PokémonStats Official Documentation

By: Garret Gallo

Written on: July 10, 2025

Section 1: Introduction

The objective of this project was straightforward: build a streamlined ETL pipeline that delivers clear, actionable insights to help players optimize their team preparation. Currently, The Pokémon Company does not provide any public analytics on tournament trends - meaning there's no easy way to track Pokémon picks, item usage, move distributions, or Tera types for a single event, let alone over a series of tournaments. Without these statistics, competitors can find themselves at a serious disadvantage simply because they lack the data to make informed decisions.

While others have attempted and forged their own solutions to combat this problem - I elected to solve this problem using my ETL and data visualization skills to come up with my own unique result. The final product is a clean and detailed analytics dashboard that any player, whether new or veteran, can use to optimize their teambuilding process.

Section 2: Process + Final Design

While I will describe my thought process in more detail, the image below serves as an overview of the final ETL pipeline that was used to create that final result.

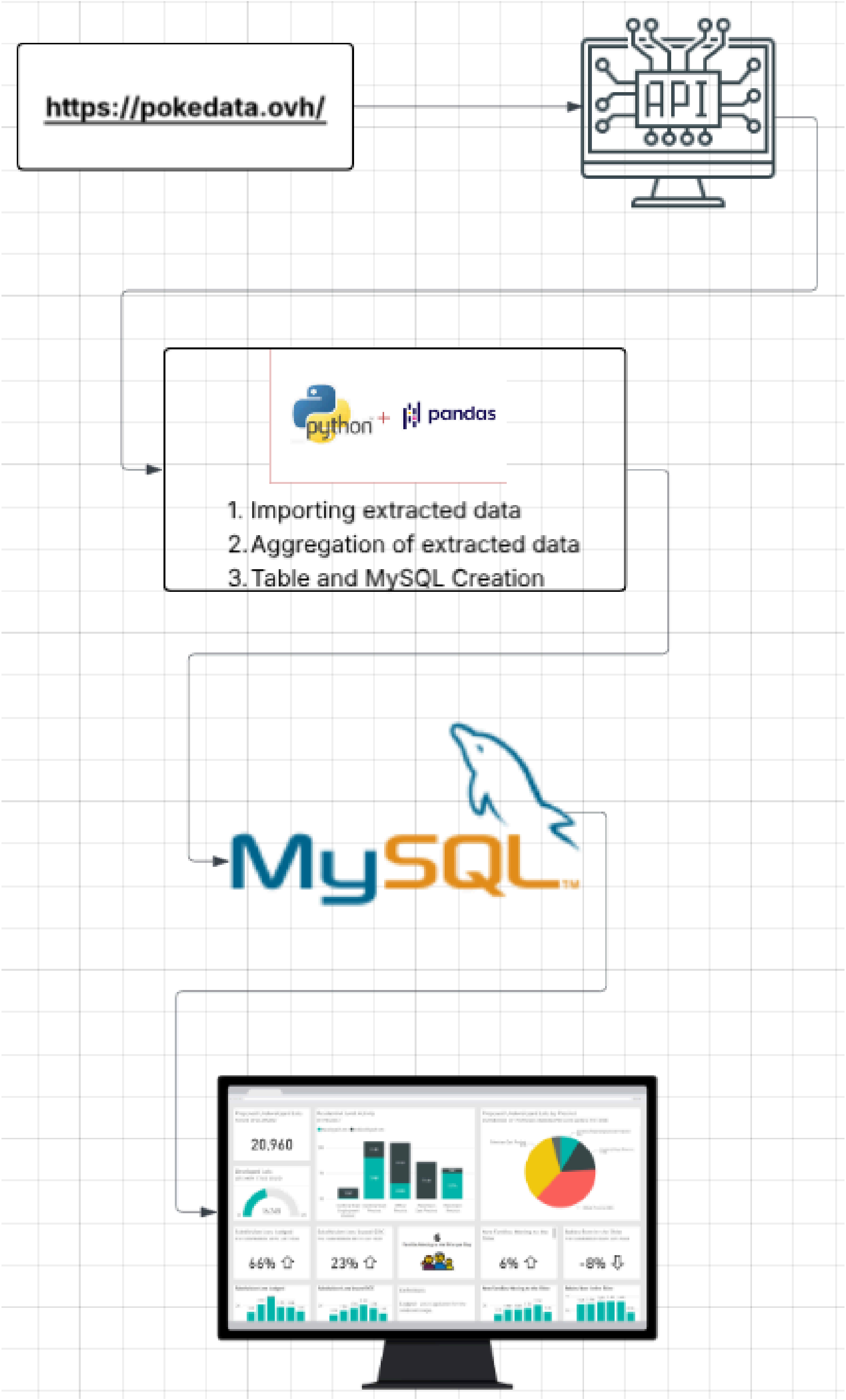

https://pokedata.ovh/
API
python + pandas
1. Importing extracted data
2. Aggregation of extracted data
3. Table and MySQL Creation
MySQL
20,960
66%
23%
6%
-8%

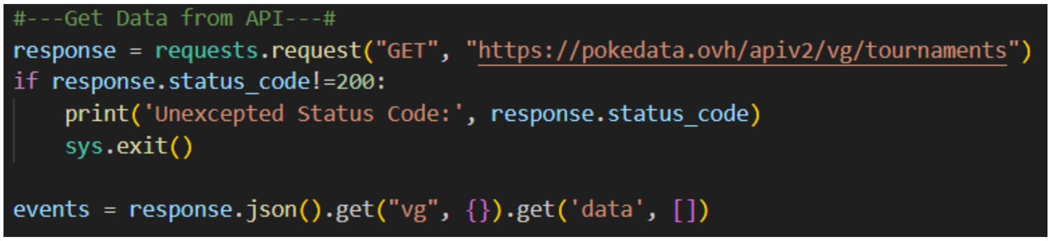
Image 1: ETL Pipeline

Section 2.1: Extraction

What made this project in particular challenging was the extraction methods, since there were no official resources to pull from. I ended up consulting with the owner of Pokedata, who hosts a third-party website, to see the result of past tournaments for advice on how they went about it. Since Pokedata doesn't have tournament statistics and usages, he was kind enough to let me use his API to pull from!

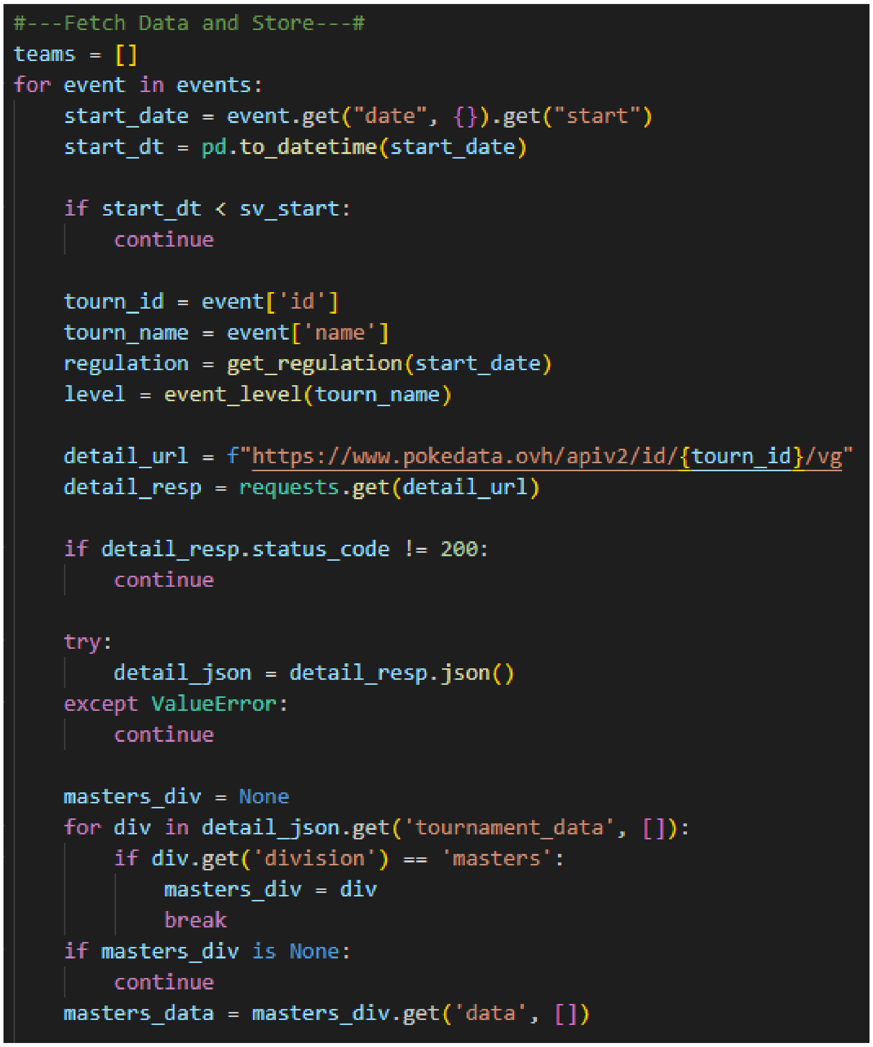
Because the PokeData API splits tournament metadata and team rosters into separate endpoints, our pipeline uses a two-step extraction. First, it makes a single GET request to '/api/v2/vg/tournaments/' to fetch each event's ID, name, and start date. Then, for each remaining tournament ID, it issues a GET request to '/api/v2/id/{tournament_id}/vg' to retrieve the full team rosters, appending every team entry into one unified collection.

This solution ended up being extremely efficient. I was able to extract the data and export it to a raw file without any data loss or other roadblocks.



```
#---Get Data from API---#
response = requests.request("GET", "https://pokedata.ovh/apiv2/vg/tournaments")
if response.status_code!=200:
    print('Unexcepted Status Code:', response.status_code)
    sys.exit()

events = response.json().get("vg", {}).get('data', [])
```



```
#---Fetch Data and Store---#
teams = []
for event in events:
    start_date = event.get("date", {}).get("start")
    start_dt = pd.to_datetime(start_date)

    if start_dt < sv_start:
        continue

    tourn_id = event['id']
    tourn_name = event['name']
    regulation = get_regulation(start_date)
    level = event_level(tourn_name)

    detail_url = f"https://www.pokedata.ovh/apiv2/id/{tourn_id}/vg"
    detail_resp = requests.get(detail_url)

    if detail_resp.status_code != 200:
        continue

    try:
        detail_json = detail_resp.json()
    except ValueError:
        continue

    masters_div = None
    for div in detail_json.get('tournament_data', []):
        if div.get('division') == 'masters':
            masters_div = div
            break
    if masters_div is None:
        continue
    masters_data = masters_div.get('data', [])
```

Images 2 & 3: Code snippets on the API extraction process

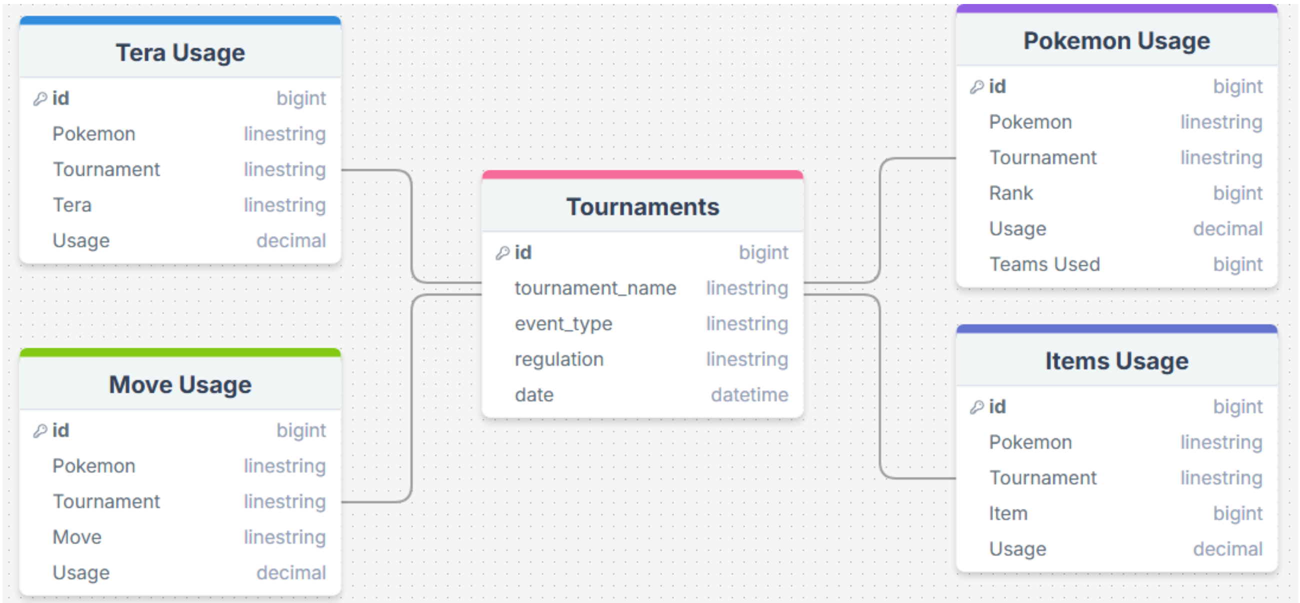
Section 2.2: Transformation

For the purposes of this project, I initially saved the extracted data to a raw Excel file. My reasoning for doing this was that while I was testing my aggregation and MySQL uploads in the future, I didn't want to keep pulling from the API every time in case of rate limits or other potential roadblocks.

The raw Excel file was transformed using pandas to perform the needed aggregations, which included stats for usage, items, moves, and tera types. Each of these aggregated statistics was saved in its own separate pandas dataframe so that they could be easily uploaded to MySQL down the line. I elected these four statistics since they are the most crucial aspects for preparation and what players look for when analyzing the current metagame. I also created a tournament directory dataframe, which will be used down the line in the MySQL database.

Section 2.3: Load and Visualization

Finally, once all my aggregations were complete, I uploaded the final data to MySQL. The following schematic is the outline for the relational database.



Tera Usage	
id	bigint
Pokemon	linestring
Tournament	linestring
Tera	linestring
Usage	decimal

Move Usage	
id	bigint
Pokemon	linestring
Tournament	linestring
Move	linestring
Usage	decimal

Tournaments	
id	bigint
tournament_name	linestring
event_type	linestring
regulation	linestring
date	datetime

Pokemon Usage	
id	bigint
Pokemon	linestring
Tournament	linestring
Rank	bigint
Usage	decimal
Teams Used	bigint

Items Usage	
id	bigint
Pokemon	linestring
Tournament	linestring
Item	bigint
Usage	decimal

Image 4: MySQL database schematic

I chose to have Tera, Move, Pokemon, and Item Usage all in their own separate tables. This creates a more concise and organized method of storing the data, as well as making the visualization process down the road much easier. I choose to have tournaments and all of the event information in its own table, almost to be the 'central database' that all the usage stats connect to. Having all event information in 1 table is extremely optimal as you can have deleted information about each event, without creating unnecessary columns in other tables.

From there, all I had to do was simply connect Power BI and MySQL for the final visualizations. Examples of the visualization can be seen in the GitHub Repository.

Section 3: Future Improvements

While the goal of this project was successfully completed - there are definitely a handful of improvements that could be made to improve efficiency, durability, and the visualization of the final product.

1. Using a NoSQL Database for the raw data

   While there might not be anything inherently wrong with storing the extracted data in an Excel file, using a NoSQL database like MongoDB would be a more preferable option.

   The upside to using MongoDB would be that it can handle highl volumes of data, and Excel can get extremely lagging or even corrupted if the file becomes too large. MongoDB also includes better querying and indexing, especially as the database grows to considerable sizes. Finally, MongoDB also has seamless integration with Python, making it an extremely reliable choice.

2. Additional Visualizations

   While there is almost every statistic needed to successfully prepare for the next major tournament, there is one aspect missing. A feature I am looking to add in the future is “most common partner”. This would show what Pokémon are used together in sets of 2, 3, and 4 (Example: Urshifu Rapid-Strike + Amoonguss (25.6%).

   This statistic would add another dimension to team building and preparation, as there would be visuals to see what is being paired together, as well as common cores.

3. Updated Platform

   While Power BI is a very powerful platform that my data scientists and analysts use with the ability to publish your reports to the web for large scale sharing. For the purposes of this project this was sufficient, however in my opinion other platforms could be better to display this data on a large scale.

   Potential solutions could be platforms such as Heroku, integrating the Power BI dashboards on an integrated web app, or other BI services such as Tableau. These solutions ideally should not be overly difficult to implement - as a bulk of the work is already done, especially since the MySQL are already created.