https://github.com/PratikshaMayekar/Alteryx.git

Question 2 Answer:

Alteryx compares to SQL

Alteryx is simplistic workflow-based environment that allows you to prepare, blend and analyze your data regardless of how many various structured data sources you have included. I am giving comparison for the tools I used in Alteryx Exercise. SQL allows us to blend data as Alteryx does, but SQL along with python or any procedural language gives more flexibility to interconnect operations. Whereas in Alteryx provides simple drag and drop tool to connect between the multiple operations. In Alteryx, with workflow design anyone can easily understand entire program whereas in SQL you need read entire code.
Alteryx is very easy tool to learn compared to SQL

Accessibility by coders and non-coders, in contrast to Excel and Python.

Alteryx is very user friendly for compared to Excel and Python.
Alteryx require minimal coding and we can see the flow of data during the process and immediately make adjustments without having to wait for the output. Python perform better in machine learning, API connections, and automation.

Excel struggles with advanced data manipulations and transformations. Tasks like fuzzy matching, address hygiene, missing data, and more all require plugins, formulas or are impossible to do in Excel

Auditability & traceability. How can Alteryx facilitate trust internally and externally (government auditors) that data processing is reliable and transparent?

Alteryx has Transparency, Repeatability, Operationalize, Scalability & Advanced Analytics.
Alteryx is auditable at each step, so the data processing is reliable and transparent.
With Input and output button on each tool allows you to trace blended data very easily.
Debugging is very easy compared to SQL and python for large program.

Readme :

1. Converted csv files of four tables provided for capstone project
2. Input Data => four csv files

Sales_week table

Datatime tool => Converted "Date" column to date data type with new column "Start Date"
Multifield formula => Added "w" in week0 column
Multifield formula => Added new filed for End date
Multifield formula => Added Quarter inplace of quarter number

Sales_history table

Transpose => transpose weekly quantity in tabular format
Filter => removed all records with "0" quantities
Select => fixed column alias as required for final output and fixed datatype to use min memory

Sales Team

TextToColumn => Split "Sales Team Lead" column to firstname and latname
DataCleansing => Removed leading space from FirstName
AutoField => fixed data types and column sizes
select => removed sales team lead and isfound column

Sales_product

Product_code table

Transpose => converted product table in tabular format to get price of each quarter for the given item code

AutoField => fixed column sizes and datatypes

select => product code information with correct alias

ESP_code table

AutoField => fixed column sizes and datatypes

select => ESP information selected from product table

Joins

Join1 : Sales_week and Sales_history table on (W# and Sales_year)
Join2 : Join1 and Sales_team table on (empID)
Join3 : Join2 and prod_code on (ItemCode and Sales_quarter)
Join4 : Join2 and ESP_code on (ItemCode)

Question1 Demo

Merge tool => Join3 + Join4
Multifield formula => calculate total price (Qty * price)
datetime tool => convert start and end date to String and correct Alias
select => Question1 final output with the correct column order and names

Question 3

Multifield formula => calculate total price (Qty * price) for ESP records
select => Question3 final output with the correct column order and names

Summary of ESP_CODE

Summarize => Group by ESP_CODE,price (sum of Qty and sum of ESP_total$)
Summarize => Summarize ESP_total$