databricks Milestone

(https://databricks.com)

Instruction :

You are a data scientist working for a data analytics firm. Your firm has explored a multitude of data sources and is tasked with providing key insights that your clients can make actionable. Your manager has asked you to provide some data analytics guidance for one of the firm's clients.

In a typical scenario, you would iteratively work with your client to understand the data wanting to be analyzed. Having a solid understanding of the data and any underlying assumptions present is crucial to the success of a data analysis project. However, in this case, you will need to do a little more of the "heavy lifting".

To begin, you will prepare a project proposal detailing:

- The questions we are wanting to answer,
- initial hypothesis about the data relationships, and
- the approach you will take to get your answers.

NOTE: The proposal is just a plan for how we will travel. It's there to help keep you on your path by keeping the end goal in mind. You will then will execute your plan and in the end present your findings in a month to your management.

Milestone 1: Project Proposal and Data Selection/Preparation

Step 1 : Preparing for Your Proposal

You will document your preparation in developing the project proposal. This includes:

1. Which client/dataset did you select and why?
2. Describe the steps you took to import and clean the data.
3. Perform initial exploration of data and provide some screenshots or display some stats of the data you are looking at.
4. Create an ERD or proposed ERD to show the relationships of the data you are exploring.

1) Which client/dataset did you select and why?

The client/dataset I have selected for this milestone is SportsStats. My rationale for this selection is twofold. Firstly, I have a personal affinity for sports, which I consider not only a favorite pastime but also an area where I possess substantial foundational knowledge. Secondly, the nature of the data provided by SportsStats aligns well with my interests and expertise.

To provide some context / information about the client : SportsStats is a sports analysis firm partnering with local news and elite personal trainers to provide "interesting" insights to help their partners. Insights could be patterns/trends highlighting certain groups/events/countries, etc. for the purpose of developing a news story or discovering key health insights.

- I believe working with the SportsStats dataset will yield meaningful and engaging results.

2) Describe the steps you took to import and clean the data.

The initial approach I prefer to employ for this dataset involves utilizing Python packages, specifically Data Wrangler and ProfileReport. These tools facilitate a comprehensive understanding of the dataset and streamline the data cleaning process. Below are the steps I intend to take to accomplish this.

- However, it's important to note that the final deliverable for this milestone is SQL-based, and as such, the final implementation will be conducted using MySQL.

Python:

Import

Import the packages and the dataset

```
#packages
import pandas as pd
import matplotlib
import numpy as np
matplotlib.use('module://ipykernel.pylab.backend_inline')
from ydata_profiling import ProfileReport
from sqlalchemy import create_engine as ce
from sqlalchemy import inspect

# Set the display option
pd.set_option('display.max_colwidth', None)


# Define the data types
data_types = {
    'ID': 'Int64',
    'Name': str,
    'Sex': str,
    'Age': 'Int64',
    'Height': 'Int64',
    'Weight': float,
    'Team': str,
    'NOC': str,
    'Games': str,
    'Year': 'Int64',
    'Season': str,
    'City': str,
    'Sport': str,
    'Event': str,
    'Medal': str
}

# Read the CSV file
athlete_events = pd.read_csv('athlete_events.csv', dtype=data_types, quotechar='"', delimiter=',')
noc_regions = pd.read_csv("noc_regions.csv", quotechar='"', delimiter=',')
```

Merge the two datasets and reorder the columns.

```
# merge the data
df1 = pd.merge(athlete_events, noc_regions, on='NOC', how='left')

# re order the columns
df1 = df1[['ID', 'Name', 'Sex', 'Age', 'Height', 'Weight', 'Team', 'NOC', 'region', 'notes', 'Games', 'Year', 'Season', 'City', '

# rename the 'region' and 'notes' columns to 'Region' and 'Notes'
df1 = df1.rename(columns={'Name': 'Name_max','region': 'Region', 'notes': 'Notes'})
```

```
#create a copy of the df1
df = df1.copy()
```

```
# dl the new data created to use it in sql
df.to_csv('athlete_events_region.csv', index=False,header=True)
```

Cleaning

```
df.head()
```

	ID	Name_max	Sex	Age	Height	Weight	Team	NOC	Region	Notes	Games	Year	Season	City	Sport	Event
0	1	A Dijiang	M	24	180	80.0	China	CHN	China	NaN	1992 Summer	1992	Summer	Barcelona	Basketball	Basketball Men's Basketball
1	2	A Lamusi	M	23	170	60.0	China	CHN	China	NaN	2012 Summer	2012	Summer	London	Judo	Judo Men's Extra-Lightweight
		Gunnar									1920					Football

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 271116 entries, 0 to 271115
Data columns (total 17 columns):
 #   Column    Non-Null Count   Dtype
---  ------    --------------   -----
 0   ID        271116 non-null  Int64
 1   Name_max  271116 non-null  object
 2   Sex       271116 non-null  object
 3   Age       261642 non-null  Int64
 4   Height    210945 non-null  Int64
 5   Weight    208241 non-null  float64
 6   Team      271116 non-null  object
 7   NOC       271116 non-null  object
 8   Region    270746 non-null  object
 9   Notes     5039 non-null    object
 10  Games     271116 non-null  object
 11  Year      271116 non-null  Int64
 12  Season    271116 non-null  object
 13  City      271116 non-null  object
 14  Sport     271116 non-null  object
 15  Event     271116 non-null  object
```

It appears that we have various <NA> and NaN values present in the DataFrame. These will be replaced with 'null' values to facilitate easier data cleaning in subsequent steps.

```
# replace
# df = df.replace({np.nan: None})
```

Upon initial examination of the DataFrame, it appears that the 'Game' column is a concatenation of the 'Year' and 'Season' columns. If this is indeed the case, the 'Game' column will be deemed redundant and subsequently removed. A thorough check will be conducted to confirm this observation.

```
# split the Game column
df[['Year2', 'Season2']] = df['Games'].str.split(' ', expand=True)
```

```
#convert the Year2 column
df = df.astype({"Year2": 'Int64'})
```

```
#check if this 2 columns are the same
df['Year'].equals(df['Year2'])
```

```
True
```

```
#check if this 2 columns are the same
df['Season'].equals(df['Season2'])
```

```
True
```

As confirmed, the 'Game' column is indeed redundant as it is a concatenation of the 'Year' and 'Season' columns. Therefore, it will be appropriately removed to maintain the efficiency and relevance of the dataset.

```
#drop the columns
df = df.drop(columns=['Games', 'Year2', 'Season2'])
```

The names of some athletes are too long, so they will be trimmed to only Two.

```
# check the Name length
df['Name_length'] = df['Name_max'].str.len()
```

```
# print the lenght of name to have a better view
df['Name_length'].unique()
```

```
array([  9,   8,  19,  20,  24,  15,  12,  34,  16,  30,  18,  17,  23,
        21,  25,  13,  11,  28,  22,  10,  14,  39,  26,  32,  27,  40,
        33,  41,  29,  35,  36,  38,  37,  31,  51,  44,   4,   7,  59,
        43,  48,  49,   6,  42,  50,  47,  58,  46,  45,   5,  53,  63,
        55,  56,  70,   3,  57,  52,  54,  68,  61,   2,  62,  79,  60,
        69,  65,  74, 100,  64,  78,  75,  82,  88,  85,  67,  93,  66,
       108,  97], dtype=int64)
```

```
df[df['Name_length'] > 100]
```

ID	Name_max	Sex	Age	Height	Weight	Team	NOC	Region	Notes	Year	Season	City	Sport	Event	Medal
	Max Emanuel														

```
# create a new columns and store only max 2 name of each athletes
df['Name'] = df['Name_max'].str.split().str[:2].str.join(' ')
```

check the len of the Event column

```
# check the Name length
df['event_max'] = df['Event'].str.len()

# print the lenght of name to have a better view
df['event_max'].unique()
```

```
array([27, 28, 23, 32, 34, 40, 51, 50, 38, 35, 26, 31, 30, 22, 24, 49, 25,
       39, 46, 45, 20, 21, 42, 33, 43, 19, 36, 29, 37, 47, 41, 44, 59, 62,
       58, 60, 18, 48, 54, 55, 65, 69, 57, 56, 61, 17, 52, 15, 68, 53, 63,
       85], dtype=int64)
```

The length is a bit long, but it is okay.

Drop the column that we added to understand the data. We don't need it anymore.

```
#drop the columns
df = df.drop(columns=['event_max', 'Name_max', 'Name_length'])
```

Check duplicate and missing values

```
# Check for duplicates
print('1 ) -Shape of dataframe:', df.shape)
print('2 ) -Total Duplicates:', df.duplicated().sum())
print('There are many duplicate values in the data, but it's unclear whether they are truly duplicates.')

# Check for missing values in dataframe
print('4 ) -Total count of missing values:', df.isna().sum().sum())
print('')
# Display missing values per column in dataframe
print('5 ) -Missing values per column:')
df.isna().sum()
```

```
1 ) -Shape of dataframe: (271116, 18)
2 ) -Total Duplicates: 1385
3 ) -Shape of dataframe with duplicates dropped: (269731, 18)
4 ) -Total count of missing values: 630300

5 ) -Missing values per column:
ID                  0
Name_max            0
Sex                 0
Age              9474
Height          60171
Weight          62875
Team                0
NOC                 0
Region            370
Notes          266077
Year                0
Season              0
City                0
Sport               0
Event               0
Medal          231333
Name_length         0
Name                0
dtype: int64
```

There are many duplicate values in the data, but it's unclear whether they are truly duplicates. This is because certain sports have numerous events that occur consecutively. Sports such as 'Art Competitions', 'Cycling', 'Sailing', and 'Equestrianism' are examples of this.

It appears that there are numerous missing values in your DataFrame. At this stage of the project, it's unclear whether these missing values will impact the results. Therefore, we will retain them for now. If we find that they significantly affect our analysis, we will address these missing values accordingly.

Pivot Table of the Sport and sex by age

```
#pivot_table = df.pivot_table(values='Age', index='Sex', columns='Sport', aggfunc='median')
#pivot_table
```

Fill the missing values with the Median

- change the code to do the median for each column this code fill with the age in the height and weight

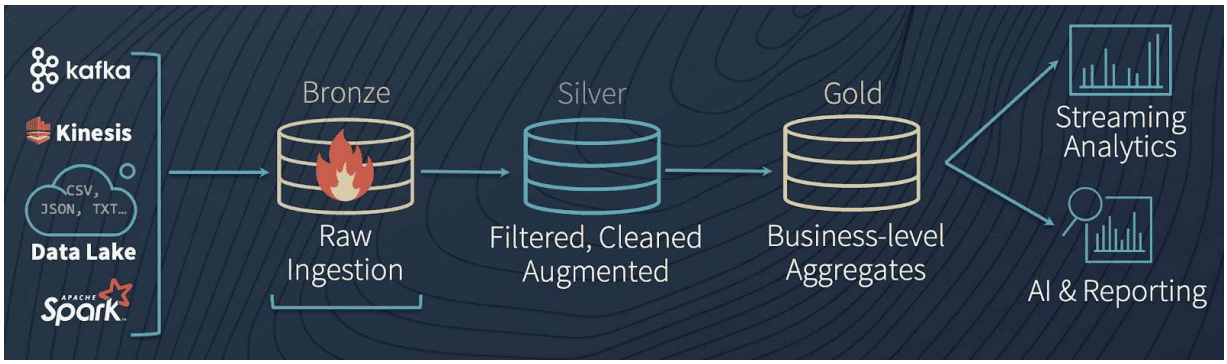
```
#cols_to_fill = ['Age', 'Height', 'Weight']

# Convert columns to numeric
#for col in cols_to_fill:
#    df[col] = pd.to_numeric(df[col], errors='coerce')

# Now you can fill NA values with the mean
#mean_values = df.groupby(['Sex', 'Sport'])[cols_to_fill].transform('median')
#df[cols_to_fill] = df[cols_to_fill].fillna(mean_values)
```

```
# Define the columns to be filled
#cols_to_fill = ['Age', 'Height', 'Weight']

# Calculate the mean for each column grouped by 'Sex' and 'Sport'
#mean_values = df.groupby(['Sex', 'Sport'])[cols_to_fill].transform('mean')

# Fill NA values in df with the calculated mean values
#df[cols_to_fill] = df[cols_to_fill].fillna(mean_values)
```

SQL:

Creating Delta Lake

- Create medallion architecture (bronze, silver, gold) with Delta Lake (http://delta.io/)



The files were uploaded using Databricks' "Create Table" feature.

```
%sql
USE default;
```

OK

```
%fs head  /FileStore/tables/athlete_events.csv
```

```
[Truncated to first 65536 bytes]
"ID","Name","Sex","Age","Height","Weight","Team","NOC","Games","Year","Season","City","Sport","Event","Medal"
"1","A Dijiang","M",24,180,80,"China","CHN","1992 Summer",1992,"Summer","Barcelona","Basketball","Basketball Men's Basketbal
l",NA
"2","A Lamusi","M",23,170,60,"China","CHN","2012 Summer",2012,"Summer","London","Judo","Judo Men's Extra-Lightweight",NA
"3","Gunnar Nielsen Aaby","M",24,NA,NA,"Denmark","DEN","1920 Summer",1920,"Summer","Antwerpen","Football","Football Men's Foo
tball",NA
```

```
"4","Edgar Lindenau Aabye","M",34,NA,NA,"Denmark/Sweden","DEN","1900 Summer",1900,"Summer","Paris","Tug-Of-War","Tug-Of-War M
en's Tug-Of-War","Gold"
"5","Christine Jacoba Aaftink","F",21,185,82,"Netherlands","NED","1988 Winter",1988,"Winter","Calgary","Speed Skating","Speed
Skating Women's 500 metres",NA
"5","Christine Jacoba Aaftink","F",21,185,82,"Netherlands","NED","1988 Winter",1988,"Winter","Calgary","Speed Skating","Speed
Skating Women's 1,000 metres",NA
"5","Christine Jacoba Aaftink","F",25,185,82,"Netherlands","NED","1992 Winter",1992,"Winter","Albertville","Speed Skating","S
peed Skating Women's 500 metres",NA
"5","Christine Jacoba Aaftink","F",25,185,82,"Netherlands","NED","1992 Winter",1992,"Winter","Albertville","Speed Skating","S
peed Skating Women's 1,000 metres",NA
"5","Christine Jacoba Aaftink","F",27,185,82,"Netherlands","NED","1994 Winter",1994,"Winter","Lillehammer","Speed Skating","S
peed Skating Women's 500 metres",NA
"5","Christine Jacoba Aaftink","F",27,185,82,"Netherlands","NED","1994 Winter",1994,"Winter","Lillehammer","Speed Skating","S
```

```
%fs ls  /FileStore/tables/athlete_events.csv
```

Table

	path	name	size	modificationTime
1	dbfs:/FileStore/tables/athlete_events.csv	athlete_events.csv	41500688	1706430500000

1 row

```
%fs ls  /FileStore/tables/noc_regions.csv
```

Table

	path	name	size	modificationTime
1	dbfs:/FileStore/tables/noc_regions.csv	noc_regions.csv	3595	1706430585000

1 row

```
athlete_events_df = spark.sql("SELECT * FROM athlete_events_csv")
athlete_events_df.createOrReplaceTempView('athlete_events')
```

```
noc_regions_df = spark.sql("SELECT * FROM noc_regions_csv")
noc_regions_df.createOrReplaceTempView('noc_regions')
```

```
%sql
SELECT * FROM athlete_events LIMIT 2
```

Table

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Games
1	1	A Dijiang	M	24	180	80	China	CHN	1992 Summ
2	2	A Lamusi	M	23	170	60	China	CHN	2012 Summ

2 rows

```
%sql
SELECT * FROM noc_regions LIMIT 2
```

Table

	NOC	region	notes
1	AFG	Afghanistan	null
2	AHO	Curacao	Netherlands Antilles

2 rows

Write raw data into Delta Bronze

```
%sql
CREATE DATABASE IF NOT EXISTS Databricks;
USE Databricks;
CREATE OR REPLACE TABLE athlete_events_Bronze
USING DELTA
AS
  SELECT * FROM athlete_events;
```

Query returned no results

```
%sql
USE Databricks;
CREATE OR REPLACE TABLE noc_regions_Bronze
USING DELTA
AS
  SELECT * FROM noc_regions;
```

Query returned no results

Refine bronze tables, write to Delta Silver

Clean and Filter unnecessary columns and nulls.

```
%sql
CREATE OR REPLACE TABLE athlete_events_Silver
USING DELTA
AS
  SELECT
    CAST(ID AS int) AS ID,
    CAST(Name AS string) AS Name,
    CAST(Sex AS string) AS Sex,
    CAST(Age AS int) AS Age,
    CAST(Height AS int) AS Height,
    CAST(Weight AS float) AS Weight,
    CAST(Team AS string) AS Team,
    CAST(NOC AS string) AS NOC,
    CAST(Games AS string) AS Games,
    CAST(Year AS int) AS Year,
    CAST(Season AS string) AS Season,
    CAST(City AS string) AS City,
    CAST(Sport AS string) AS Sport,
    CAST(Event AS string) AS Event,
    CAST(Medal AS string) AS Medal
  FROM athlete_events_Bronze;

SELECT * FROM athlete_events_Silver LIMIT 10;
```

Table

	ID	Name	Sex	Age	Height	Weight	Team	NOC
1	1	A Dijiang	M	24	180	80	China	CHN
2	2	A Lamusi	M	23	170	60	China	CHN
3	3	Gunnar Nielsen Aaby	M	24	null	null	Denmark	DEN
4	4	Edgar Lindenau Aabye	M	34	null	null	Denmark/Sweden	DEN
5	5	Christine Jacoba Aaftink	F	21	185	82	Netherlands	NED
6	5	Christine Jacoba Aaftink	F	21	185	82	Netherlands	NED

7	5	Christine Jacoba Aaftink	F	25	185	82	Netherlands	NED

10 rows

```
%sql
CREATE OR REPLACE TABLE noc_regions_Silver
USING DELTA
AS
  SELECT NOC,
         region AS Region,
         notes AS Note
  FROM noc_regions_Bronze;

SELECT * FROM noc_regions_Silver LIMIT 10;
```

Table

	NOC	Region	Note
1	AFG	Afghanistan	null
2	AHO	Curacao	Netherlands Antilles
3	ALB	Albania	null
4	ALG	Algeria	null
5	AND	Andorra	null
6	ANG	Angola	null
7	ANT	Antigua	Antigua and Barbuda

10 rows

```
%sql DESCRIBE athlete_events_Silver;
```

Table

	col_name	data_type	comment
1	ID	int	null
2	Name	string	null
3	Sex	string	null
4	Age	int	null
5	Height	int	null
6	Weight	float	null
7	Team	string	null
8	NOC	string	null
9	Games	string	null
10	Year	int	null
11	Season	string	null
12	City	string	null
13	Sport	string	null
14	Event	string	null
15	Medal	string	null

15 rows

```
%sql DESCRIBE noc_regions_Silver;
```

Table

	col_name	data_type	comment
1	NOC	string	null
2	Region	string	null
3	Note	string	null

3 rows

Upon initial examination of the DataFrame, it appears that the ‘Game’ column is a concatenation of the ‘Year’ and ‘Season’ columns. If this is indeed the case, the ‘Game’ column will be deemed redundant and subsequently removed. A thorough check will be conducted to confirm this observation.

```
%sql
CREATE OR REPLACE TEMPORARY VIEW temp_Games_concat
AS
SELECT  Year,
        CAST(SUBSTRING_INDEX(Games, ' ', 1) AS int) AS Games_Year,
        SUBSTRING_INDEX(Games, ' ', -1) AS Games_Season,
        Season
FROM athlete_events_Silver;

SELECT * FROM temp_Games_concat LIMIT 10;
```

Table

	Year	Games_Year	Games_Season	Season
1	1992	1992	Summer	Summer
2	2012	2012	Summer	Summer
3	1920	1920	Summer	Summer
4	1900	1900	Summer	Summer
5	1988	1988	Winter	Winter
6	1988	1988	Winter	Winter
7	1992	1992	Winter	Winter

10 rows

An easy way to determine if two columns are identical is to use the <> operator to compare them. If the columns are the same, there will be no result. If they are not the same, there will be an output.

```
%sql
SELECT *
FROM temp_Games_concat
WHERE Year <> Games_Year OR Season <> Games_Season
```

Query returned no results

As confirmed, the 'Game' column is indeed redundant as it is a concatenation of the 'Year' and 'Season' columns. Therefore, it will be appropriately removed to maintain the efficiency and relevance of the dataset.

```
%sql
-- drop the temporary view we created it is useless now
DROP VIEW temp_Games_concat
```

OK

Delta table doesn’t support the DROP COLUMN operation because Column Mapping is not enabled. You can enable Column Mapping on your Delta table with mapping mode ‘name’

```
%sql

-- This command alters the properties of the 'athlete_events_Silver' table.
ALTER TABLE athlete_events_Silver SET TBLPROPERTIES (
   -- Enables column mapping with 'name' mode. This allows operations like adding, dropping, and renaming columns.
   'delta.columnMapping.mode' = 'name',

   -- Sets the minimum reader version to '2'. This means that the table can only be read by Delta Lake version 0.2.0 and above.
   'delta.minReaderVersion' = '2',

   -- Sets the minimum writer version to '5'. This means that the table can only be written by Delta Lake version 0.5.0 and above
   'delta.minWriterVersion' = '5'
)
```

OK

```
%sql
-- Delete the 'Games' column
ALTER TABLE athlete_events_Silver DROP COLUMNS (Games)
```

OK

```
%sql
SELECT * FROM athlete_events_Silver
```

Table

	ID	Name	Sex	Age	Height
1	1	A Dijiang	M	24	180
2	2	A Lamusi	M	23	170
3	3	Gunnar Nielsen Aaby	M	24	null
4	4	Edgar Lindenau Aabye	M	34	null
5	5	Christine Jacoba Aaftink	F	21	185
6	5	Christine Jacoba Aaftink	F	21	185
7	5	Christine Jacoba Aaftink	F	25	185

10,000 rows | Truncated data

```
%sql
-- 1) Get the number of rows in the dataframe
SELECT COUNT(*) AS NumberOfRows
FROM athlete_events_Silver;
```

Table

	NumberOfRows
1	271116

1 row

```
%sql
-- 2) Get the number of duplicate rows
SELECT COUNT(*) - COUNT(DISTINCT *) AS NumberOfDuplicates
FROM athlete_events_Silver;
```

Table

	NumberOfDuplicates
1	64964

1 row

There are many duplicate values in the data, but it's unclear whether they are truly duplicates. This is because certain sports have numerous events that occur consecutively. Sports such as 'Art Competitions', 'Cycling', 'Sailing', and 'Equestrianism' are examples of this.

```
%sql
-- missing values for each columns
SELECT
  COUNT(*) - COUNT(ID) as ID,
  COUNT(*) - COUNT(Name) as Name,
  COUNT(*) - COUNT(Sex) as Sex,
  COUNT(*) - COUNT(Age) as Age,
  COUNT(*) - COUNT(Height) as Height,
  COUNT(*) - COUNT(Weight) as Weight,
  COUNT(*) - COUNT(Team) as Team,
  COUNT(*) - COUNT(NOC) as NOC,
  COUNT(*) - COUNT(Year) as Year,
  COUNT(*) - COUNT(Season) as Season,
  COUNT(*) - COUNT(City) as City,
  COUNT(*) - COUNT(Sport) as Sport,
  COUNT(*) - COUNT(Event) as Event,
  COUNT(*) - COUNT(Medal) as Medal
FROM
  athlete_events_silver
```

Table

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Year
1	0	0	0	9474	60171	62875	0	0	0

1 row

```
%sql DESCRIBE HISTORY athlete_events_Silver
```

Table

	version	timestamp	userId	userName	operation	ope
1	15	2024-01-28T08:32:54Z	2932016787551600	golden-tix@hotmail.fr	DROP COLUMNS	▸ {"
2	14	2024-01-28T08:32:48Z	2932016787551600	golden-tix@hotmail.fr	SET TBLPROPERTIES	▸ {"
3	13	2024-01-28T08:32:21Z	2932016787551600	golden-tix@hotmail.fr	CREATE OR REPLACE TABLE AS SELECT	▸ {"
4	12	2024-01-25T08:38:30Z	2932016787551600	golden-tix@hotmail.fr	DROP COLUMNS	▸ {"
5	11	2024-01-25T08:38:26Z	2932016787551600	golden-tix@hotmail.fr	SET TBLPROPERTIES	▸ {"
6	10	2024-01-25T08:38:03Z	2932016787551600	golden-tix@hotmail.fr	CREATE OR REPLACE TABLE AS SELECT	▸ {"
7	9	2024-01-24T13:34:36Z	2932016787551600	golden-tix@hotmail.fr	DROP COLUMNS	▸ {"
8	8	2024-01-24T13:34:32Z	2932016787551600	golden-tix@hotmail.fr	SET TBLPROPERTIES	▸ {"

16 rows

Refine, Enrich and Aggregate Data, write to Delta Gold

```
%sql
CREATE OR REPLACE TABLE athlete_events_Gold
USING DELTA
AS
  SELECT *
  FROM athlete_events_Silver;

SELECT * FROM athlete_events_Gold LIMIT 10;
```

Table

	ID	Name	Sex	Age	Height	Weight	Team	NOC
1	1	A Dijiang	M	24	180	80	China	CHN
2	2	A Lamusi	M	23	170	60	China	CHN
3	3	Gunnar Nielsen Aaby	M	24	null	null	Denmark	DEN
4	4	Edgar Lindenau Aabye	M	34	null	null	Denmark/Sweden	DEN

5	5	Christine Jacoba Aaftink	F	21	185	82	Netherlands	NED
6	5	Christine Jacoba Aaftink	F	21	185	82	Netherlands	NED
7	5	Christine Jacoba Aaftink	F	25	185	82	Netherlands	NED

10 rows

```
%sql
CREATE OR REPLACE TABLE noc_regions_Gold
USING DELTA
AS
  SELECT *
  FROM noc_regions_Silver;

SELECT * FROM noc_regions_Gold LIMIT 10;
```

Table

	NOC	Region	Note
1	AFG	Afghanistan	null
2	AHO	Curacao	Netherlands Antilles
3	ALB	Albania	null
4	ALG	Algeria	null
5	AND	Andorra	null
6	ANG	Angola	null
7	ANT	Antigua	Antigua and Barbuda

10 rows

3) Perform initial exploration of data and provide some screenshots or display some stats of the data you are looking at.

- Note: We will switch from using SQL Delta tables to utilizing the Python package `pandas_profiling.ProfileReport` to gain a comprehensive understanding of the dataframe.

brief explanation of the package : The ydata-profiling package is a powerful tool that provides a quick and detailed exploratory data analysis (EDA) with just a single line of code. It's designed to help data scientists understand their dataset better by providing insights such as:

- Type Inference: Automatically detects the types of columns in your DataFrame.
- Descriptive Statistics: Provides summary statistics for each column in your DataFrame.
- Correlations: Identifies relationships between different variables in your DataFrame.
- Missing Values: Highlights columns with missing values and their impact on other columns.
- Warnings: Alerts you to potential issues with your data, such as high cardinality.

Python:

join the two Tables and reorder the columns.

```
df1 = sql(""" SELECT at.ID, at.Name, at.Sex, at.Age, at.Height, at.Weight, at.Team, at.NOC, nr.Region, nr.Note, at.Year, at.Seaso
FROM athlete_events_Gold at
LEFT JOIN noc_regions_Gold nr ON at.NOC = nr.NOC""").toPandas()
df = df1.replace('NA', None)
```

```
df.head()
```

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Region	Note	Year	Season	City	Sport	Event	Medal
0	1	A Dijiang	M	24.0	180.0	80.0	China	CHN	China	None	1992	Summer	Barcelona	Basketball	Basketball Men's Basketball	None
1	2	A Lamusi	M	23.0	170.0	60.0	China	CHN	China	None	2012	Summer	London	Judo	Judo Men's Extra-Lightweight	None
2	3	Gunnar Nielsen	M	24.0	NaN	NaN	Denmark	DEN	Denmark	None	1920	Summer	Antwerpen	Football	Football Men's	None

```
from ydata_profiling import ProfileReport
#ProfileReport
ProfileReport(df)
```

```
Summarize dataset:   0%|          | 0/5 [00:00<?, ?it/s]
Generate report structure:   0%|          | 0/1 [00:00<?, ?it/s]
Render HTML:   0%|          | 0/1 [00:00<?, ?it/s]
```

Overview

Dataset statistics

Number of variables	16
Number of observations	271116
Missing cells	630300
Missing cells (%)	14.5%
Duplicate rows	612
Duplicate rows (%)	0.2%
Total size in memory	30.0 MiB
Average record size in memory	116.0 B

Variable types

Numeric	5
Text	6
Categorical	5

Alerts

Dataset has 612 (0.2%) duplicate rows	Duplicates
`Height` is highly overall correlated with `Weight`	High correlation
`Weight` is highly overall correlated with `Height` and 1 other fields (Height, Sex)	High correlation
`Year` is highly overall correlated with `City`	High correlation
`Sex` is highly overall correlated with `Weight`	High correlation
`Season` is highly overall correlated with `City`	High correlation
`City` is highly overall correlated with `Year` and 1 other fields (Year, Season)	High correlation

Descriptive Statistics of the DataFrame

```
df.describe(include="all")
```

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Region	Note	Year	Seasor
count	271116.000000	271116	271116	261642.000000	210945.000000	208241.000000	271116	271116	270746	5039	271116.000000	271116
unique	NaN	134732	2	NaN	NaN	NaN	1184	230	205	21	NaN	2
top	NaN	Robert Tait McKenzie	M	NaN	NaN	NaN	United States	USA	USA	Yugoslavia	NaN	Summe
freq	NaN	58	196594	NaN	NaN	NaN	17847	18853	18853	2583	NaN	22255
mean	68248.954396	NaN	NaN	25.556898	175.338970	70.702408	NaN	NaN	NaN	NaN	1978.378480	NaN
std	39022.286345	NaN	NaN	6.393561	10.518462	14.348020	NaN	NaN	NaN	NaN	29.877632	NaN
min	1.000000	NaN	NaN	10.000000	127.000000	25.000000	NaN	NaN	NaN	NaN	1896.000000	NaN
25%	34643.000000	NaN	NaN	21.000000	168.000000	60.000000	NaN	NaN	NaN	NaN	1960.000000	NaN
50%	68205.000000	NaN	NaN	24.000000	175.000000	70.000000	NaN	NaN	NaN	NaN	1988.000000	NaN
75%	102097.250000	NaN	NaN	28.000000	183.000000	79.000000	NaN	NaN	NaN	NaN	2002.000000	NaN
max	135571.000000	NaN	NaN	97.000000	226.000000	214.000000	NaN	NaN	NaN	NaN	2016.000000	NaN

```
df['Medal'].value_counts()
```

```
Gold      13372
Bronze    13295
Silver    13116
Name: Medal, dtype: int64
```

```
df.head()
```

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Region	Note	Year	Season	City	Sport	Event	Medal
0	1	A Dijiang	M	24.0	180.0	80.0	China	CHN	China	None	1992	Summer	Barcelona	Basketball	Basketball Men's Basketball	None
1	2	A Lamusi	M	23.0	170.0	60.0	China	CHN	China	None	2012	Summer	London	Judo	Judo Men's Extra-Lightweight	None
2	3	Gunnar Nielsen	M	24.0	NaN	NaN	Denmark	DEN	Denmark	None	1920	Summer	Antwerpen	Football	Football Men's	None

Top 10 Region with the most Medal

```
# Create the pivot table
pivot_table = df.pivot_table(values='ID', index='Region', columns='Medal', aggfunc='count')

# Calculate the total medals for each region
total_medals = pivot_table.sum(axis=1).sort_values(ascending=False)

# Reorder the rows of the pivot table based on total_medals
pivot_table = pivot_table.reindex(total_medals.index)

# top 10 of the region with the most medal
pivot_table.head(10)
```

Medal	Bronze	Gold	Silver
Region			
USA	1358.0	2638.0	1641.0
Russia	1178.0	1599.0	1170.0
Germany	1260.0	1301.0	1195.0
UK	651.0	678.0	739.0
France	666.0	501.0	610.0
Italy	531.0	575.0	531.0
Sweden	535.0	479.0	522.0
Canada	451.0	463.0	438.0
Australia	522.0	368.0	459.0
Hungary	371.0	432.0	332.0

Based on the data, it appears that the United States leads in terms of medal count, followed closely by Russia and Germany.

Top 10 Most Medaled Athletes

```
# Create the pivot table
pivot_table = df.pivot_table(values='ID', index='Name', columns='Medal', aggfunc='count')

# Calculate the total medals for each region
total_medals = pivot_table.sum(axis=1).sort_values(ascending=False)

# Reorder the rows of the pivot table based on total_medals
pivot_table = pivot_table.reindex(total_medals.index)

# top 10 of the region with the most medal
pivot_table.head(20)
```

Medal Name	Bronze	Gold	Silver
Michael Fred Phelps, II	2.0	23.0	3.0
Larysa Semenivna Latynina (Diriy-)	4.0	9.0	5.0
Nikolay Yefimovich Andrianov	3.0	7.0	5.0
Borys Anfiyanovych Shakhlin	2.0	7.0	4.0
Takashi Ono	4.0	5.0	4.0
Ole Einar Bjrndalen	1.0	8.0	4.0
Edoardo Mangiarotti	2.0	6.0	5.0
Ryan Steven Lochte	3.0	6.0	3.0
Birgit Fischer-Schmidt	NaN	8.0	4.0
Paavo Johannes Nurmi	NaN	9.0	3.0
Natalie Anne Coughlin (-Hall)	5.0	3.0	4.0
Jennifer Elisabeth "Jenny" Thompson (-Cumpelik)	1.0	8.0	3.0
Sawao Kato	1.0	8.0	3.0
Dara Grace Torres (-Hoffman, -Minas)	4.0	4.0	4.0
Aleksey Yuryevich Nemov	6.0	4.0	2.0
Aleksandr Vladimirovich Popov	NaN	5.0	6.0
Matthew Nicholas "Matt" Biondi	1.0	8.0	2.0
Mark Andrew Spitz	1.0	9.0	1.0
Viktor Ivanovych Chukarin	1.0	7.0	3.0
Vra slavsk (-Odloilov)	NaN	7.0	4.0

Upon conducting research on Michael Phelps, it is evident that he stands as the most successful and decorated Olympian of all time, boasting a total of 28 medals. Phelps holds the all-time records for Olympic gold medals, Olympic gold medals in individual events, and Olympic medals in individual events.

During the 2004 Summer Olympics in Athens, Phelps matched the record of eight medals of any color at a single Games, previously held by gymnast Alexander Dityatin, by securing six gold and two bronze medals.

In a remarkable feat four years later, Phelps won eight gold medals at the 2008 Beijing Games, surpassing fellow American swimmer Mark Spitz's 1972 record of seven first-place finishes at any single Olympic Games.

Utilizing the data at our disposal, we will verify the information we have discovered

```
# Create the pivot table
pivot_table = df.pivot_table(values='ID', index=['Name', 'Year'], columns='Medal', aggfunc='count')

# Calculate the total medals for each athlete and year
total_medals = pivot_table.sum(axis=1).sort_values(ascending=False)

# Reorder the rows of the pivot table based on total_medals
pivot_table = pivot_table.reindex(total_medals.index)

# Display the top 20 athletes and years with the most medals
pivot_table.head(20)
```

	Medal	Bronze	Gold	Silver
Name	Year			
Aleksandr Nikolayevich Dityatin	1980	1.0	3.0	4.0
Michael Fred Phelps, II	2008	NaN	8.0	NaN
	2004	2.0	6.0	NaN
Lloyd Spencer Spooner	1920	2.0	4.0	1.0
Mark Andrew Spitz	1972	NaN	7.0	NaN
Mariya Kindrativna Horokhovska	1952	NaN	2.0	5.0
Nikolay Yefimovich Andrianov	1976	1.0	4.0	2.0
Borys Anfiyanovych Shakhlin	1960	1.0	4.0	2.0
Willis Augustus Lee, Jr.	1920	1.0	5.0	1.0
Mikhail Yakovlevich Voronin	1968	1.0	2.0	4.0
Matthew Nicholas "Matt" Biondi	1988	1.0	5.0	1.0
Viljo Eino "Ville" Ritola (Koukkari-)	1924	NaN	4.0	2.0
George Louis Eyser	1904	1.0	3.0	2.0
Viktor Ivanovych Chukarin	1952	NaN	4.0	2.0
Larysa Semenivna Latynina (Diriy-)	1964	2.0	2.0	2.0
Carl Townsend Osburn	1920	1.0	4.0	1.0
Vra slavsk (-Odloilov)	1968	NaN	4.0	2.0
Viorica Daniela Siliva (-Harper)	1988	1.0	3.0	2.0
Hermann Otto Ludwig Weingrtner	1896	1.0	3.0	2.0
Michael Fred Phelps, II	2016	NaN	5.0	1.0

```
# Filter the dataframe for only 'Gold' medals
df_gold = df[df['Medal'] == 'Gold']

# Create the pivot table
pivot_table_gold = df_gold.pivot_table(values='ID', index=['Name', 'Year'], aggfunc='count')

# Calculate the total gold medals for each athlete and year
total_gold_medals = pivot_table_gold.sum(axis=1).sort_values(ascending=False)

# Reorder the rows of the pivot table based on total_gold_medals
pivot_table_gold = pivot_table_gold.reindex(total_gold_medals.index)

# Display the top 20 athletes and years with the most gold medals
pivot_table_gold = pivot_table_gold.rename(columns={'ID': 'Gold'})
pivot_table_gold.head(20)
```

		Gold
Name	Year	
Michael Fred Phelps, II	2008	8
Mark Andrew Spitz	1972	7
Michael Fred Phelps, II	2004	6
Kristin Otto	1988	6
Vitaly Venediktovich Shcherbo	1992	6
Matthew Nicholas "Matt" Biondi	1988	5
Anton Heida	1904	5
Willis Augustus Lee, Jr.	1920	5
Eric Arthur Heiden	1980	5
Paavo Johannes Nurmi	1924	5
Nedo Nadi	1920	5
Michael Fred Phelps, II	2016	5
Larysa Semenivna Latynina (Diriy-)	1956	4
Lloyd Spencer Spooner	1920	4
Vladimir Nikolayevich Artyomov	1988	4
gnes Keleti-Srkny (Klein)	1956	4
Viktor Ivanovych Chukarin	1952	4
Simone Arianne Biles	2016	4
Carl Schuhmann	1896	4
Donald Arthur "Don" Schollander	1964	4

The information seems to be correct. Michael Phelps came close to matching the record set by Mark Andrew Spitz in 1972 of seven gold medals during the 2004 Olympics, having secured six gold medals himself. However, in the 2008 Olympics, Phelps exceeded this record and established himself as the most decorated athlete in Olympic history with eight gold medals. Furthermore, with a total of 28 medals, he is the most decorated Olympian of all time.

4) Create an ERD or proposed ERD to show the relationships of the data you are exploring.

Python

Split the columns to create my ERD

```
df.head()
```

	ID	Name	Sex	Age	Height	Weight	Team	NOC	Region	Note	Year	Season	City	Sport	Event	Medal
0	1	A Dijiang	M	24.0	180.0	80.0	China	CHN	China	None	1992	Summer	Barcelona	Basketball	Basketball Men's Basketball	None
1	2	A Lamusi	M	23.0	170.0	60.0	China	CHN	China	None	2012	Summer	London	Judo	Judo Men's Extra-Lightweight	None
2	3	Gunnar Nielsen	M	24.0	NaN	NaN	Denmark	DEN	Denmark	None	1920	Summer	Antwerpen	Football	Football Men's	None

```
# Fact Table: Competitions
df_competitions = df1[['Year', 'Season', 'City', 'Sport', 'Event', 'Medal']].copy()
df_competitions.insert(0, 'Competition_ID', range(1, 1 + len(df_competitions)))

# Dimension Tables
df_athletes = df1[['ID', 'Name', 'Sex', 'Age', 'Height', 'Weight']]
df_teams = df1[['ID', 'Team', 'NOC', 'Region', 'Note']]

# Add Athlete_ID to Competitions table
df_competitions['Athlete_ID'] = df1['ID']
```

Save the result as csv

```
# Save the Competitions dataframe as a CSV file
display(df_competitions)
```

Table

	Competition_ID	Year	Season	City	Sport	Event
1	1	1992	Summer	Barcelona	Basketball	Basketball Men's Basketball
2	2	2012	Summer	London	Judo	Judo Men's Extra-Lightweight
3	3	1920	Summer	Antwerpen	Football	Football Men's Football
4	4	1900	Summer	Paris	Tug-Of-War	Tug-Of-War Men's Tug-Of-Wa
5	5	1988	Winter	Calgary	Speed Skating	Speed Skating Women's 500 r
6	6	1988	Winter	Calgary	Speed Skating	Speed Skating Women's 1,000
7	7	1992	Winter	Albertville	Speed Skating	Speed Skating Women's 500 r

10,000 rows | Truncated data

```
# Save the Athletes dataframe as a CSV file
display(df_athletes)
```

Table

	ID	Name	Sex	Age	Height
1	1	A Dijiang	M	24	180
2	2	A Lamusi	M	23	170
3	3	Gunnar Nielsen Aaby	M	24	null
4	4	Edgar Lindenau Aabye	M	34	null
5	5	Christine Jacoba Aaftink	F	21	185
6	5	Christine Jacoba Aaftink	F	21	185
7	5	Christine Jacoba Aaftink	F	25	185

10,000 rows | Truncated data

```
# Save the Teams dataframe as a CSV file
display(df_teams)
```

Table

	ID	Team	NOC	Region	Note
1	1	China	CHN	China	null
2	2	China	CHN	China	null
3	3	Denmark	DEN	Denmark	null
4	4	Denmark/Sweden	DEN	Denmark	null
5	5	Netherlands	NED	Netherlands	null
6	5	Netherlands	NED	Netherlands	null
7	5	Netherlands	NED	Netherlands	null

10,000 rows | Truncated data

Step 2 : Develop Project Proposal

In this step, you will need to include the following:

1) Description Write a 5-6 sentence paragraph describing your project; include who might be interested to learn about your findings. Who might be your audience?

2) Questions Create 2-3 questions that you want to answer with the data:

- This will be easier to answer once you've had an opportunity to look at the data and do some initial exploration.
- Don't get carried away on the analysis piece at this stage as there will be more analysis later.
- Do focus on key data elements that are present. For instance: What are they, when are they, who are they about? Do they connect? How do they connect? Jot down ideas as you brainstorm.

3) Hypothesis What are your initial hypotheses about the data?

Write 2-3 assumptions about the data that you'll want to go back to prove or disprove. You will want to keep them in front of you as you look at the data to keep them or change them. You may see relationships that you want to explore and will develop a "belief" about the data.

- Start documenting what you think you can tell from the data.
- What pops up as interesting to you? Most likely it will be interesting to others as well.
- Use the discussion boards to discuss with others about your client and the data to brainstorm together.

4) Approach Describe in 5-6 sentences what approach you are going to take in order to prove (or disprove) your hypotheses. Think about the following in your answer:

- What features (fields/columns) are you going to look at first?
- Is there a relationship that exists that you want to explore?
- What metric/ evaluation measure will you use?

1) Description Write a 5-6 sentence paragraph describing your project; include who might be interested to learn about your findings. Who might be your audience?

This project aims to analyze a dataset from SportsStats, a sports analysis firm that partners with local news outlets and elite personal trainers to provide insightful patterns and trends. The dataset contains detailed information about various sports competitions, including athletes' details, teams, and competition results. The findings from this project could be of interest to sports enthusiasts, news outlets looking for unique sports stories, personal trainers seeking health insights, and even sports analysts looking for patterns and trends in sports competitions.

2) Questions Create 2-3 questions that you want to answer with the data:

- Which are the top 10 regions with the most medals?
- Who are the top 10 athletes with the most medals?
- Conduct a detailed analysis on Michael Phelps and his all-time records for the Olympics. Is he really the athlete with the most gold medals?

3) Hypothesis What are your initial hypotheses about the data?

Write 2-3 assumptions about the data that you'll want to go back to prove or disprove. You will want to keep them in front of you as you look at the data to keep them or change them. You may see relationships that you want to explore and will develop a "belief" about the data.

- Regional Dominance: Certain regions might have a higher medal count due to factors such as better training facilities, more funding, or a larger pool of athletes.
- Athlete Excellence: Some athletes might have significantly more medals than others. This could be due to their exceptional skills, physical attributes, or the nature of the sport they participate in.
- Michael Phelps' Record: Michael Phelps might indeed be the athlete with the most gold medals in the history of the Olympics. His exceptional performance could be attributed to factors such as his training regimen, physical attributes, or mental strength.

4) Approach Describe in 5-6 sentences what approach you are going to take in order to prove (or disprove) your hypotheses.

The best approach is to use a pivot table.

- Regional Medal Count: To identify the top 10 regions with the most medals, I will first focus on the 'Region' and 'Medal' columns. I will count the number of medals for each region and then sort the regions based on this count.
- Top Athletes: To find the top 10 athletes with the most medals, I will look at the 'Name' and 'Medal' columns. Similar to the regional medal count, I will count the number of medals for each athlete and then sort the athletes based on this count.
- Michael Phelps Analysis: For a detailed analysis on Michael Phelps, I will create a pivot table similar to the ones used in previous questions. The pivot table will be based on the 'Name', 'Year', and 'Medal' columns. I will filter the data to include only the top 10 athletes based on the number of 'Gold' medals won. This will allow me to analyze and compare the performances of Michael Phelps and other top athletes over the years in terms of gold medals won.

Throughout the analysis, I will use visualizations to better understand the data and to present the findings. Depending on the nature of the relationships discovered, appropriate statistical tests may be applied later to confirm the findings.