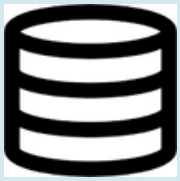
OLYMPIC DATA ANALYTICS



DATA SOURCE

DATA FACTORY

DATA LAKE GEN 2
RAW DATA

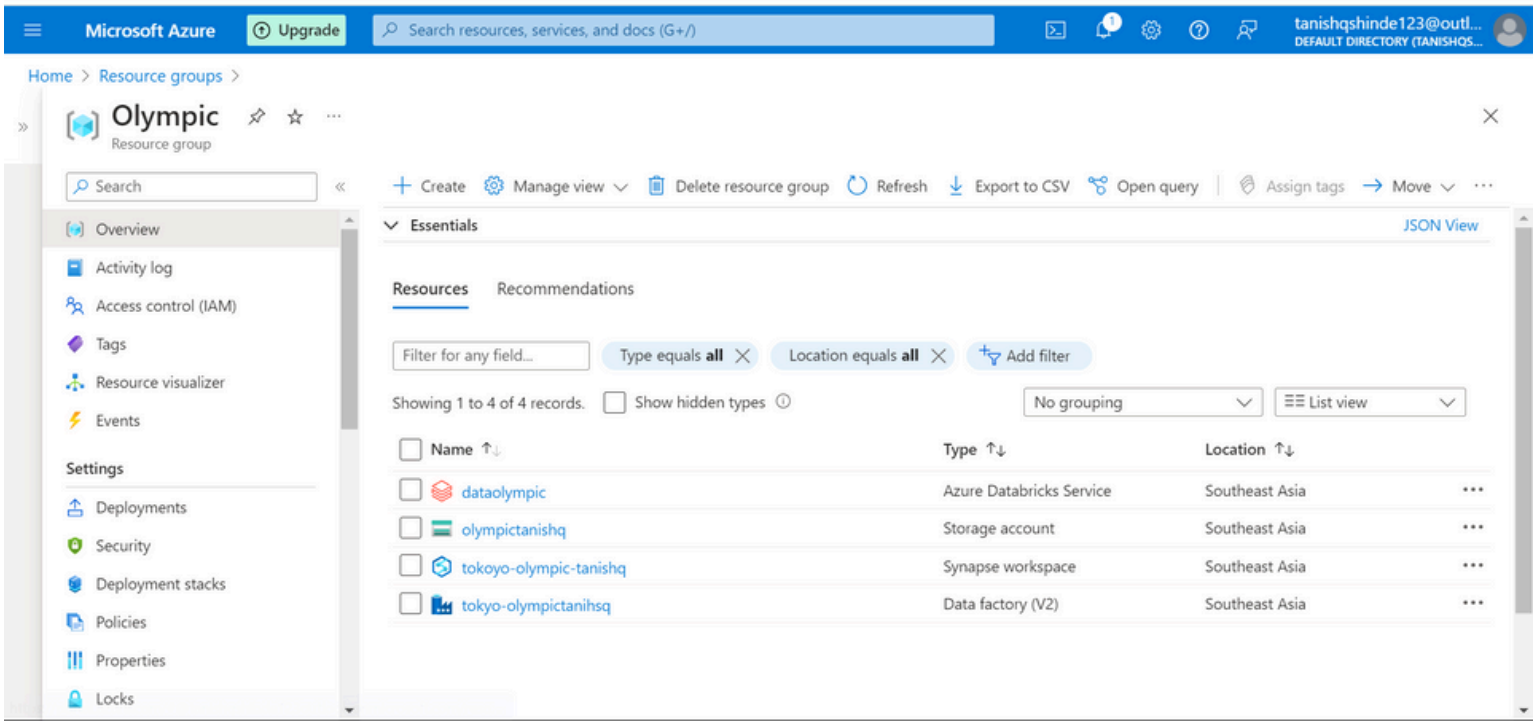
DATABRICKS

DATA LAKE GEN 2
TRANSFORMED
DATA

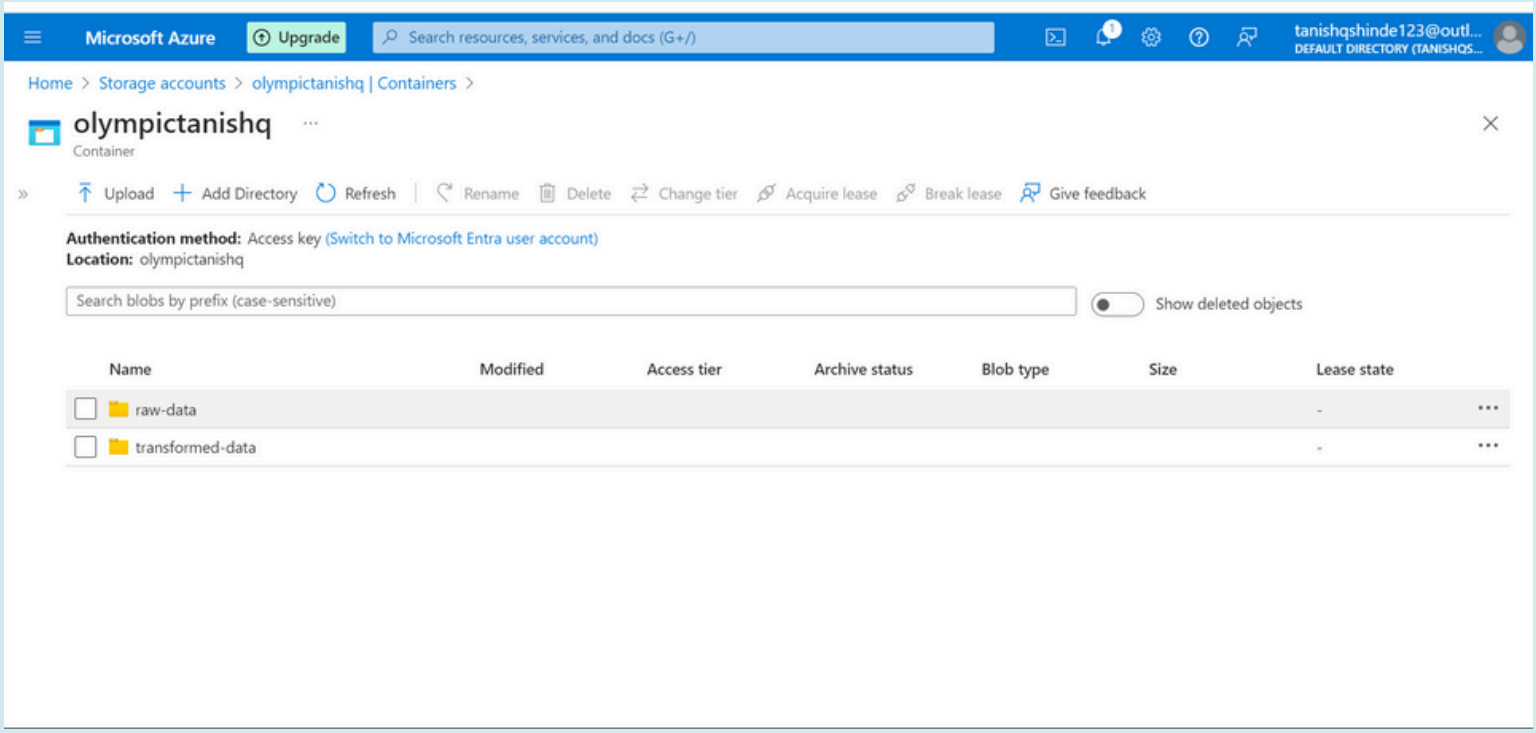
AZURE SYNAPSE
ANALYTICS

RESOURCE GROUPS

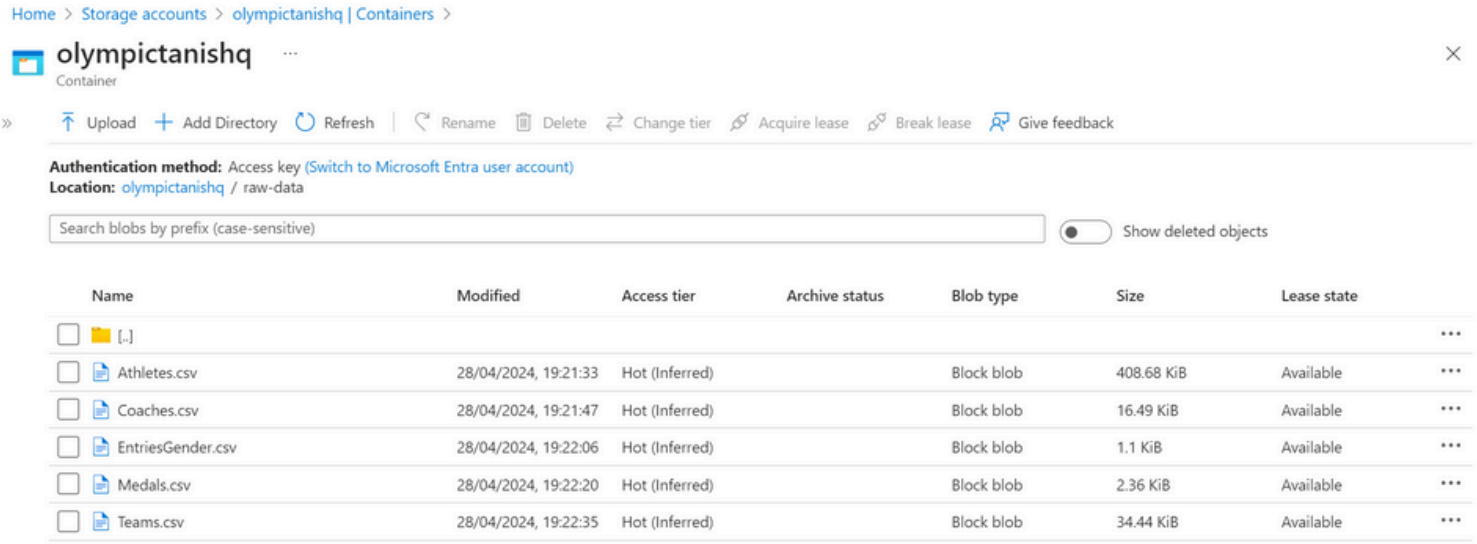
OLYMPIC



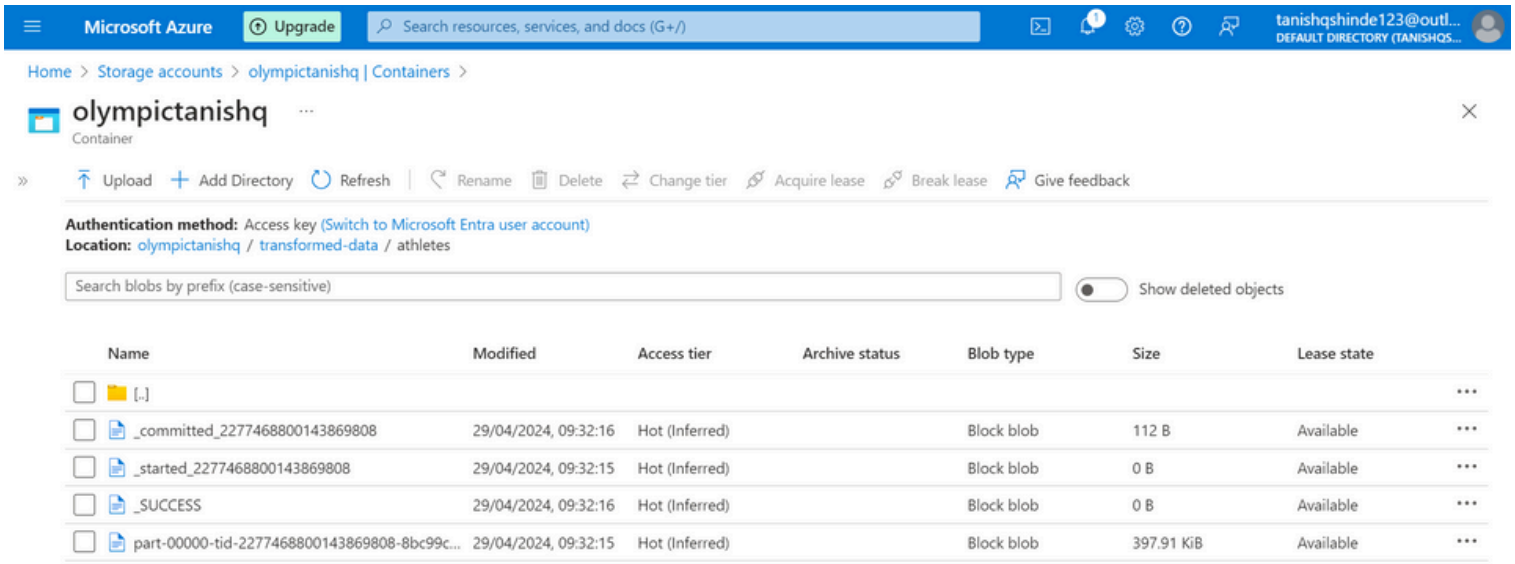
STORAGE ACCOUNTS



RAW DATA



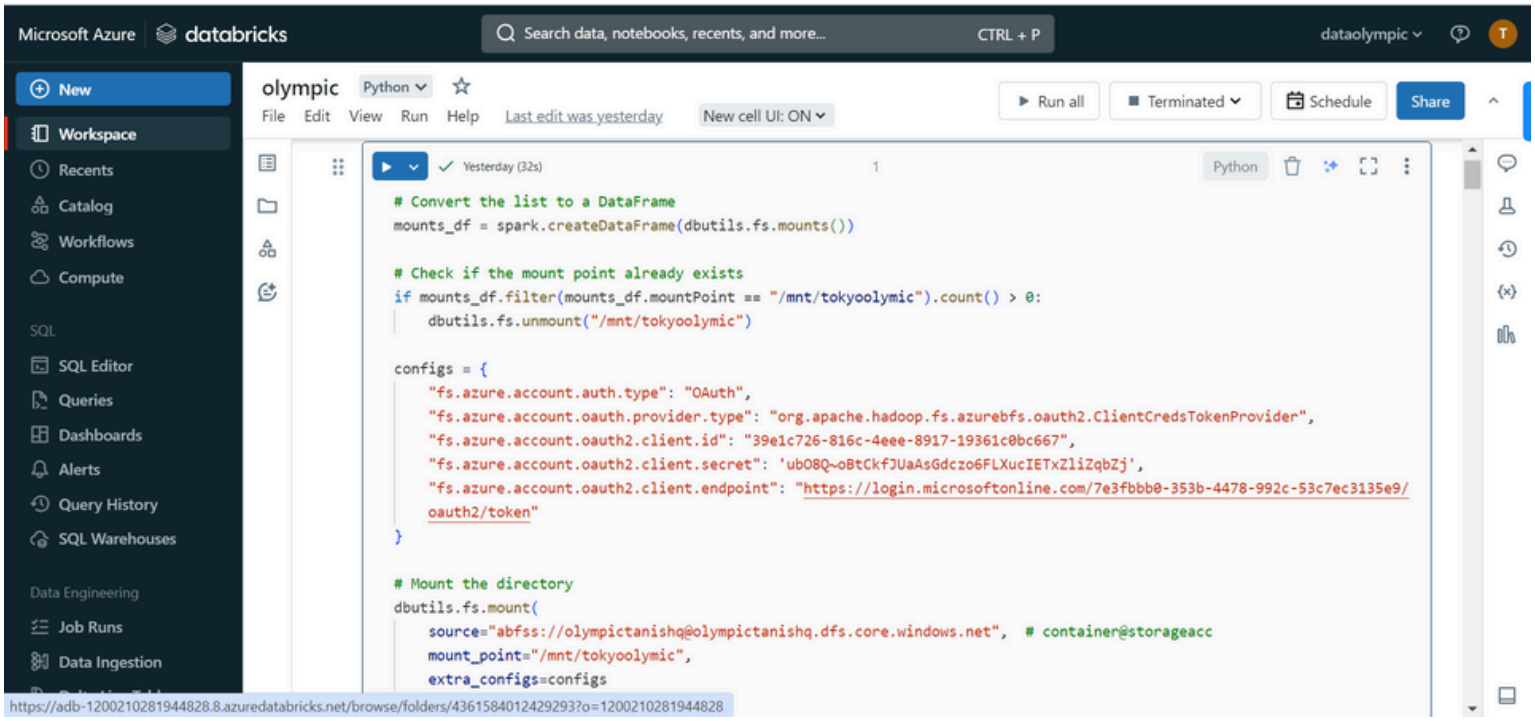
TRANSFORMED-DATA



DATA FACTORIES

DATA PIPELINE

DATABRICKS



```
%fs
ls "/mnt/tokyoolymic"
```

Table + New result table: ON Search

	path	name	size	modificationTime
1	dbfs:/mnt/tokyoolymic/raw-data/	raw-data/	0	1714308047000
2	dbfs:/mnt/tokyoolymic/transformed-dat...	transformed-dat...	0	1714308116000

Yesterday (6s) 4 Python

```
athletes = spark.read.format("csv").option("header","true").option("inferSchema","true").load("/mnt/tokyoolymic/raw-data/Athletes.csv")
coaches = spark.read.format("csv").option("header","true").option("inferSchema","true").load("/mnt/tokyoolymic/raw-data/Coaches.csv")
entriesgender = spark.read.format("csv").option("header","true").option("inferSchema","true").load("/mnt/tokyoolymic/raw-data/EntriesGender.csv")
medals = spark.read.format("csv").option("header","true").option("inferSchema","true").load("/mnt/tokyoolymic/raw-data/Medals.csv")
teams = spark.read.format("csv").option("header","true").option("inferSchema","true").load("/mnt/tokyoolymic/raw-data/Teams.csv")
```

(10) Spark Jobs

- athletes: pyspark.sql.dataframe.DataFrame = [PersonName: string, Country: string ... 1 more field]
- coaches: pyspark.sql.dataframe.DataFrame = [Name: string, Country: string ... 2 more fields]
- entriesgender: pyspark.sql.dataframe.DataFrame = [Discipline: string, Female: integer ... 2 more fields]
- medals: pyspark.sql.dataframe.DataFrame = [Rank: integer, Team_Country: string ... 5 more fields]
- teams: pyspark.sql.dataframe.DataFrame = [TeamName: string, Discipline: string ... 2 more fields]

Yesterday (1s) 5 Python

```
athletes.show()
```

(1) Spark Jobs

```
+--------------------+--------------------+--------------------+
|          PersonName|             Country|          Discipline|
+--------------------+--------------------+--------------------+
|      AALERUD Katrine|              Norway|       Cycling Road|
|         ABAD Nestor|               Spain|Artistic Gymnastics|
|    ABAGNALE Giovanni|               Italy|              Rowing|
|       ABALDE Alberto|               Spain|          Basketball|
|        ABALDE Tamara|               Spain|          Basketball|
|           ABALO Luc|              France|            Handball|
|         ABAROA Cesar|               Chile|              Rowing|
```

Yesterday (<1s) 6

```
athletes.printSchema()
```

```
root
 |-- PersonName: string (nullable = true)
 |-- Country: string (nullable = true)
 |-- Discipline: string (nullable = true)
```

Yesterday (1s) 7 Python

```
top_gold_medal_countries = medals.orderBy("Gold", ascending=False).select("Team_Country","Gold").show()
```

(1) Spark Jobs

```
+--------------------+----+
|        Team_Country|Gold|
+--------------------+----+
|United States of ...|  39|
|People's Republic...|  38|
|               Japan|  27|
|       Great Britain|  22|
|                ROC|  20|
|           Australia|  17|
|         Netherlands|  10|
|              France|  10|
|             Germany|  10|
|               Italy|  10|
```

Yesterday (1s) 8 Python

```
# Calculate the average number of entries by gender for each discipline
average_entries_by_gender = entriesgender.withColumn(
    'Avg_Female', entriesgender['Female'] / entriesgender['Total']
).withColumn(
    'Avg_Male', entriesgender['Male'] / entriesgender['Total']
)
average_entries_by_gender.show()
```

(1) Spark Jobs

average_entries_by_gender: pyspark.sql.dataframe.DataFrame = [Discipline: string, Female: integer ... 4 more fields]

```
+--------------------+------+----+-----+------------------+------------------+
|          Discipline|Female|Male|Total|        Avg_Female|          Avg_Male|
+--------------------+------+----+-----+------------------+------------------+
|      3x3 Basketball|    32|  32|   64|               0.5|               0.5|
|             Archery|    64|  64|  128|               0.5|               0.5|
| Artistic Gymnastics|    98|  98|  196|               0.5|               0.5|
|   Artistic Swimming|   105|   0|  105|               1.0|               0.0|
|           Athletics|   969|1072| 2041| 0.4747672709456149| 0.5252327290543851|
|           Badminton|    86|  87|  173|0.49710982658959535| 0.5028901734104047|
```

TRANSFORMED-DATA

Yesterday (1s) 10 Python

```
# Read the athletes CSV file into a Spark DataFrame
athletes_transformed = spark.read.csv("/mnt/tokyoolymic/transformed-data/athletes", header=True, inferSchema=True)

# Show the first few rows of the DataFrame
athletes_transformed.show()
```

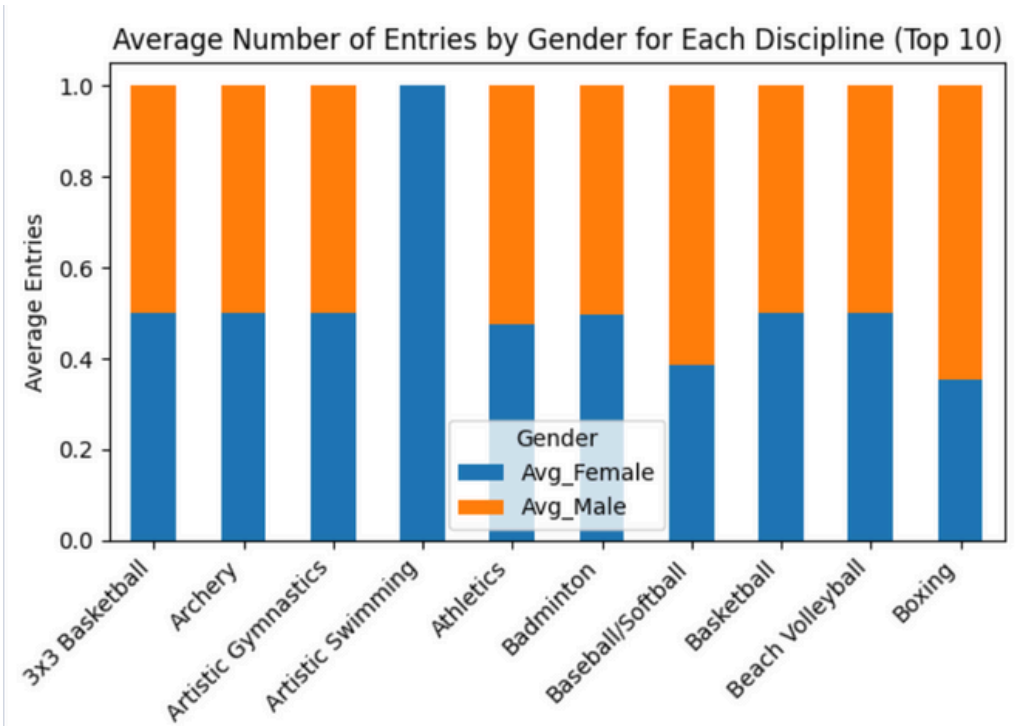
(3) Spark Jobs

athletes_transformed: pyspark.sql.dataframe.DataFrame = [PersonName: string, Country: string ... 1 more field]

PersonName	Country	Discipline
AALERUD Katrine	Norway	Cycling Road
ABAD Nestor	Spain	Artistic Gymnastics
ABAGNALE Giovanni	Italy	Rowing
ABALDE Alberto	Spain	Basketball
ABALDE Tamara	Spain	Basketball
ABALO Luc	France	Handball

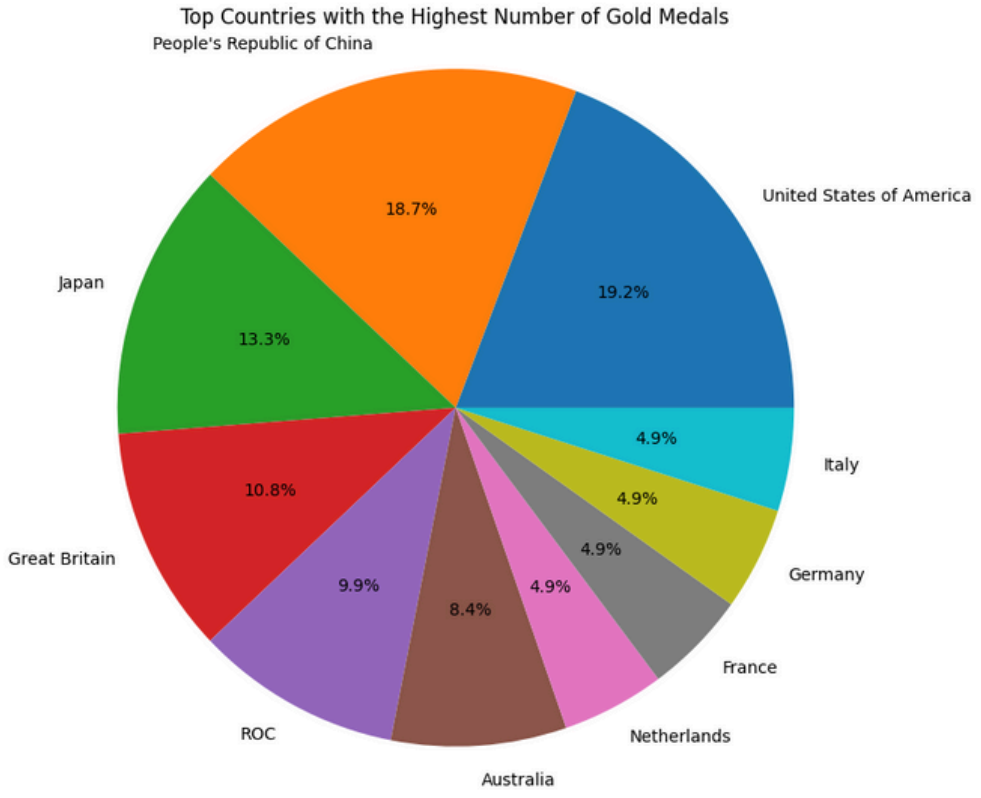
Yesterday (1s) 12

```
# Convert Spark DataFrame to Pandas DataFrame
average_entries_by_gender_pd = average_entries_by_gender.limit(10).toPandas()

# Plot the average number of entries by gender for each discipline
plt.figure(figsize=(12, 8))
average_entries_by_gender_pd.plot(kind='bar', x='Discipline', y=['Avg_Female', 'Avg_Male'], stacked=True)
plt.xlabel('Discipline')
plt.ylabel('Average Entries')
plt.title('Average Number of Entries by Gender for Each Discipline (Top 10)')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Gender')
plt.tight_layout()
plt.show()
```



Yesterday (1s) 13 Python

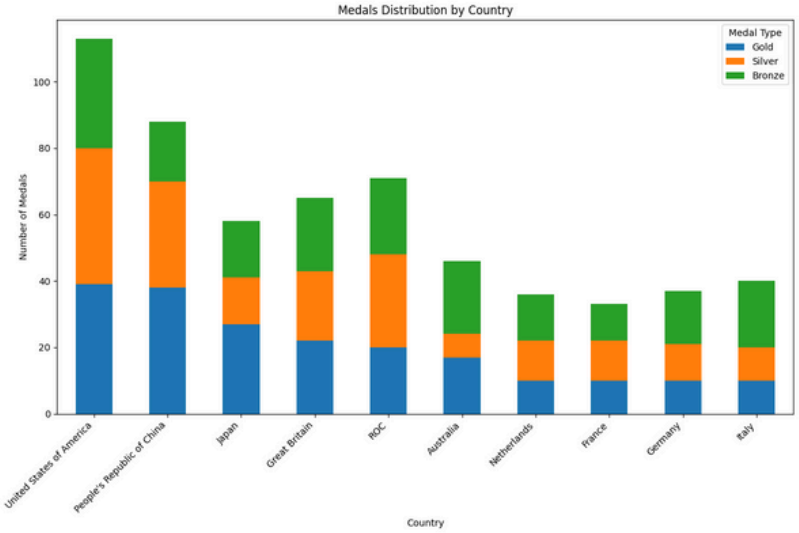
```
# Find the top countries with the highest number of gold medals
top_gold_medal_countries = medals.orderBy("Gold", ascending=False).select("Team_Country","Gold")

# Convert Spark DataFrame to Pandas DataFrame
top_gold_medal_countries_pd = top_gold_medal_countries.limit(10).toPandas()

# Plot the top countries with the highest number of gold medals using a pie chart
plt.figure(figsize=(8, 8))
plt.pie(top_gold_medal_countries_pd['Gold'], labels=top_gold_medal_countries_pd['Team_Country'], autopct='%1.1f%%')
plt.title('Top Countries with the Highest Number of Gold Medals')
plt.axis('equal')
plt.show()
```



Yesterday (1s) 14 Python

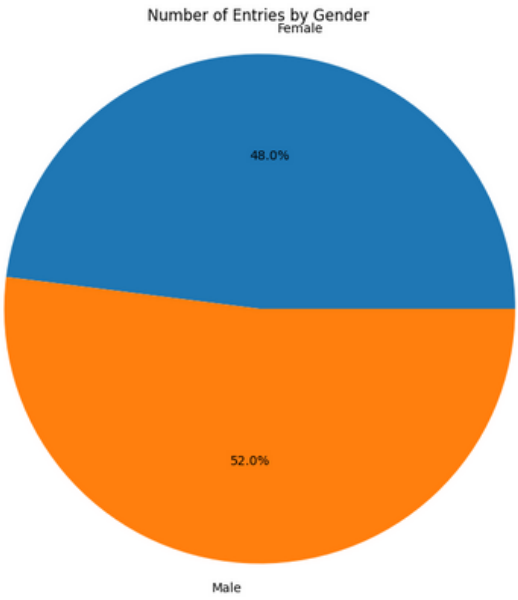
```
# Convert Spark DataFrame to Pandas DataFrame
medals_pd = medals.limit(10).toPandas()

# Plot the distribution of medals by country using a stacked bar chart
medals_pd.set_index('Team_Country')[['Gold', 'Silver', 'Bronze']].plot(kind='bar', stacked=True, figsize=(12, 8))
plt.xlabel('Country')
plt.ylabel('Number of Medals')
plt.title('Medals Distribution by Country')
plt.xticks(rotation=45, ha='right')
plt.legend(title='Medal Type')
plt.tight_layout()
plt.show()
```

(1) Spark Jobs

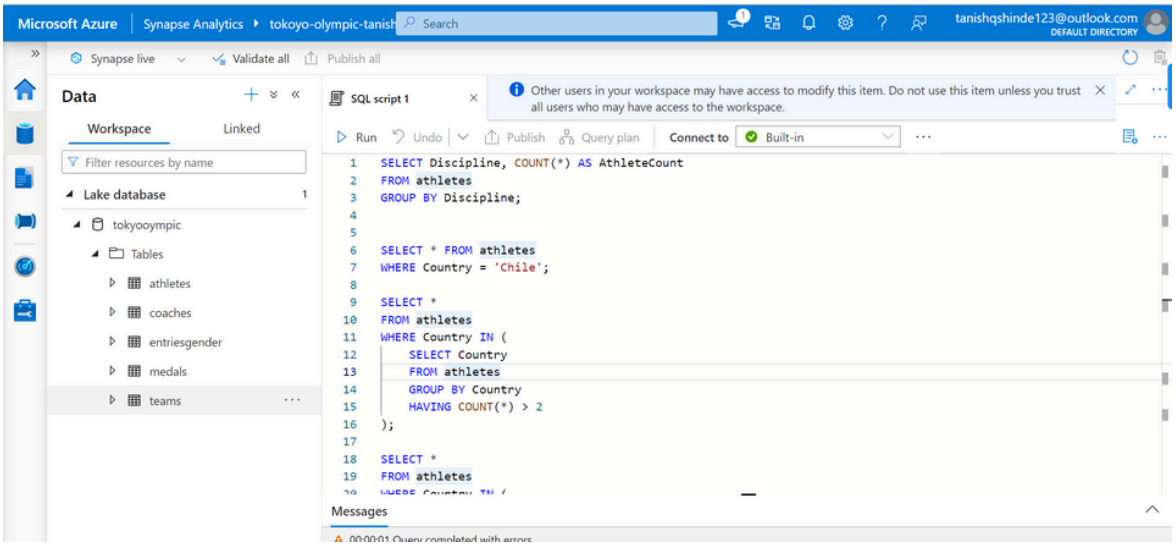


Yesterday (1s) 15 Python

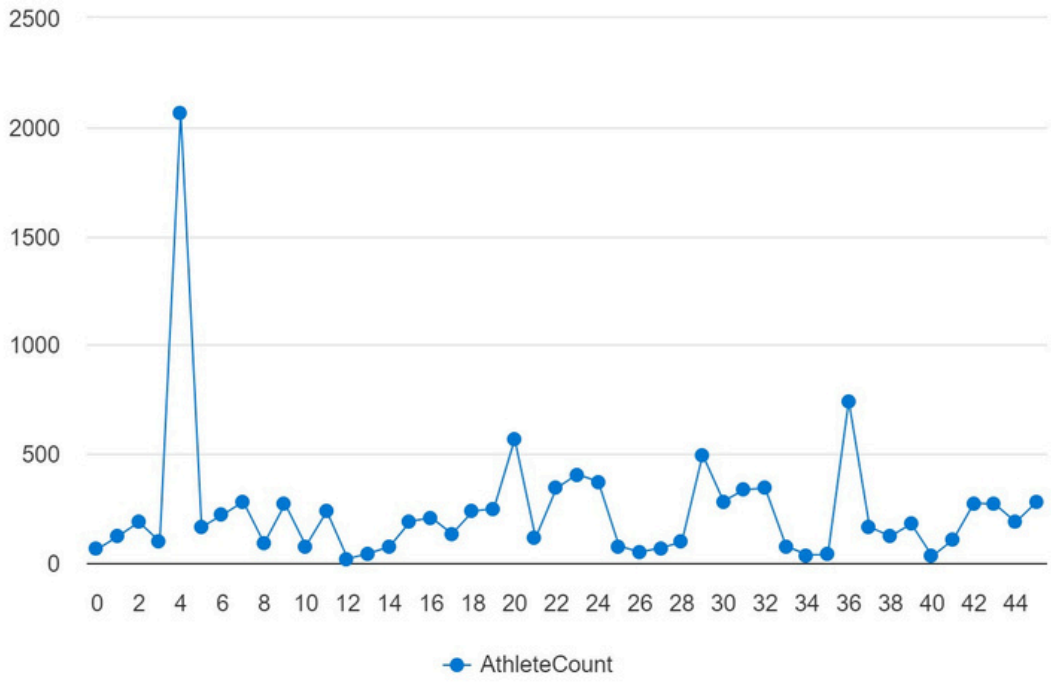
```python
# Calculate total entries by gender
total_female_entries = average_entries_by_gender.selectExpr("sum(Female) as TotalFemale").collect()[0]['TotalFemale']
total_male_entries = average_entries_by_gender.selectExpr("sum(Male) as TotalMale").collect()[0]['TotalMale']

# Plot the number of entries by gender using a pie chart
plt.figure(figsize=(8, 8))
plt.pie([total_female_entries, total_male_entries], labels=['Female', 'Male'], autopct='%1.1f%%')
plt.title('Number of Entries by Gender')
plt.axis('equal')
plt.show()
```



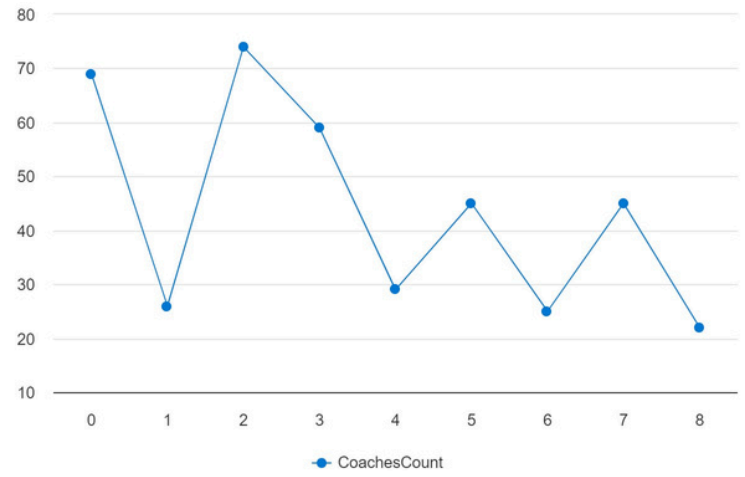
SYNAPSE STUDIO



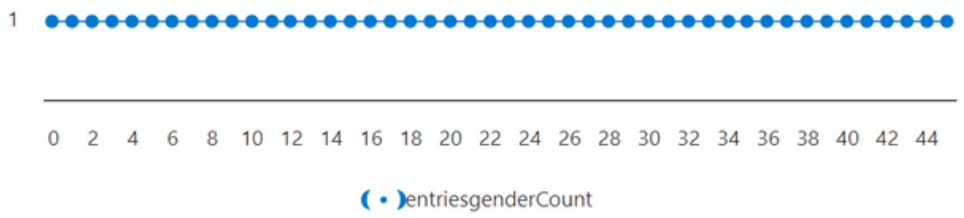
ATHLETES



COACHES



ENTRIESGENDER



TEAMS