSE	Number	`RANK_DENSE(expression [, 'asc'	'desc'])`
RANK_PERCENTILE	Number	`RANK_PERCENTILE(expression)`	Returns percentile rank between 0 and 1.
RANDOM	Number	`RANDOM()`	Returns a random number between 0 and 1.

SIN	Number	`SIN(number)`	Returns the sine of a number (in radians).
COS	Number	`COS(number)`	Returns the cosine of a number (in radians).
TAN	Number	`TAN(number)`	Returns the tangent of a number (in radians).
ASIN	Number	`ASIN(number)`	Returns the arcsine of a number.
ACOS	Number	`ACOS(number)`	Returns the arccosine of a number.
ATAN	Number	`ATAN(number)`	Returns the arctangent of a number.
PI	Number	`PI()`	Returns the value of π.

2. Tableau String Functions

Function	Category	Formula / Syntax	Description / Use
LEFT	String	`LEFT(string, number_of_chars)`	Returns the leftmost characters of a string.
RIGHT	String	`RIGHT(string, number_of_chars)`	Returns the rightmost characters of a string.
MID	String	`MID(string, start, length)`	Returns a substring starting at a specified position.
LEN	String	`LEN(string)`	Returns the length (number of characters) of a string.
TRIM	String	`TRIM(string)`	Removes leading and trailing spaces from a string.
LTRIM	String	`LTRIM(string)`	Removes leading spaces from a string.

RTRIM	String	`RTRIM(string)`	Removes trailing spaces from a string.
LOWER	String	`LOWER(string)`	Converts all characters to lowercase.
UPPER	String	`UPPER(string)`	Converts all characters to uppercase.
REPLACE	String	`REPLACE(string, start, length, replacement)`	Replaces part of a string with another string.
REPLACE_REGEX	String	`REPLACE_REGEX(string, pattern, replacement)`	Replaces all matches of a regex pattern in a string.
SPLIT	String	`SPLIT(string, delimiter, token_number)`	Splits a string by a delimiter and returns the specified token.
FIND	String	`FIND(string, substring [, start])`	Returns the position of a substring within a string.
CONTAINS	String	`CONTAINS(string, substring)`	Returns TRUE if a string contains the specified substring.
STARTSWITH	String	`STARTSWITH(string, substring)`	Returns TRUE if a string starts with the specified substring.
ENDSWITH	String	`ENDSWITH(string, substring)`	Returns TRUE if a string ends with the specified substring.
STR	String	`STR(number)`	Converts a number or date to a string.
FORMAT	String	`FORMAT(value, format_string)`	Formats a value as a string according to a pattern.
ASCII	String	`ASCII(character)`	Returns the ASCII code of the first character in a string.

CHAR	String	`CHAR(number)`	Returns the character corresponding to an ASCII code.

3. Tableau Date Functions

Function	Category	Formula / Syntax	Description / Use
TODAY	Date	`TODAY()`	Returns the current date.
NOW	Date	`NOW()`	Returns the current date and time.
DATE	Date	`DATE(year, month, day)`	Creates a date from year, month, and day values.
DATEADD	Date	`DATEADD(date_part, increment, date)`	Adds a specified interval (days, months, years) to a date.
DATEDIFF	Date	`DATEDIFF(start_date, end_date, date_part)`	Returns the difference between two dates in specified units.
DATENAME	Date	`DATENAME(date_part, date)`	Returns the name of the specified date part (e.g., “January”).
DATEPART	Date	`DATEPART(date_part, date)`	Returns the numeric value of a specified date part (e.g., month number).
MAKEDATE	Date	`MAKEDATE(year, month, day)`	Returns a date from individual year, month, and day values.
MAKETIME	Date	`MAKETIME(hour, minute, second)`	Returns a time from individual hour, minute, and second values.
MAKEDATETIME	Date	`MAKEDATETIME(date, time)`	Combines a date and time into a datetime value.
YEAR	Date	`YEAR(date)`	Returns the year of a date.

MONTH	Date	`MONTH(date)`	Returns the month of a date.
DAY	Date	`DAY(date)`	Returns the day of a date.
HOUR	Date	`HOUR(time)`	Returns the hour of a time.
MINUTE	Date	`MINUTE(time)`	Returns the minute of a time.
SECOND	Date	`SECOND(time)`	Returns the second of a time.
WEEKDAY	Date	`WEEKDAY(date [, start_of_week])`	Returns the weekday number of a date.
NOW()	Date	`NOW()`	Returns the current date and time.
TODAY()	Date	`TODAY()`	Returns the current date.
ISDATE	Date	`ISDATE(string)`	Checks if a string can be converted to a valid date.

4. Tableau Logical Functions

Function	Category	Formula / Syntax	Description / Use
IF	Logical	`IF test THEN result ELSE result END`	Returns one value if a condition is true, another if false.
IIF	Logical	`IIF(test, true_result, false_result [, unknown_result])`	Returns one value if true, another if false; optional value if unknown.
CASE	Logical	`CASE expression WHEN value1 THEN result1 … ELSE result END`	Checks multiple conditions and returns corresponding results.
ISNULL	Logical	`ISNULL(expression)`	Returns TRUE if the value is null, otherwise FALSE.
ZN	Logical	`ZN(expression)`	Converts null values to zero.
AND	Logical	`expression1 AND expression2`	Returns TRUE if both conditions are TRUE.
OR	Logical	`expression1 OR expression2`	Returns TRUE if at least one condition is TRUE.

NOT	Logical	`NOT(expression)`	Reverses the logical value: TRUE becomes FALSE, FALSE becomes TRUE.
MIN	Logical	`MIN(expression1, expression2, …)`	Returns the minimum value from a set of expressions.
MAX	Logical	`MAX(expression1, expression2, …)`	Returns the maximum value from a set of expressions.

5. Tableau Aggregate / Table Calculation Functions

Function	Category	Formula / Syntax	Description / Use
SUM	Aggregate	`SUM(expression)`	Returns the sum of all values in a field.
AVG	Aggregate	`AVG(expression)`	Returns the average of all values in a field.
MIN	Aggregate	`MIN(expression)`	Returns the minimum value of a field.
MAX	Aggregate	`MAX(expression)`	Returns the maximum value of a field.
COUNT	Aggregate	`COUNT(expression)`	Returns the number of non-null values.
COUNTD	Aggregate	`COUNTD(expression)`	Returns the number of distinct values.
MEDIAN	Aggregate	`MEDIAN(expression)`	Returns the median of a field.
STDEV	Aggregate	`STDEV(expression)`	Returns the standard deviation of a field.
STDEVP	Aggregate	`STDEVP(expression)`	Returns the standard deviation of a population.

VAR	Aggregate	`VAR(expression)`	Returns the variance of a field.
VARP	Aggregate	`VARP(expression)`	Returns the variance of a population.
WINDOW_SUM	Table Calc	`WINDOW_SUM(expression [, start, end])`	Returns the sum of values within a window.
WINDOW_AVG	Table Calc	`WINDOW_AVG(expression [, start, end])`	Returns the average of values within a window.
WINDOW_MIN	Table Calc	`WINDOW_MIN(expression [, start, end])`	Returns the minimum value within a window.
WINDOW_MAX	Table Calc	`WINDOW_MAX(expression [, start, end])`	Returns the maximum value within a window.
WINDOW_COUNT	Table Calc	`WINDOW_COUNT(expression [, start, end])`	Returns the count of values within a window.
RANK	Table Calc	`RANK(expression [, 'asc'	'desc'])`
RANK_DENSE	Table Calc	`RANK_DENSE(expression [, 'asc'	'desc'])`
RUNNING_SUM	Table Calc	`RUNNING_SUM(expression)`	Returns the cumulative sum of values across a partition.
RUNNING_AVG	Table Calc	`RUNNING_AVG(expression)`	Returns the cumulative average across a partition.

Power BI

1. Text Functions

Function	Category	Formula / Syntax	Description / Use
CONCATENATE	Text	`=CONCATENATE(Text1, Text2)`	Join two text strings into one.
CONCATENATEX	Text	`=CONCATENATEX( Table, Expression [, Delimiter [, OrderBy_Expression [, Order]]])`	Joins text from multiple rows in a table with a delimiter.
EXACT	Text	`=EXACT(Text1, Text2)`	Checks if two text strings are exactly the same (case-sensitive).
FIND	Text	`=FIND(FindText, WithinText [, StartPosition] [, NotFoundValue] )`	Finds the position of a substring within text (case-sensitive).
SEARCH	Text	`=SEARCH(FindText, WithinText [, StartPosition] [, NotFoundValue] )`	Finds the position of a substring within text (case-insensitive).
FIXED	Text	`=FIXED(Number [, Decimals [, NoCommas]])`	Converts a number to text with fixed decimals and optional commas.

FORMAT	Text	`=FORMAT(Value, FormatString)`	Formats numbers, dates, or text into a specific display format.
LEFT	Text	`=LEFT(Text, NumChars)`	Returns the first N characters from the start of text.
RIGHT	Text	`=RIGHT(Text, NumChars)`	Returns the last N characters from the end of text.
MID	Text	`=MID(Text, StartPosition, NumChars)`	Returns a substring from the middle of text.
LEN	Text	`=LEN(Text)`	Returns the length of the text string.
TRIM	Text	`=TRIM(Text)`	Removes extra spaces from text (leaves single spaces between words).
REPT	Text	`=REPT(Text, NumberTimes)`	Repeats text a given number of times.
REPLACE	Text	`=REPLACE(OldText, StartPosition, NumChars, NewText)`	Replaces part of a text string with new text.
SUBSTITUTE	Text	`=SUBSTITUTE(Text, OldText, NewText [, InstanceNum])`	Replaces specific text (can target a particular instance).
UPPER	Text	`=UPPER(Text)`	Converts text to uppercase.

LOWER	Text	`=LOWER(Text)`	Converts text to lowercase.
UNICHAR	Text	`=UNICHAR(Number)`	Returns the Unicode character for a given numeric code.
UNICODE	Text	`=UNICODE(Text)`	Returns the numeric Unicode value for the first character of text.

2. Aggregation

Function	Category	Formula / Syntax	Description / Use
SUM	Aggregation	`=SUM(ColumnName)`	Adds up all numeric values in a column.
SUMX	Aggregation	`=SUMX(TableName, Expression)`	Adds up values after evaluating an expression for each row in a table.
AVERAGE	Aggregation	`=AVERAGE(ColumnName)`	Calculates the average of numeric values in a column.
AVERAGEX	Aggregation	`=AVERAGEX(TableName, Expression)`	Calculates the average after evaluating an expression for each row.

COUNT	Aggregation	`=COUNT(ColumnName)`	Counts numeric values in a column.
COUNTA	Aggregation	`=COUNTA(ColumnName)`	Counts all non-blank values in a column.
COUNTAX	Aggregation	`=COUNTAX(TableName, Expression)`	Counts rows where expression is not blank.
COUNTBLANK	Aggregation	`=COUNTBLANK(ColumnName)`	Counts blank cells in a column.
COUNTROWS	Aggregation	`=COUNTROWS(TableName)`	Counts the number of rows in a table.
COUNTX	Aggregation	`=COUNTX(TableName, Expression)`	Counts rows where expression is not blank for each row.
MAX	Aggregation	`=MAX(ColumnName)`	Returns the maximum value in a column.
MAXX	Aggregation	`=MAXX(TableName, Expression)`	Returns the maximum value after evaluating an expression for each row.
MIN	Aggregation	`=MIN(ColumnName)`	Returns the minimum value in a column.

MINX	Aggregation	`=MINX(TableName, Expression)`	Returns the minimum value after evaluating an expression for each row.
MEDIAN	Aggregation	`=MEDIAN(ColumnName)`	Returns the median of numeric values in a column.
MEDIANX	Aggregation	`=MEDIANX(TableName, Expression)`	Returns the median after evaluating an expression for each row.
DISTINCTCOUNT	Aggregation	`=DISTINCTCOUNT(ColumnName)`	Counts unique values in a column.
DISTINCTCOUNTNOBLANK	Aggregation	`=DISTINCTCOUNTNOBLANK(ColumnName)`	Counts unique non-blank values.
VAR.P	Aggregation	`=VAR.P(ColumnName)`	Calculates population variance.
VAR.S	Aggregation	`=VAR.S(ColumnName)`	Calculates sample variance.
VARX.P	Aggregation	`=VARX.P(TableName, Expression)`	Calculates population variance after evaluating expression.

VARX.S	Aggregation	`=VARX.S(TableName, Expression)`	Calculates sample variance after evaluating expression.
STDEV.P	Aggregation	`=STDEV.P(ColumnName)`	Calculates population standard deviation.
STDEV.S	Aggregation	`=STDEV.S(ColumnName)`	Calculates sample standard deviation.
STDEVX.P	Aggregation	`=STDEVX.P(TableName, Expression)`	Population standard deviation after evaluating expression.
STDEVX.S	Aggregation	`=STDEVX.S(TableName, Expression)`	Sample standard deviation after evaluating expression.

3. Date & Time

Function	Category	Formula / Syntax	Description / Use
DATE	Date & Time	`=DATE(Year, Month, Day)`	Creates a date from year, month, and day values.
DATEDIFF	Date & Time	`=DATEDIFF(StartDate, EndDate, Interval)`	Calculates the difference between two dates in specified intervals (day, month, year, etc.).
DATEVALUE	Date & Time	`=DATEVALUE(TextDate)`	Converts a text date into a date value.

DAY	Date & Time	`=DAY(Date)`	Returns the day component of a date (1–31).
EDATE	Date & Time	`=EDATE(StartDate, Months)`	Returns a date that is a specified number of months before or after a date.
EOMONTH	Date & Time	`=EOMONTH(StartDate, Months)`	Returns the last day of the month, N months before or after a date.
HOUR	Date & Time	`=HOUR(Time)`	Returns the hour component of a time.
MINUTE	Date & Time	`=MINUTE(Time)`	Returns the minute component of a time.
MONTH	Date & Time	`=MONTH(Date)`	Returns the month component of a date (1–12).
NOW	Date & Time	`=NOW()`	Returns the current date and time.
SECOND	Date & Time	`=SECOND(Time)`	Returns the second component of a time.
TIME	Date & Time	`=TIME(Hour, Minute, Second)`	Creates a time from hour, minute, and second values.
TIMEVALUE	Date & Time	`=TIMEVALUE(TextTime)`	Converts text to a time value.
TODAY	Date & Time	`=TODAY()`	Returns today's date without time.
WEEKDAY	Date & Time	`=WEEKDAY(Date[, ReturnType])`	Returns the day of the week as a number (1–7).
WEEKNUM	Date & Time	`=WEEKNUM(Date[, ReturnType])`	Returns the week number of the year for a date.

YEAR	Date & Time	`=YEAR(Date)`	Returns the year component of a date.

4. Time Intelligence

Function	Category	Formula / Syntax	Description / Use
DATESYTD	Time Intelligence	`=DATESYTD(DatesColumn [, YearEndDate] [, Filter])`	Returns a table of dates from the start of the year to the latest date in context.
DATESQTD	Time Intelligence	`=DATESQTD(DatesColumn [, Filter])`	Returns a table of dates from the start of the quarter to the latest date in context.
DATESMTD	Time Intelligence	`=DATESMTD(DatesColumn [, Filter])`	Returns a table of dates from the start of the month to the latest date in context.
DATEADD	Time Intelligence	`=DATEADD(DatesColumn, NumberOfIntervals, Interval)`	Shifts a set of dates forward or backward by a specified interval (day, month, quarter, year).

PARALLELPERIOD	Time Intelligence	`=PARALLELPERIOD(DatesColumn, NumberOfIntervals, Interval)`	Returns a parallel period table for comparison (e.g., same period last year).
PREVIOUSDAY	Time Intelligence	`=PREVIOUSDAY(DatesColumn)`	Returns a table of dates for the previous day.
PREVIOUSMONTH	Time Intelligence	`=PREVIOUSMONTH(DatesColumn)`	Returns a table of dates for the previous month.
PREVIOUSQUARTER	Time Intelligence	`=PREVIOUSQUARTER(DatesColumn)`	Returns a table of dates for the previous quarter.
PREVIOUSYEAR	Time Intelligence	`=PREVIOUSYEAR(DatesColumn [, YearEndDate])`	Returns a table of dates for the previous year.
NEXTDAY	Time Intelligence	`=NEXTDAY(DatesColumn)`	Returns a table of dates for the next day.
NEXTMONTH	Time Intelligence	`=NEXTMONTH(DatesColumn)`	Returns a table of dates for the next month.
NEXTQUARTER	Time Intelligence	`=NEXTQUARTER(DatesColumn)`	Returns a table of dates for the next quarter.
NEXTYEAR	Time Intelligence	`=NEXTYEAR(DatesColumn [, YearEndDate])`	Returns a table of dates for the next year.

STARTOFMONTH	Time Intelligence	`=STARTOFMONTH(DatesColumn [, Filter])`	Returns the first date of the month in the current context.
STARTOFQUARTER	Time Intelligence	`=STARTOFQUARTER(DatesColumn [, Filter])`	Returns the first date of the quarter in the current context.
STARTOFYEAR	Time Intelligence	`=STARTOFYEAR(DatesColumn [, Filter])`	Returns the first date of the year in the current context.
ENDOFMONTH	Time Intelligence	`=ENDOFMONTH(DatesColumn [, Filter])`	Returns the last date of the month in the current context.
ENDOFQUARTER	Time Intelligence	`=ENDOFQUARTER(DatesColumn [, Filter])`	Returns the last date of the quarter in the current context.
ENDOFYEAR	Time Intelligence	`=ENDOFYEAR(DatesColumn [, Filter])`	Returns the last date of the year in the current context.
SAMEPERIODLASTYEAR	Time Intelligence	`=SAMEPERIODLASTYEAR(DatesColumn)`	Returns the same period of dates from the previous year.
TOTALMTD	Time Intelligence	`=TOTALMTD(Expression, DatesColumn [, Filter])`	Calculates the month-to-date total for an expression.

TOTALQTD	Time Intelligence	`=TOTALQTD(Expression, DatesColumn [, Filter])`	Calculates the quarter-to-date total for an expression.
TOTALYTD	Time Intelligence	`=TOTALYTD(Expression, DatesColumn [, YearEndDate] [, Filter])`	Calculates the year-to-date total for an expression.
Function	**Category**	**Formula / Syntax**	**Description / Use**
DATESYTD	Time Intelligence	`=DATESYTD(DatesColumn [, YearEndDate] [, Filter])`	Returns a table of dates from the start of the year to the latest date in context.

5. **Filter**

Function	Category	Formula / Syntax	Description / Use
ALL	Filter	`=ALL(TableNameOrColumnName)`	Removes all filters from a table or column.
ALLCROSSFILTERED	Filter	`=ALLCROSSFILTERED(TableNameOrColumnName)`	Returns all rows from a table, ignoring filters applied by cross-filtering.

ALLSELECTED	Filter	`=ALLSELECTED(TableNameOrColumnName)`	Returns all rows in a table or column after considering only filters applied outside the visual.
ALLEXCEPT	Filter	`=ALLEXCEPT(TableName, Column1 [, Column2…])`	Removes all filters except those on specified columns.
CALCULATE	Filter	`=CALCULATE(Expression [, Filter1] [, Filter2]…)`	Evaluates an expression in a modified filter context.
CALCULATETABLE	Filter	`=CALCULATETABLE(Expression, Filter1 [, Filter2]…)`	Returns a table evaluated in a modified filter context.
FILTER	Filter	`=FILTER(Table, Condition)`	Returns a table with rows that meet a condition.
FILTERS	Filter	`=FILTERS(ColumnName)`	Returns the current filter applied on a column.
KEEPFILTERS	Filter	`=KEEPFILTERS(Filter)`	Modifies a filter to keep existing filters instead of replacing them.

REMOVEFILTERS	Filter	`=REMOVEFILTERS([TableNameOrColumnName])`	Removes filters from a table or column (similar to ALL).
USERELATIONSHIP	Filter	`=USERELATIONSHIP(Column1, Column2)`	Activates an inactive relationship between two columns in a calculation.
ISFILTERED	Filter	`=ISFILTERED(ColumnName)`	Checks if a column is currently filtered. Returns TRUE or FALSE.
ISCROSSFILTERED	Filter	`=ISCROSSFILTERED(ColumnName)`	Checks if a column is filtered due to a cross-table relationship.
LOOKUPVALUE	Filter	`=LOOKUPVALUE(ResultColumn, SearchColumn1, SearchValue1 [, SearchColumn2, SearchValue2…])`	Returns the value in a column for the row that meets search criteria.
TREATAS	Filter	`=TREATAS(TableExpression, Column1 [, Column2…])`	Applies the values of a table as filters on another table.

6. Logical

Function	Category	Formula / Syntax	Description / Use
AND	Logical	`=AND(Logical1, Logical2)`	Returns TRUE if both conditions are TRUE; otherwise FALSE.
OR	Logical	`=OR(Logical1, Logical2)`	Returns TRUE if at least one condition is TRUE; otherwise FALSE.
NOT	Logical	`=NOT(Logical)`	Reverses a logical value: TRUE becomes FALSE, and FALSE becomes TRUE.
IF	Logical	`=IF(LogicalTest, ResultIfTrue [, ResultIfFalse])`	Returns one value if a condition is TRUE, another if FALSE.
IF.EAGER	Logical	`=IF.EAGER(LogicalTest, ResultIfTrue [, ResultIfFalse])`	Similar to IF, but evaluates both outcomes regardless of condition (used in complex calculations).
IFERROR	Logical	`=IFERROR(Value, ValueIfError)`	Returns a value if no error occurs, otherwise returns an alternate value.
SWITCH	Logical	`=SWITCH(Expression, Value1, Result1 [, Value2, Result2 … [, ElseResult]])`	Checks multiple conditions and returns corresponding results; optional default.

TRUE	Logical	`=TRUE()`	Returns the logical value TRUE.
FALSE	Logical	`=FALSE()`	Returns the logical value FALSE.
COALESCE	Logical	`=COALESCE(Value1, Value2 [, Value3 …])`	Returns the first non-blank value from a list of expressions.

7. Math & Trig

Function	Category	Formula / Syntax	Description / Use
ABS	Math & Trig	`=ABS(Number)`	Returns the absolute value of a number.
ACOS	Math & Trig	`=ACOS(Number)`	Returns the arccosine (inverse cosine) of a number.
ACOSH	Math & Trig	`=ACOSH(Number)`	Returns the inverse hyperbolic cosine of a number.
ASIN	Math & Trig	`=ASIN(Number)`	Returns the arcsine (inverse sine) of a number.
ASINH	Math & Trig	`=ASINH(Number)`	Returns the inverse hyperbolic sine of a number.
ATAN	Math & Trig	`=ATAN(Number)`	Returns the arctangent (inverse tangent) of a number.
ATANH	Math & Trig	`=ATANH(Number)`	Returns the inverse hyperbolic tangent of a number.

CEILING	Math & Trig	`=CEILING(Number, Significance)`	Rounds a number up to the nearest multiple of significance.
COSH	Math & Trig	`=COSH(Number)`	Returns the hyperbolic cosine of a number.
COT	Math & Trig	`=COT(Number)`	Returns the cotangent of a number.
COTH	Math & Trig	`=COTH(Number)`	Returns the hyperbolic cotangent of a number.
DEGREES	Math & Trig	`=DEGREES(Number)`	Converts radians to degrees.
DIVIDE	Math & Trig	`=DIVIDE(Numerator, Denominator [, AlternateResult])`	Divides two numbers, optionally returning an alternate value if division by zero.
EVEN	Math & Trig	`=EVEN(Number)`	Rounds a number up to the nearest even integer.
EXP	Math & Trig	`=EXP(Number)`	Returns e raised to the power of a number.
FACT	Math & Trig	`=FACT(Number)`	Returns the factorial of a number.
FLOOR	Math & Trig	`=FLOOR(Number, Significance)`	Rounds a number down to the nearest multiple of significance.
INT	Math & Trig	`=INT(Number)`	Rounds a number down to the nearest integer.
ISO.CEILING	Math & Trig	`=ISO.CEILING(Number [, Significance])`	Rounds a number up to the nearest integer or multiple.

LN	Math & Trig	`=LN(Number)`	Returns the natural logarithm of a number.
LOG	Math & Trig	`=LOG(Number, Base)`	Returns the logarithm of a number to a specified base.
LOG10	Math & Trig	`=LOG10(Number)`	Returns the base-10 logarithm of a number.
MOD	Math & Trig	`=MOD(Number, Divisor)`	Returns the remainder after division.
ODD	Math & Trig	`=ODD(Number)`	Rounds a number up to the nearest odd integer.
PI	Math & Trig	`=PI()`	Returns the value of π.
POWER	Math & Trig	`=POWER(Number, Power)`	Raises a number to a specified power.
QUOTIENT	Math & Trig	`=QUOTIENT(Numerator, Denominator)`	Returns the integer portion of a division.
RADIANS	Math & Trig	`=RADIANS(Number)`	Converts degrees to radians.
RAND	Math & Trig	`=RAND()`	Returns a random number between 0 and 1.
RANDBETWEEN	Math & Trig	`=RANDBETWEEN(Bottom, Top)`	Returns a random integer between two numbers.
ROUND	Math & Trig	`=ROUND(Number, NumDigits)`	Rounds a number to the specified number of digits.
ROUNDDOWN	Math & Trig	`=ROUNDDOWN(Numbe r,`	Rounds a number down toward zero.

		`NumDigits)`	
ROUNDUP	Math & Trig	`=ROUNDUP(Number, NumDigits)`	Rounds a number up away from zero.
SIGN	Math & Trig	`=SIGN(Number)`	Returns 1 if positive, -1 if negative, 0 if zero.
SIN	Math & Trig	`=SIN(Number)`	Returns the sine of a number (in radians).
SINH	Math & Trig	`=SINH(Number)`	Returns the hyperbolic sine of a number.
SQRT	Math & Trig	`=SQRT(Number)`	Returns the square root of a number.
SQRTPI	Math & Trig	`=SQRTPI(Number)`	Returns the square root of (π * number).
TAN	Math & Trig	`=TAN(Number)`	Returns the tangent of a number (in radians).
TANH	Math & Trig	`=TANH(Number)`	Returns the hyperbolic tangent of a number.
TRUNC	Math & Trig	`=TRUNC(Number [, NumDigits])`	Truncates a number to an integer or specified number of decimal places.
Function	**Category**	**Formula / Syntax**	**Description / Use**
ABS	Math & Trig	`=ABS(Number)`	Returns the absolute value of a number.
ACOS	Math & Trig	`=ACOS(Number)`	Returns the arccosine (inverse cosine) of a number.

ACOSH	Math & Trig	`=ACOSH(Number)`	Returns the inverse hyperbolic cosine of a number.

8. Parent and Child Functions

Function	Category	Formula / Syntax	Description / Use
PATH	Parent/Child	`=PATH(ChildColumn, ParentColumn)`	Returns a delimited text string showing the path from a child row to its top-level parent.
PATHCONTAINS	Parent/Child	`=PATHCONTAINS(PathColumn, Value)`	Checks if a specific value exists in a path and returns TRUE/FALSE.
PATHITEM	Parent/Child	`=PATHITEM(PathColumn, Position [, ReturnAsText])`	Returns the item at a specified position in a path.
PATHITEMREVERSE	Parent/Child	`=PATHITEMREVERSE(PathColumn, Position [, ReturnAsText])`	Returns the item at a specified position from the end of a path.
PATHLENGTH	Parent/Child	`=PATHLENGTH(PathColumn)`	Returns the number of items in a path.

RELATED	Relationship	`=RELATED(ColumnName)`	Retrieves a value from a related table via an existing relationship.
RELATEDTABLE	Relationship	`=RELATEDTABLE(TableName)`	Returns a table containing all rows related to the current row.
CROSSFILTER	Relationship	`=CROSSFILTER(Column1, Column2, Direction)`	Changes the direction of cross-filtering for a relationship.

9. Statistical

Function	Category	Formula / Syntax	Description / Use
BETA.DIST	Statistical	`=BETA.DIST(x, Alpha, Beta [, Cumulative [, A, B]])`	Returns the beta probability density or cumulative distribution function.
BETA.INV	Statistical	`=BETA.INV(Probability, Alpha, Beta [, A, B])`	Returns the inverse of the beta cumulative distribution.
CHISQ.DIST	Statistical	`=CHISQ.DIST(x, DegreesFreedom, Cumulative)`	Returns the chi-squared distribution.
CHISQ.DIST.RT	Statistical	`=CHISQ.DIST.RT(x, DegreesFreedom)`	Returns the right-tailed chi-squared distribution.

CHISQ.INV	Statistical	=CHISQ.INV(Probability, DegreesFreedom)	Returns the inverse of the chi-squared distribution.
CHISQ.INV.RT	Statistical	=CHISQ.INV.RT(Probability, DegreesFreedom)	Returns the inverse of the right-tailed chi-squared distribution.
CONFIDENCE.NORM	Statistical	=CONFIDENCE.NORM(Alpha, StandardDev, Size)	Returning the confidence interval for a population mean using a normal distribution.
CONFIDENCE.T	Statistical	=CONFIDENCE.T(Alpha, StandardDev, Size)	Returning the confidence interval for a population mean using a t-distribution.
EXPON.DIST	Statistical	=EXPON.DIST(x, Lambda [, Cumulative])	Returns the exponential distribution.
F.DIST	Statistical	=F.DIST(x, DegFreedom1, DegFreedom2, Cumulative)	Returns the F probability distribution.
F.DIST.RT	Statistical	=F.DIST.RT(x, DegFreedom1, DegFreedom2)	Returns the right-tailed F probability distribution.
F.INV	Statistical	=F.INV(Probability, DegFreedom1, DegFreedom2)	Returns the inverse of the F probability distribution.
F.INV.RT	Statistical	=F.INV.RT(Probability, DegFreedom1,	Returns the inverse of the right-tailed F distribution.

		DegFreedom 2)	
GAMMA	Statistical	=GAMMA(x)	Returns the gamma function value for x.
GAMMA.DIST	Statistical	=GAMMA.DIST(x, Alpha, Beta, Cumulative)	Returns the gamma probability density or cumulative distribution.
GAMMA.INV	Statistical	=GAMMA.INV(Probability, Alpha, Beta)	Returns the inverse of the gamma cumulative distribution.
GAMMALN	Statistical	=GAMMALN(x)	Returns the natural logarithm of the gamma function.
GAMMALN.PRECISE	Statistical	=GAMMALN.PRECISE(x)	Returns the precise natural logarithm of the gamma function.
GAUSS	Statistical	=GAUSS(x)	Returns the probability that a value is within a standard normal distribution.
LOGNORM.DIST	Statistical	=LOGNORM.DIST(x, Mean, StdDev, Cumulative)	Returns the log-normal probability density or cumulative distribution.
LOGNORM.INV	Statistical	=LOGNORM.INV(Probability, Mean, StdDev)	Returns the inverse of the log-normal cumulative distribution.
NORM.DIST	Statistical	=NORM.DIST(x, Mean, StdDev, Cumulative)	Returns the normal probability density or cumulative distribution.

NORM.INV	Statistical	`=NORM.INV(Probability, Mean, StdDev)`	Returns the inverse of the normal cumulative distribution.
NORM.S.DIST	Statistical	`=NORM.S.DIST(z, Cumulative)`	Returns the standard normal distribution.
NORM.S.INV	Statistical	`=NORM.S.INV(Probability)`	Returns the inverse of the standard normal distribution.
POISSON.DIST	Statistical	`=POISSON.DIST(x, Mean, Cumulative)`	Returns the Poisson probability mass or cumulative distribution.
TDIST	Statistical	`=TDIST(x, DegreesFreedom, Tails)`	Returns the Student's t-distribution.
TINV	Statistical	`=TINV(Probability, DegreesFreedom)`	Returns the inverse of the Student's t-distribution.
T.DIST	Statistical	`=T.DIST(x, DegreesFreedom, Cumulative)`	Returns the Student's t-distribution (cumulative).
T.DIST.2T	Statistical	`=T.DIST.2T(x, DegreesFreedom)`	Returns the two-tailed Student's t-distribution.
T.DIST.RT	Statistical	`=T.DIST.RT(x, DegreesFreedom)`	Returns the right-tailed Student's t-distribution.
T.INV	Statistical	`=T.INV(Probability, DegreesFreedom)`	Returns the inverse of the Student's t-distribution.
T.INV.2T	Statistical	`=T.INV.2T(Probability, DegreesFreedom)`	Returns the inverse two-tailed Student's t-distribution.

WEIBULL.DIST	Statistical	`=WEIBULL.DIST(x, Alpha, Beta, Cumulative)`	Returns the Weibull probability density or cumulative distribution.
Z.TEST	Statistical	`=Z.TEST(Array, X, Sigma)`	Returns the one-tailed probability-value of a z-test.

10. Table Functions

Function	Category	Formula / Syntax	Description / Use
ADDCOLUMNS	Table	`=ADDCOLUMNS(Table, "ColumnName", Expression)`	Adds calculated columns to a table.
ADDMISSINGITEMS	Table	`=ADDMISSINGITEMS(Table)`	Adds rows for missing items in a table based on existing hierarchies.
CROSSJOIN	Table	`=CROSSJOIN(Table1, Table2 [, Table3 …])`	Returns the Cartesian product of two or more tables.
DATATABLE	Table	`=DATATABLE(Column1, DataType1, …, { {Value1_1, …}, {Value2_1, …} })`	Creates a table from specified columns and data.

EXCEPT	Table	`=EXCEPT(Table1, Table2)`	Returns rows in Table1 that do not exist in Table2.
FILTER	Table	`=FILTER(Table, Condition)`	Returns a table containing only rows that satisfy a condition.
GENERATE	Table	`=GENERATE(Table1, Table2)`	For each row in Table1, returns all rows from Table2.
GENERATEALL	Table	`=GENERATEALL(Table1, Table2)`	Similar to GENERATE, but ignores existing filters on Table1.
GROUPBY	Table	`=GROUPBY(Table, Column1 [, Column2…], Name, Expression)`	Groups a table by columns and optionally adds aggregated expressions.
INTERSECT	Table	`=INTERSECT(Table1, Table2)`	Returns rows that exist in both tables.
NATURALINNERJOIN	Table	`=NATURALINNERJOIN(Table1, Table2)`	Returns a table of rows matching common columns in both tables.

NATURALLEFT OUTERJOIN	Table	`=NATURALLEFTOUTERJOIN(Table1, Table2)`	Returns all rows from Table1 with matching rows from Table2; unmatched Table2 values are blank.
ROLLUP	Table	`=ROLLUP(Table, Column1 [, Column2 …])`	Returns a table with subtotals and totals for columns.
ROLLUPADDISSUBTOTAL	Table	`=ROLLUPADDISSUBTOTAL(Table, Column1 [, Column2 …])`	Adds an "IsSubtotal" flag to a ROLLUP table.
ROLLUPGROUP	Table	`=ROLLUPGROUP(Table, Column1 [, Column2 …])`	Groups a table for rollup aggregation.
SELECTCOLUMNS	Table	`=SELECTCOLUMNS(Table, "ColumnName", Expression [, …])`	Returns a table with selected columns and optionally calculated columns.
SUMMARIZE	Table	`=SUMMARIZE(Table, Column1 [, Column2 …] [, Name, Expression …])`	Groups a table by columns and calculates aggregates.
SUMMARIZECOLUMNS	Table	`=SUMMARIZECOLUMNS(Column1 [, Column2 …] [, Filter1 …])`	Groups a table by columns and evaluates expressions in a filter context.

UNION	Table	`=UNION(Table1, Table2 [, Table3 …])`	Returns a table with all rows from multiple tables.
VALUES	Table	`=VALUES(Column)`	Returns a one-column table of unique values from a column.
TOPN	Table	`=TOPN(N, Table, OrderBy_Expression [, Order])`	Returns the top N rows of a table based on sorting.
ORDERBY	Table	`=ORDERBY(Table, OrderBy_Expression [, Order])`	Returns a table sorted by one or more expressions.

11. Miscellaneous

Function	Category	Formula / Syntax	Description / Use
BLANK	Information	`=BLANK()`	Returns a blank value.
ISBLANK	Information	`=ISBLANK(Value)`	Checks if a value is blank; returns TRUE/FALSE.
ISERROR	Information	`=ISERROR(Value)`	Checks if a value returns an error; returns TRUE/FALSE.

IFERROR	Information	`=IFERROR(Value, ValueIfError)`	Returns ValueIfError if an error occurs; otherwise returns Value.
ISNUMBER	Information	`=ISNUMBER(Value)`	Checks if a value is a number; returns TRUE/FALSE.
ISTEXT	Information	`=ISTEXT(Value)`	Checks if a value is text; returns TRUE/FALSE.
ISLOGICAL	Information	`=ISLOGICAL(Value)`	Checks if a value is TRUE/FALSE; returns TRUE/FALSE.
ISNONTEXT	Information	`=ISNONTEXT(Value)`	Checks if a value is not text; returns TRUE/FALSE.
ISINSCOPE	Information	`=ISINSCOPE(ColumnName)`	Checks if a column is in the current visual context; returns TRUE/FALSE.
HASONEVALUE	Information	`=HASONEVALUE(ColumnName)`	Returns TRUE if there is only one value in a column in the current context.
HASONEFILTER	Information	`=HASONEFILTER(ColumnName)`	Returns TRUE if a column has only one filter applied.

SELECTEDVALUE	Information	`=SELECTEDVALUE(Column [, AlternateResult])`	Returns the single value of a column if only one is selected; otherwise returns AlternateResult.
LOOKUPVALUE	Information	`=LOOKUPVALUE(ResultColumn, SearchColumn1, SearchValue1 [, …])`	Returns a value in a column for the row that meets specified search criteria.
PATH	Information	`=PATH(ChildColumn, ParentColumn)`	Returns a delimited text string showing the path from a child row to its top-level parent.
PATHITEM	Information	`=PATHITEM(PathColumn, Position [, ReturnAsText])`	Returns the item at a specific position in a path.
PATHCONTAINS	Information	`=PATHCONTAINS(PathColumn, Value)`	Returns TRUE if the path contains a specified value.
PATHLENGTH	Information	`=PATHLENGTH(PathColumn)`	Returns the number of items in a path.
RELATED	Information	`=RELATED(ColumnName)`	Retrieves a value from a related table

			via an existing relationship.
RELATEDTABLE	Information	`=RELATEDTABLE(TableName)`	Returns a table containing all rows related to the current row.

We have included a range of additional links to further resources and information that you may find useful, these can be found within your revision guide.

END OF WORKBOOK

Please check through your work thoroughly before submitting and update the table of contents if required.

Please send your completed work booklet to your trainer.