PROPOSAL

The goal of the project is to provide Adventure Works executives with an understanding of the data in relation to the questions detailed below. A brief analysis was conducted on the Adventure Works 2019 database. All the data was initially gathered and analysed using Microsoft SQL Server Management Studio, the results were saved as individual CSV files with were cleaned for visual purposes and imported into Visual studio code using the following python libraries: pandas, matplotlib, and seaborn for data visualisation and further analysis.

The portfolio includes this report with the full outline of the methodology, along with a SQL text file used to obtain the relevant data, a python script used to visualise the data, and a PowerPoint presentation providing a clear understanding of the data in a digestible format.

OVERVIEW

The Adventure Works 2019 is an OLAP database that uses a normalised snowflake schema to logically organise data per department, such as:

- Human Resources: storing employee, department, and job-related information such as holidays, sick leave, and salary.
- Production: storing product information, manufacturing, and inventory.
- Sales: storing customers, bonuses, sales orders, and sales-related information.
- Purchasing: storing vendor-related data.

For this project, we're focused on Human Resources, Sales, and Person, and included the materialised views such as Sales.vStoreWithDemographics' with additional user permission on additional information necessary to answer the questions, such as store size and number of employees.

The table structure in the database included primary keys and foreign keys such as Business Entity IDs.

The relationship entities within the database are:

- One-to-many relationship: SalesPersonID > SalesOrderID
- Many-to-many: BusinessEntityID > ShiftID

REPORT

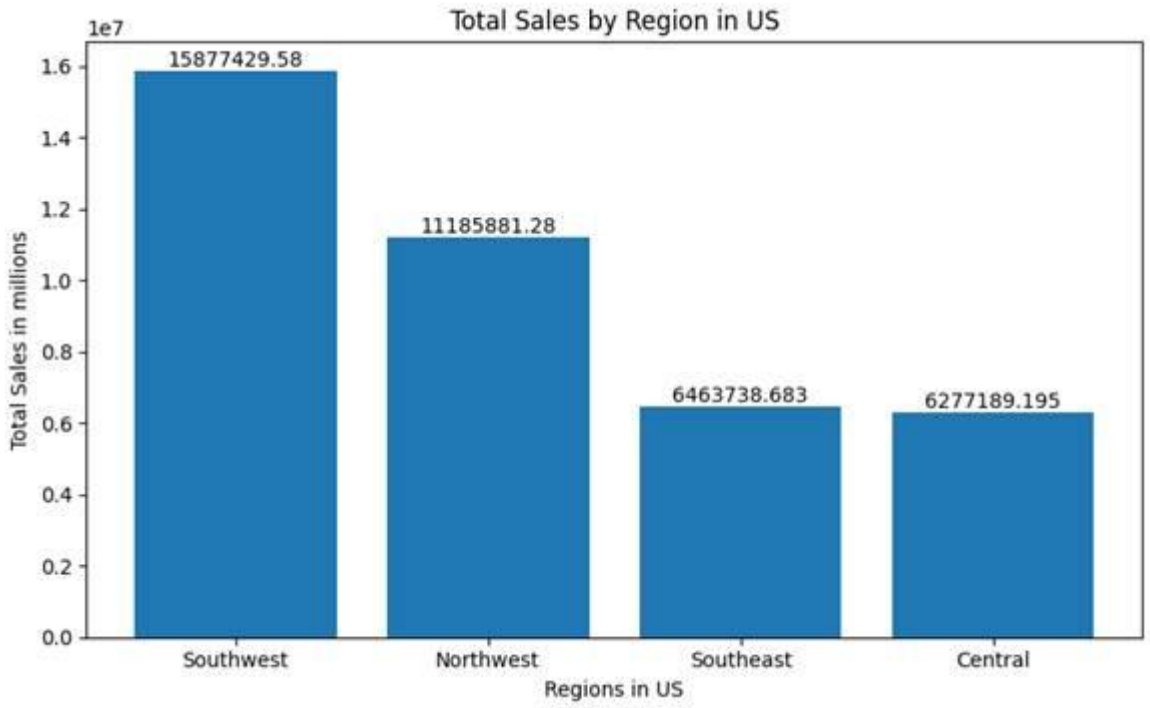
Q1. What are the regional sales in the best-performing country?

This question had two parts that was, to determine the best performing country then the highest sales in the region of that country.

For the first query, we selected the 'CountrlapyRegioncode', 'Sales YTD' (year to date), 'Sales Last Year', and 'Name' as the alias country_name from the 'Sales.SalesTerritory Table'. The MAX() of the sales YTD and sales last year was summed up as the 'total_sales', the 'Name' was 'countryregioncode' were placed in the group by function. This query was further completed using the Order by function of the total-sales in descending order to determine the highest sales.

The result from the first query was used in a second query similar to the first query, this time adding the 'WHERE' function to specify the best performing country to find the regional sales, which is shown in the plot below.

The table for the second answer was imported into visual studio code using the pandas function 'pd.read_csv()' to input the data. A bar chart was chosen for this question as it clearly shows the highest performing regions and the total sales.

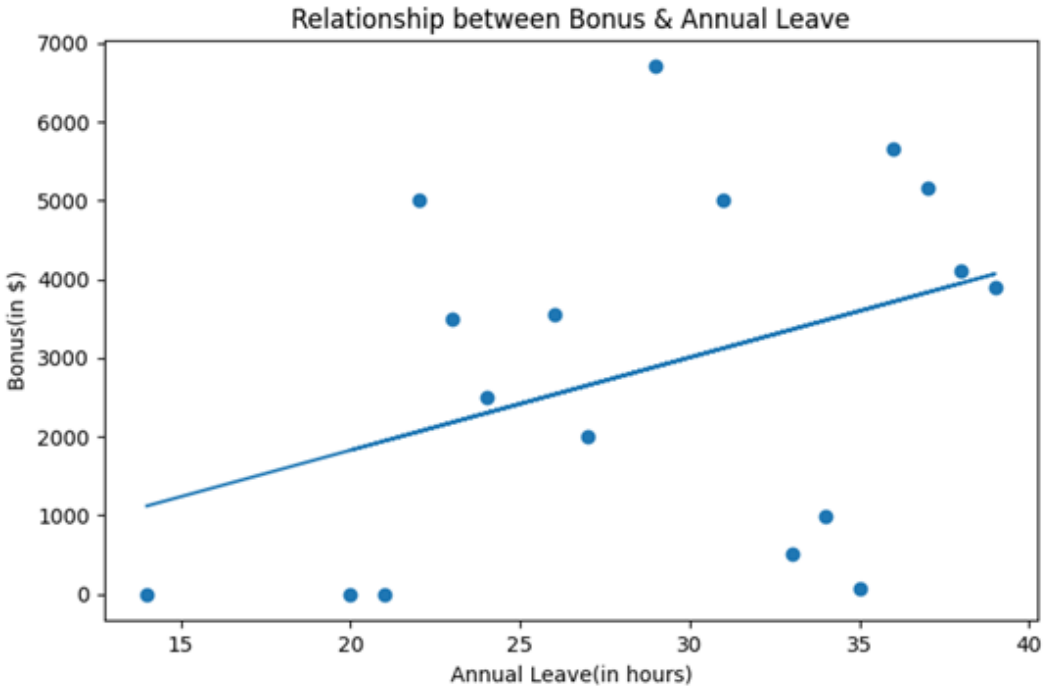


Q2. What is the relationship between annual leave and bonus?

First was to look through tables in the database, mainly the human resources tables, to find the suitable columns; however, the two columns in different tables this meant the 'INNER JOIN' function had to be used.

The ‘VacationHours’ column, aliased as ‘annual_leave’ was selected from the ‘HumanResources.Employee’ table and the ‘bonus’ column from the ‘Sales.SalesPerson’. An inner join was used on ‘BusinessEntityID’, which is a common column on both tables. The result of this query produced a table with matching records.

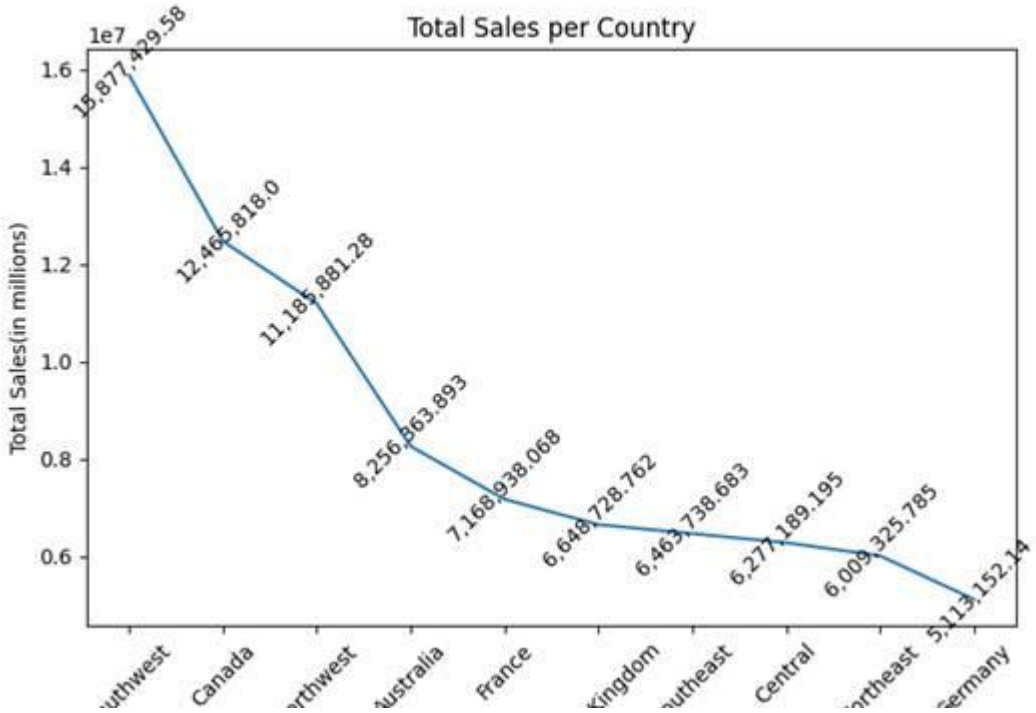
A scatter plot from the matplotlib library was created to show the different points in the annual leave and bonus; a trend line was added to the plot to display the relationship between annual leave hours and bonuses.



Q3. What is the relationship between country and revenue?

The sum of ‘Sales YTD’ and ‘SalesLastYear’ was calculated to find the revenue as total_sum. These columns were selected with the ‘Name’ column from the ‘Sales.SalesTerritory’ table. The selected columns were used in the ‘Group by’ function. The total_sum was placed in descending order using the ‘Order by’ to start with the country with the highest revenue.

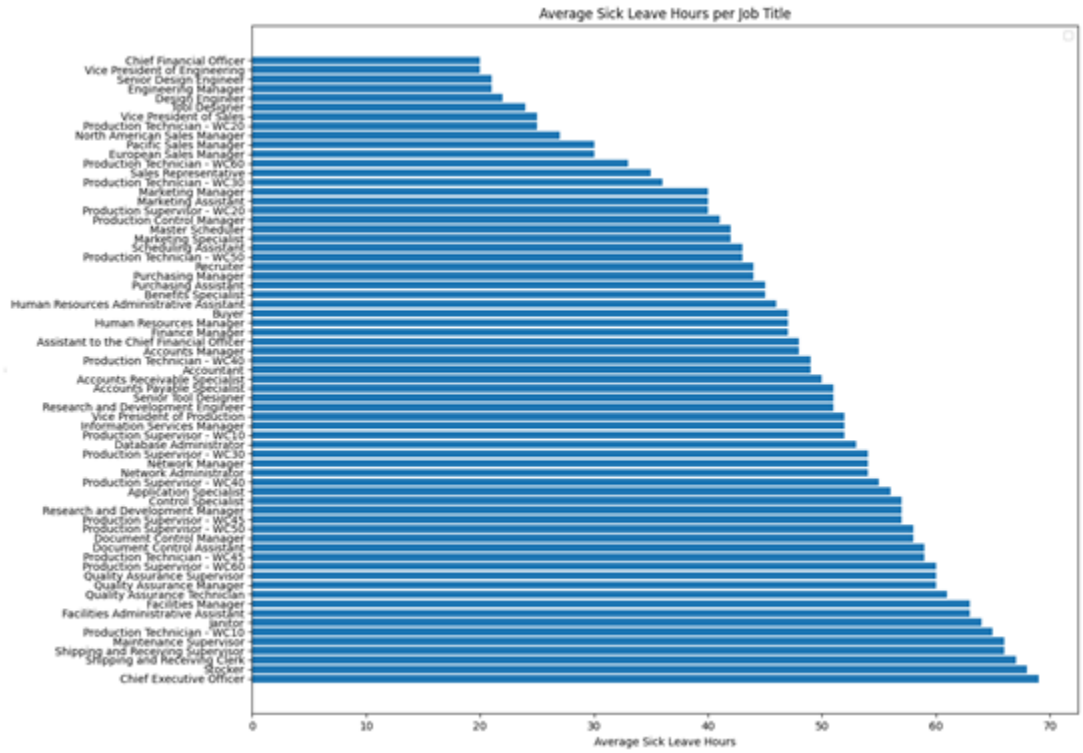
For visualisation a line chart was created in Visual Studio code using the name and the total_sum. The value of each country’s revenue was displayed on the graph.



Q4. What is the relationship between sick leave and Job Title (PersonType)?

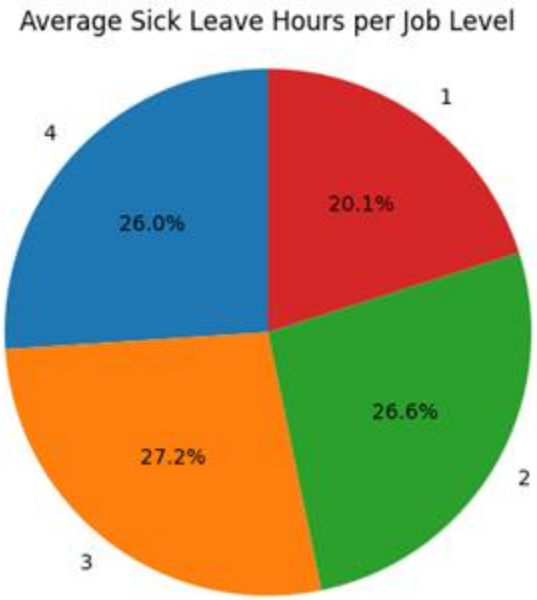
To obtain the results of this query, we obtained the information from the 'HumanResources.Employee' table. Firstly, we selected the two columns, 'Job title' and 'SickLeaveHours', using an aggregate function 'DISTINCT' to return unique values, which provided us with the desired results. In a SQL code, we ordered using 'SickLeaveHours' in descending order. We exported the results in CSV file, cleaned and sorted the data by adding the appropriate headers using Excel. To visualise the data, we used python by importing libraries such as pandas, numpy, and matplotlib.pyplot. After importing the CSV file, assigning the header as 0, and adding it as a variable 'df'.

For a smooth background, we used the figure function and assigning a size of 8,5. Using the bar in horizontal style to show the relationship between job titles and sick leave is a clear and effective way to show which job positions are more likely to use sick days in a year. For example, in the chart below, it's clear that the Chief Executive Officer is the position with the sick days taken with a total of 69 days, while the Chief Financial Officer is the sick days, totalling 20 days.



To further answer the question, we created a second chart that relates to the organisational level along with the sick days to understand the distribution of sick leave between organisational levels (Operational, Middle, Top, and Strategic). The approach was also within the 'HumanResources.Employee' table, selecting the AVG aggregate function on 'SickLeaveHours' and adding an alias to ensure the column has a name. I ensured that the group by and order by are by 'OrganizationLevel' to ensure they are grouped together. As the Chief Executive Officer has no value on the organisation level, I included a WHERE function that excludes any null values by choosing 'IS NOT NULL'. In Python, following a similar structure by using the variable 'df' to import the CSV file, and ensuring the header is labelled 0. Included a figure to provide a white background and chose matplotlib.pyplot pie using 'plt.pie'.

We chose the 'SickLeaveHours' column, and for the labels chose the 'OrganizationLevel', ensured it displayed as percentages for readability by choosing the autopct function and choosing a 1 decimal number, and started at a 90-degree angle. Finalised the chart by including the title and axis. A pie chart was chosen to represent this data as it shows that the percentage of sick leave amongst organisational levels is distributed quite evenly.



Q5. What is the relationship between store trading duration and revenue?

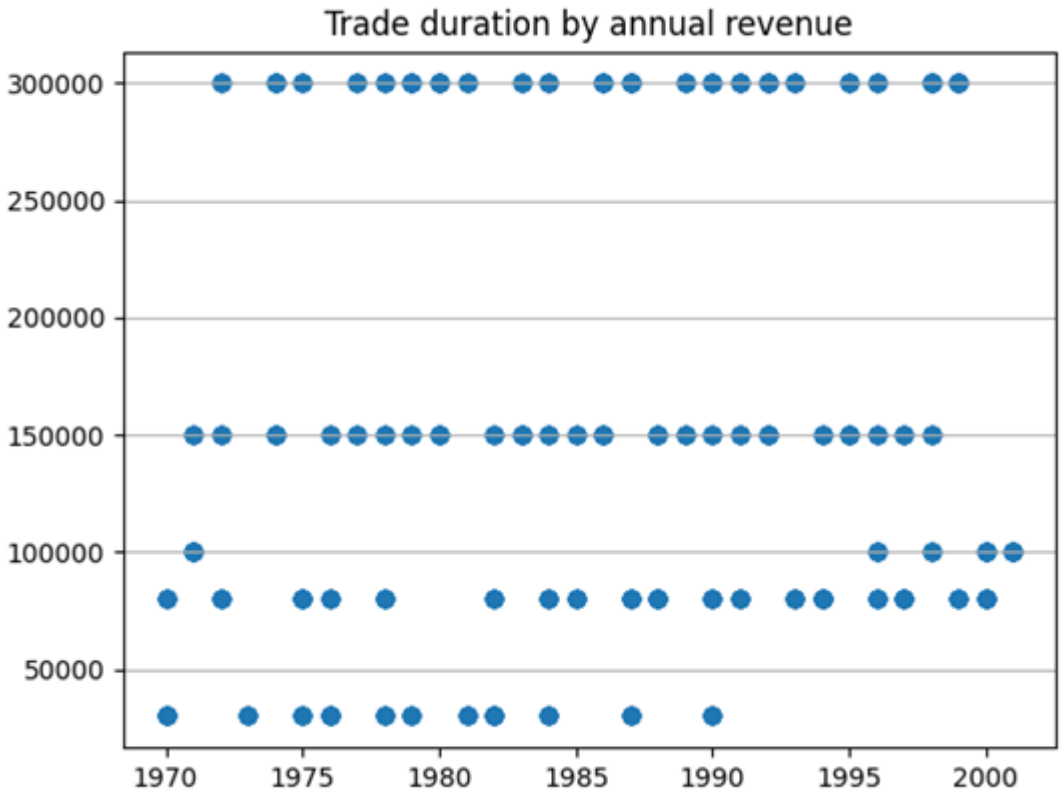
For a time-efficient approach, rather than creating joins to obtain all of the information to approach this question, we chose to select an already stored view called 'Sales.vStoreWithDemographics'. In the select clause we included 'Name, YearOpened, AnnualRevenue, 2019 (the current year of the database) minus 'YearOpened' to obtain the trade duration, which is what we aliased the new column. The results were the name of the store, the year it opened, annual revenue, and how long they've been opened.

Next, we included a group by clause for name, year opened, and annual revenue, and ordered it by 'trade_duration' in descending order, which returned all the necessary results for the visualisation in the order of the store opened for the longest to the shortest. In python we followed the similar steps as above to import the CSV file and gave it a variable of 'df' followed by an empty list of years. We started with an if, elif, and else code that separates the years into four categories:

- Before 1980
- Between 1980 and 1990
- Between 1990 and 200
- All other years thereafter

We then created a scatter plot using matplotlib with the 'YearOpened' column and the Annual Revenue column to understand if there is a correlation between them. Followed by 'plt.grids' for an easier read and included a title. We chose a scatter plot since we wanted to see all of the data points in the chart to understand if there's a positive,

negative, or no correlation. According to the chart, there's no positive or negative correlation between the two variables.



Q6. What is the relationship between the size of the stores, the number of employees, and revenue?

The required information to answer this question accurately isn't detailed on the tables included in the database, but rather in the views already stored called 'Sales.vStoreWithDemographics'. This is what we chose as the from function rather than a table or joins. Followed by selecting the 'SquareFeet' column aliased as store_size_sqfeet, 'AnnualRevenue', and 'NumberEmployees', this was then ordered by the number of employees. Following the same pattern as above, by extracting it as a CSV and importing it to Python for a visualisation. In this specific example, we adopted a scatter bubble plot to show the correlation between store size, number of employees, and annual revenue. We aliased the data frame as 'df', and made an empty list named 'color'. In this list, we used the statement, if, elif, and else with 'color.append' to coordinate the sizes of the stores with their respective colour as follows:

- If the revenue is smaller than $99,999, it'll be assigned a red colour
- If the revenue is between $100,000 and $199,999, it'll be assigned orange
- If the revenue is above $200,000, it'll be assigned blue

Next, we created the scatter plot, assigned the x axis (number of employees) and y axis (Store size), including an extra variable ‘s’ which is an np.array for the Annual Revenue divided by 100 for visibility on the chart. Nonetheless, this will not skew the relationship. We assigned the variables z and p to np.polyfit to the number of employees, and store.

A bubble chart displays the size of the store in size relative to each other plotted in the graph. While the x axis displays the number of employees, and y axis displays the annual revenue.