**How to Create and Integrate New Reports**

The reporting system is designed to be "plug-and-play." To add a new report, you simply create a new Python file in the contest\_tools/reports/ directory that follows a specific template. The system will automatically discover and make it available.

The Report Interface

Every report file must contain a class named Report that inherits from ContestReport. This base class ensures that every report has a consistent structure. You must implement the following four items:

1. report\_id (property): A unique, machine-readable string for your report (e.g., band\_summary). This is what the user will type on the command line.
2. report\_name (property): A human-readable name for your report (e.g., "QSOs per Band Summary").
3. report\_type (property): Must be one of the following strings: 'text', 'plot', or 'chart'. This determines which subdirectory the output files will be saved in.
4. generate(self, output\_path: str, \*\*kwargs) (method): This is where your main logic goes. It receives the base output path for the report and a dictionary of optional arguments.
   * The method is responsible for saving its own output file(s) to the provided output\_path.
   * It should return a string that serves as a confirmation message to be printed to the console (e.g., "Report saved to...").

Step 1: Create Your Report File

Navigate to the contest\_tools/reports/ directory. Create a new Python file. The name should be descriptive, for example, text\_multiplier\_list.py.

Step 2: Use a Template

Copy the contents of one of the existing report files (text\_summary.py, plot\_qso\_rate.py, etc.) into your new file. This gives you the correct structure to start from.

Step 3: Customize Your Report Class

Modify the Report class in your new file:

1. Update report\_id and report\_name to be unique and descriptive for your new report.
2. Set report\_type to 'text', 'plot', or 'chart'.
3. Write your logic in the generate method.
   * You can access the loaded logs via self.logs.
   * For each log in self.logs, you can get its data with log.get\_processed\_data() (which returns a DataFrame) and its metadata with log.get\_metadata().
   * Safely access optional arguments from the kwargs dictionary (e.g., include\_dupes = kwargs.get('include\_dupes', False)).
   * Use Pandas to perform your analysis on the DataFrame(s).
   * Use a library like Matplotlib/Seaborn to create your graphics, save them to a file in the output\_path, and return a confirmation message.

Step 4: Run It!

That's it. There are no other files to edit. The reports/\_\_init\_\_.py file will automatically discover your new report the next time you run the program.

You can see your new report listed by running:

python main\_cli.py --report

You can generate your new report with:

python main\_cli.py --report your\_new\_report\_id k3lr.log