
ASTRONOMY
GAIA ASTRONOMICAL DATA

Introduction

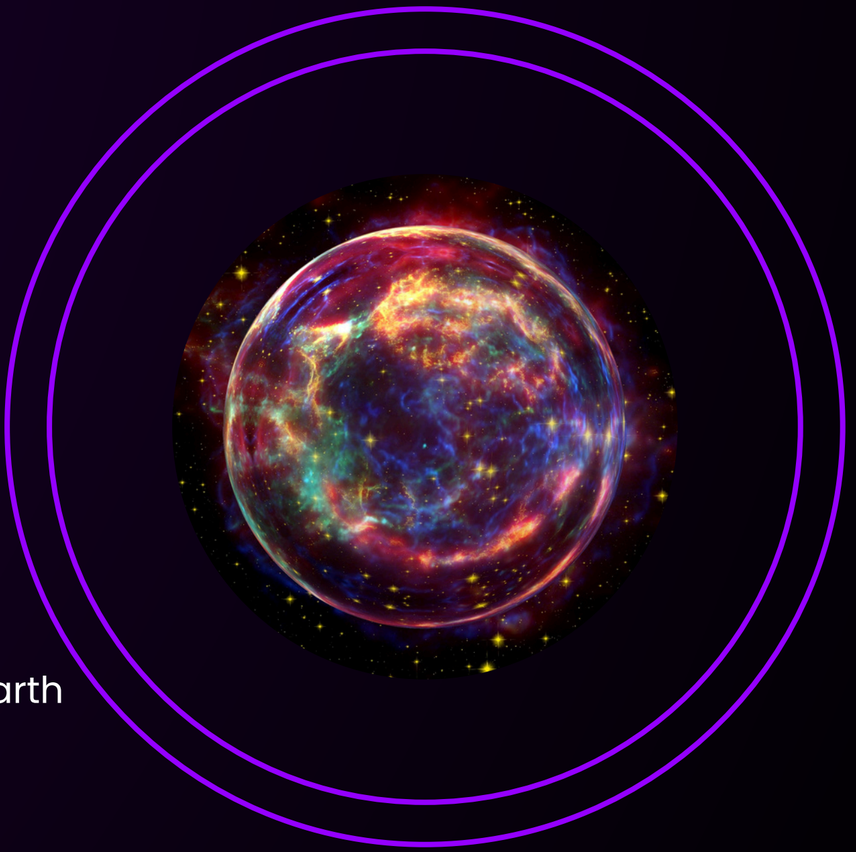
- Database Name:
  - Gaia Data Release 3 (DR3)

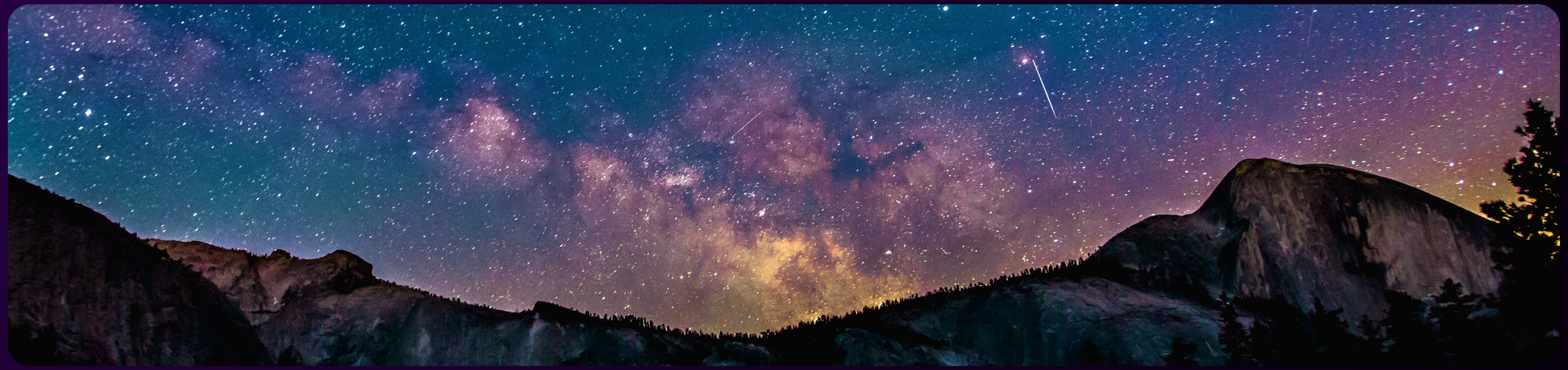
- Source:
  - European Space Agency's Gaia mission

- Purpose:
  - To map and analyze stars in the Milky Way galaxy and beyond

- Scope:
  - Detailed observations of stars and other objects
  - Supports astronomical research and spatial analysis
  - Observations from the Gaia Satellite, launched in 2013 to L2, 932,000 miles from Earth
  - 1.8 billion total objects in database, ie: stars, galaxies, quasars
  - Full dataset is 10 Terabytes

- Key Focus:
  - Understanding star distances, temperatures, and systems

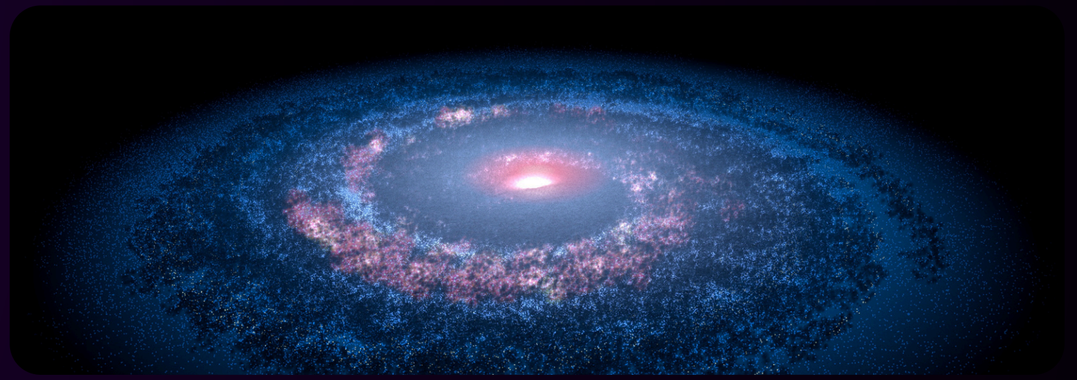




Objective

- Leverage the Gaia database for astronomical analysis.
- Explore celestial phenomena within the Milky Way and beyond.
- Clean and standardize the dataset for accuracy.
- Perform statistical analysis to identify trends and patterns.
- Derive key insights into star classifications and distributions.

Key Dataset Attributes



Designation

Name of the star for identification.

RA (Right Ascension)

Needed for 3D location graphs (horizontal celestial coordinate).

DEC (Declination)

Needed for 3D location graphs (vertical celestial coordinate).

Distance_gspphot

Measures distance from Earth in parsecs (1 parsec = 3.26 light years).

Parallax

Measured in milliarcseconds (mas), used to calculate distances to stars.

Non-Single Star

Indicates binary system (True) or single star (False).

Teff_gspphot

- Measures star temperature in Kelvin.
- Determines star classification:
  - O-class: >30,000 K
    - (Very hot, blue stars).
  - B-class: 10,000–30,000 K
    - (Blue-white stars).
  - A-class: 7,500–10,000 K
    - (White stars).
  - F-class: 6,000–7,500 K
    - (Yellow-white stars).
  - G-class: 5,200–6,000 K
    - (Yellow stars, like the Sun).
  - K-class: 3,700–5,200 K
    - (Orange stars).
  - M-class: <3,700 K
    - (Red stars).

Workflow

Data Query
Use the Gaia DR3 API to retrieve stars

Converting parallax
Convert parallax values to distances in light years.

Data Cleanup
Clean and standardize the data (e.g., handle missing values and remove unnecessary columns).

Conducting Statistical Analysis

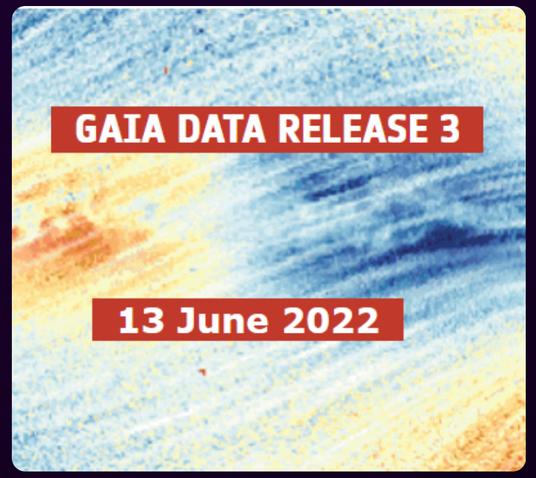
- Calculating temperature ranges.
- Identifying the closest and farthest stars.
- Determining standard deviation for temperature.

Visualization
Generate charts like scatter plots and boxplots to identify trends and outliers.

Data Export
Export to database and as CSV file.

Gaia Database API
Data Release 3 of June 2022

- Currently the largest repository of astronomical objects in existence
- 1.8 billion objects
- 1.6 billion of those objects are in our galaxy
- The most precise positions and distances
- Has an excellent web-based search interface
- Allows users to practice retrieving data before moving to an API



To access API, install:

- astropy: core Python library for astronomy
- astroquery: set of tools to query astronomy databases

Python code executes a query from main table:

- gaiadr3.gaia_source

Search is initiated with the method:

- launch_job_async()

Results are retrieved with the method:

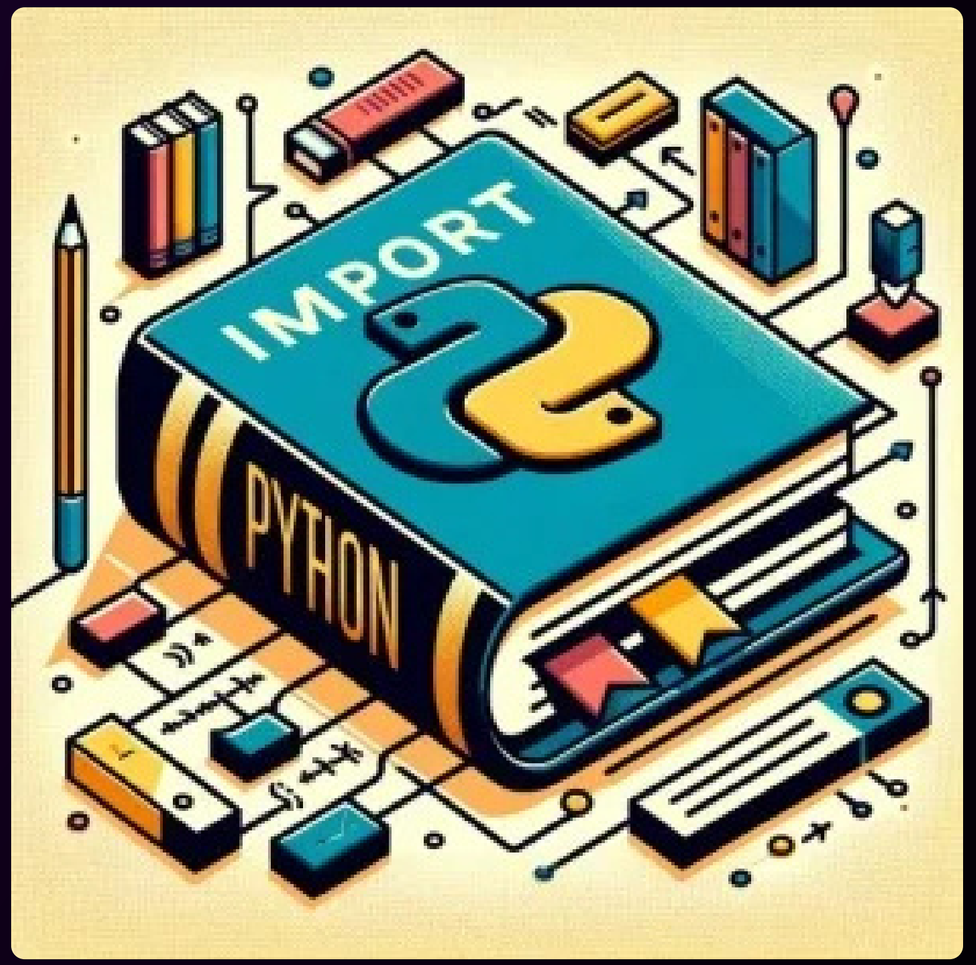
- get_results()

Select query is constructed using a form of SQL called Astronomical Data Query Language (ADQL)

Step 1: Install & Import Libraries

Objective:

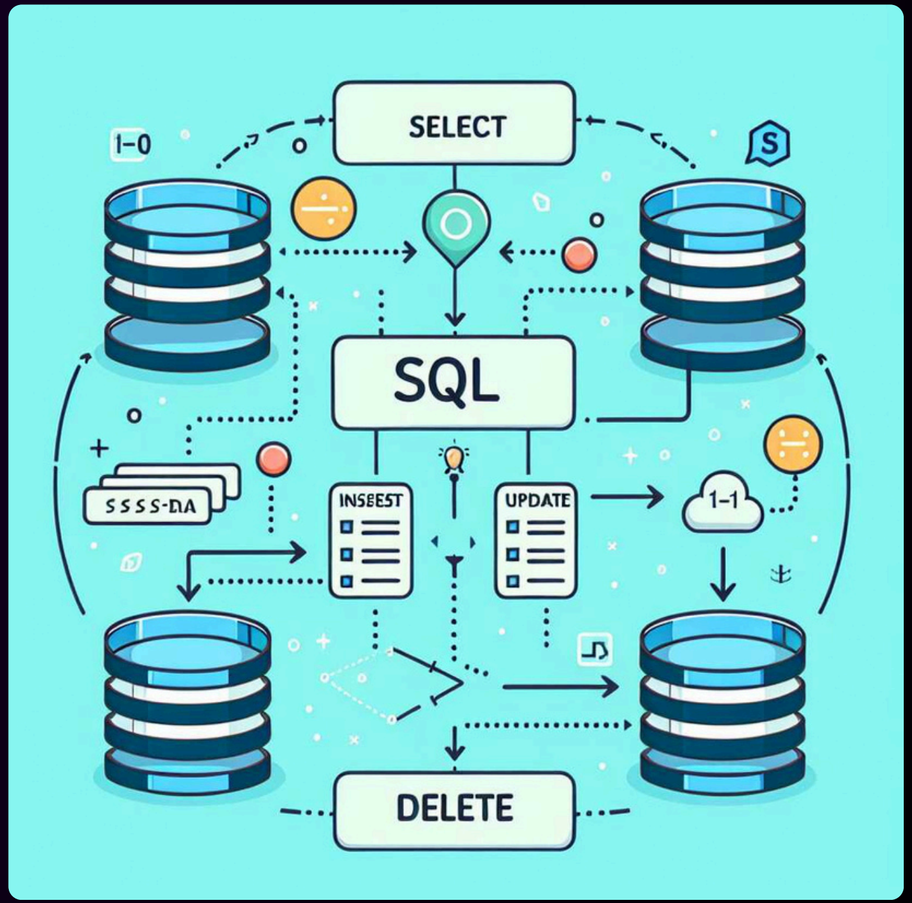
- Install & Import necessary libraries and set up tools to retrieve stellar data from the Gaia DR3 catalog.
- Astroquery: astronomy database query tool,
  - i.e.: SIMBAD, NASA Exoplanet Archive, Vizier
- Astropy: fundamental astronomy toolkit,
  - i.e.: coordinates, units, data formats



Step 2: Data Query

Objective:

- Tries to execute the query up to 3 times in case of network or server errors.
- Searches for stars with a parallax greater than 50
- Converts the result (originally an Astropy Table) into a Pandas DataFrame for easier manipulation.
- Converts distance from parallax to light-years
- Classifies the star based on temperature
- Drops columns and missing data



Step 3: Cleanup

	DESIGNATION	parallax	ra	dec	teff_gspphot	non_single_star
0	Gaia DR3 5853498713190525696	768.066539	217.392321	-62.676075	2829.354248	0
1	Gaia DR3 4472832130942575872	546.975940	269.448503	4.739420	3099.633545	0
2	Gaia DR3 3864972938605115520	415.179416	164.103190	7.002727	NaN	0
3	Gaia DR3 762815470562110464	392.752945	165.830960	35.948653	3511.044922	0
4	Gaia DR3 2947050466531873024	374.489589	101.286626	-16.720933	NaN	0

Profiling and Standardizing Column Names: After inspecting the column names, they are standardized to lowercase to maintain consistency and prevent errors during data processing.

Stellar Classification: A new column, stellar_class, is created by applying a function (get_stellar_class) to the temperature column (teff_gspphot), categorizing stars based on their temperature.

Distance Calculation: The parallax values are converted into light years using the (convert_parallax_to_lightyears) function, providing a more intuitive measure of stellar distances.

Step 3: Cleanup

Handling Missing Data: Rows with critical missing values in columns like (parallax), (teff_gspphot), and (distance_ly) are removed. The number of records removed is logged to track data quality improvements.

- Records removed due to missing data: 1466

Dropping Unnecessary Columns: Columns irrelevant to the analysis, such as (non_single_star), are removed to streamline the dataset.

Improving Readability: Numerical columns are rounded to enhance readability and presentation quality.

	designation	parallax	ra	dec	teff_gspphot	stellar_class	distance_ly
0	Gaia DR3 5853498713190525696	768.067	217.392	-62.676	2829.0	M	4.246
1	Gaia DR3 4472832130942575872	546.976	269.449	4.739	3100.0	M	5.963
3	Gaia DR3 762815470562110464	392.753	165.831	35.949	3511.0	M	8.304
7	Gaia DR3 4075141768785646848	336.027	282.459	-23.837	3117.0	M	9.706

Step 4: Statistical Analysis

Total stars retrieved:

Functions Used: len(df)

- Retrieves the total number of rows in the dataframe, indicating the total stars retrieved.

Closest and farthest star distances:

Functions Used: df['distance_ly'].min() and df['distance_ly'].max()

- Finds the minimum and maximum values of the distance_ly column, providing the closest and farthest distances, respectively.

Temperature range:

Functions Used: df['teff_gspphot'].dropna()

- Drops missing (NaN) values from the teff_gspphot column to ensure valid statistics.

Other Methods:

- .min() and .max() compute the minimum and maximum values, showing the temperature range.
- .std() calculates the standard deviation of the temperatures.

Stars by Stellar Class:

Functions Used: df['stellar_class'].value_counts().sort_index()

- value_counts() counts occurrences of each unique value in the stellar_class column, while sort_index() ensures the counts are sorted by the class labels for clarity.

Summary of Retrieved Stars:

Total Stars Retrieved: 1160
Closest Star (Shortest Distance): 4.246 light years
Farthest Star (Longest Distance): 65.165 light years
Temp Range: 2,809K - 7,938K
Standard Deviation of Temp: 985.83K

Step 5: Visualization

Distribution of Stellar Classes:

Functions Used: sns.countplot()

- Description:
  - Creates a count plot to visualize the distribution of stars across different stellar classes, using a custom color palette for better categorization.

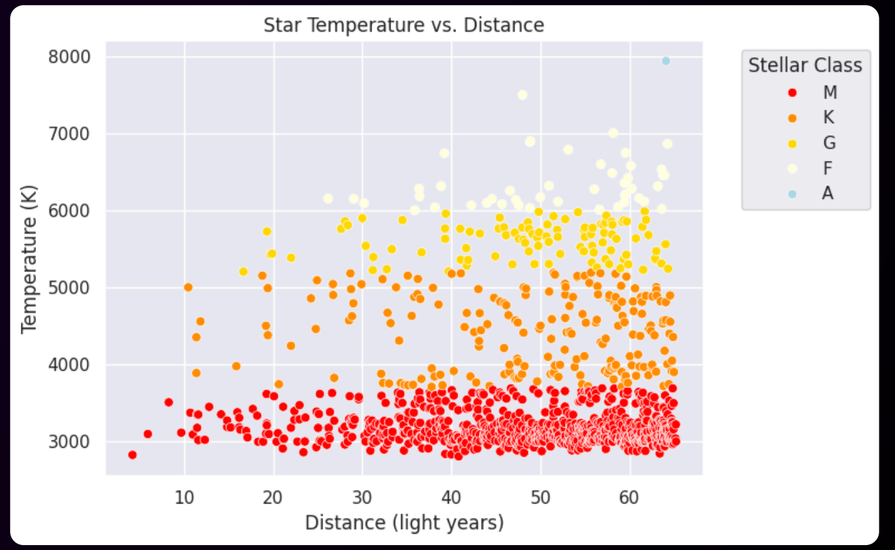
Temperature vs. Distance (Scatter Plot):

Functions Used: sns.scatterplot()

- Description:
  - Plots a scatter plot to visualize the relationship between the stars' temperature (teff_gspphot) and distance (distance_ly), with points color-coded by stellar class.

Stars by Stellar Class (stellar_class):

- A-class: 7,500–10,000 K
  - 1 (White stars).
- F-class: 6,000–7,500 K
  - 47 (Yellow-white stars).
- G-class: 5,200–6,000 K
  - 102 (Yellow stars, like the Sun).
- K-class: 3,700–5,200 K
  - 206 (Orange stars).
- M-class: <3,700 K
  - 804 (Red stars).



Step 5: Visualization

Temperature Outliers by Stellar Class:

Functions Used: seaborn and sns.boxplot()

- Description:
  - Creates a boxplot to show the distribution of temperatures within each stellar class. Outliers are highlighted, and median values are indicated in the plot for each class.

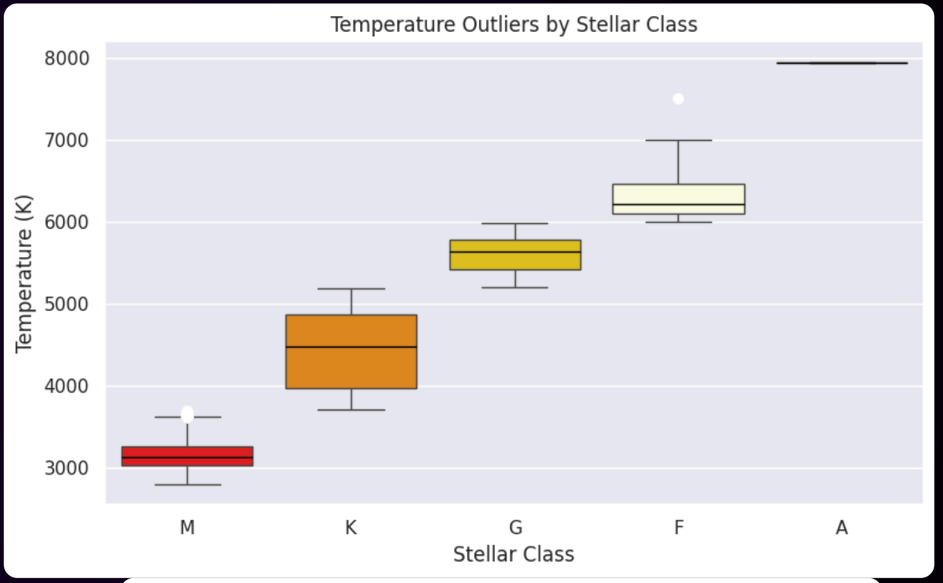
Stellar Class Distribution (Pie Chart):

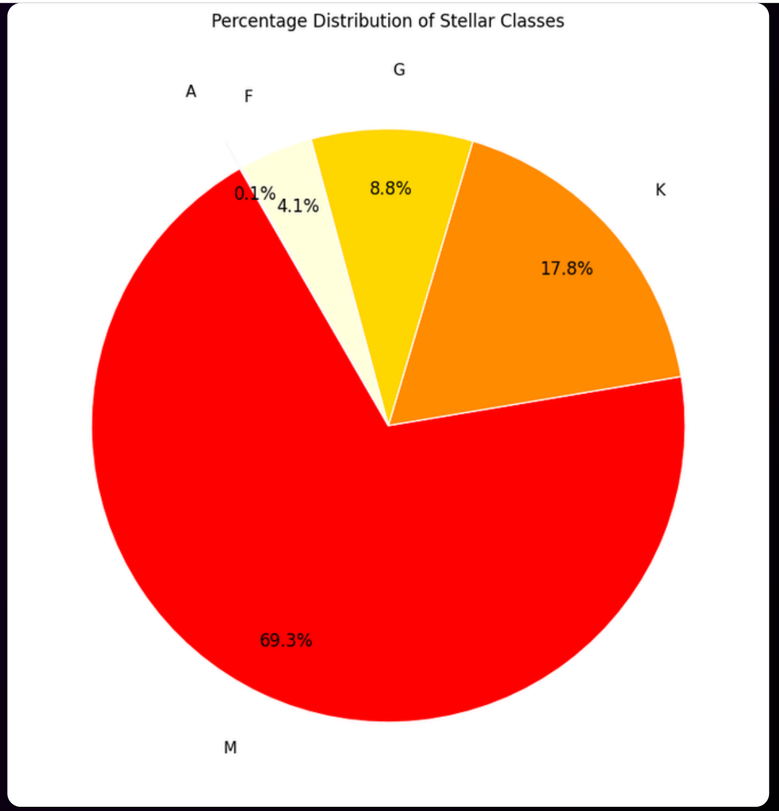
Functions Used: plt.pie()

- Description:
  - Generates a pie chart representing the percentage distribution of stars across different stellar classes. Smaller segments are "exploded" for better visibility.

Stars by Stellar Class (stellar_class):

- A-class: 7,500–10,000 K
  - 1 (White stars).
- F-class: 6,000–7,500 K
  - 47 (Yellow-white stars).
- G-class: 5,200–6,000 K
  - 102 (Yellow stars, like the Sun).
- K-class: 3,700–5,200 K
  - 206 (Orange stars).
- M-class: <3,700 K
  - 804 (Red stars).





Step 6: Export data

We are exporting to SQLite database

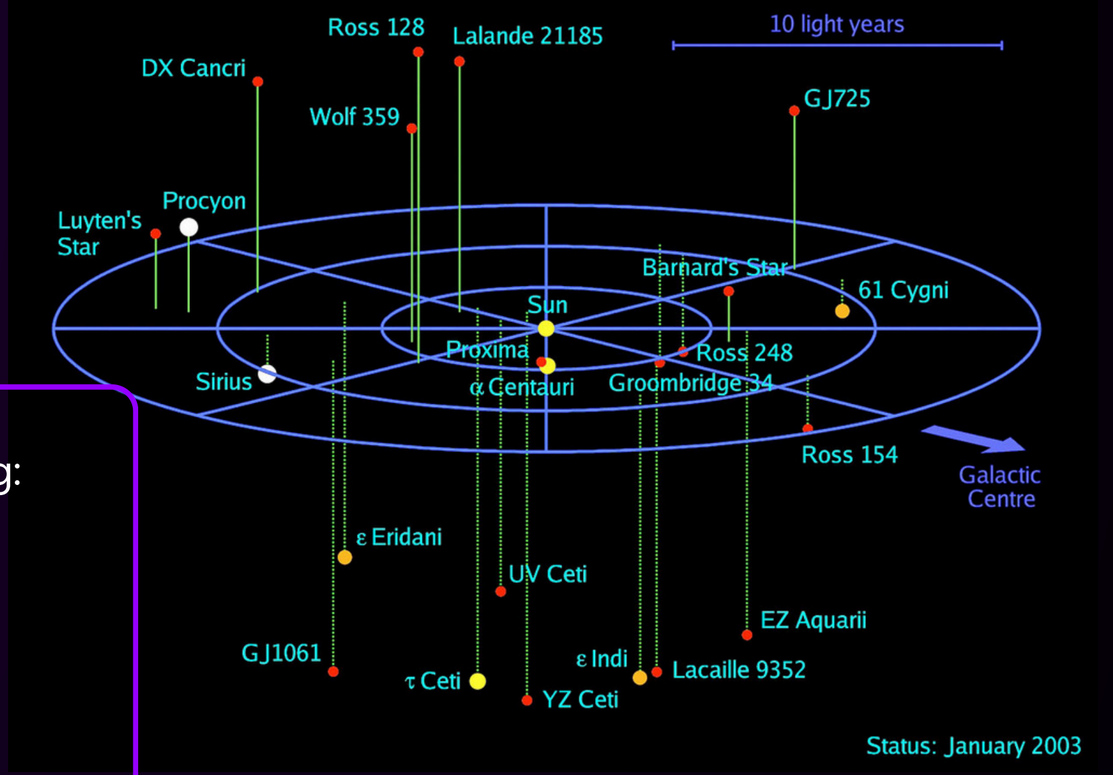
Process:

- SQLite: connect to SQLite database
- Convert data frame to SQL

Conclusion

We created a comprehensive Python program which does the following:

- Queries the GAIA database via API
- Converts difficult to understand parallax data into light-years
- Converts temperature ranges into classes of stars
- Removes bad data and columns
- Saves the results to a CSV file
- Exports the results to a SQLite database format
- Performs basic data analysis on results: min, max, farthest
- Creates 4 visualizations: a histogram, scatterplot, pie chart and boxplot



THANK YOU

FOR YOUR ATTENTION