Data Engineering 1: Big Data Databases

By Team 9
Gaurav Naik

Project Report

1. *Crawling a Website*

Initially, during the team meeting, we discussed the possible websites to crawl. Since, I have never done scraping at a large scale, I learnt about static websites which are easier to crawl and dynamic websites which are difficult. Post our team meeting, we decided on the theme of ‘Anime’ websites.

Considering the ‘Anime’ theme, my team decided on 3 websites: MyAnimeList (data crawling), AnimeNewsNetwork (image crawling) and AniDB (data crawling). I took up the task of crawling the AniDB website.

Due to my lack of experience in web crawling, I learnt crawling with Selenium and bs4. Furthermore, I went through parts of crawling with scrapy. Using selenium and bs4, I coded the crawling of anime items in each page of AniDB website while following pagination logic.

When running the code in VSCode, I used chrome as a webdriver in incognito mode. The code ran in such a way that, it opens a chrome tab in incognito mode and crawls the data from the table item wise(anime wise) in the website starting from page 0, the code further runs crawling from the next pages while following a pagination logic.

Unfortunately, when we tried to run the same code in our cluster, since chrome driver or chrome wasn’t installed in the system and installing chrome driver required multiple packages installation which in turn complicated the complete integration process of the crawling code.

Ultimately, due to the above issue, I settled on crawling the website with scrapy as the code runs without any further package installation.

2. *Setting up Hadoop Cluster as worker node*

Coordinated with teammates to join a swarm as a worker node with the help of Tailscale.

3. *Business Queries*

Made several business queries. Below are the queries with a short description.

- Query to highlight versatile anime producers with multiple-genre involvement, sorting by average popularity.

```
SELECT Producers.Producers, COUNT(DISTINCT Genres.gID) AS
NumberOfGenres, AVG(descriptions.Popularity) AS
AveragePopularity
FROM Producers
JOIN animeProducers ON Producers.pID = animeProducers.pID
JOIN descriptions ON animeProducers.malID = descriptions.malID
JOIN animeGenres ON descriptions.malID = animeGenres.malID
JOIN Genres ON animeGenres.gID = Genres.gID
GROUP BY Producers.Producers
HAVING NumberOfGenres > 1
ORDER BY AveragePopularity DESC;
```

- Query to Find Average Score for Each Genre:

```
SELECT g.Genres, AVG(d.Score) AS AverageScore
FROM anime.Genres g
JOIN anime.animeGenres ag ON g.gID = ag.gID
JOIN anime.descriptions d ON ag.malID = d.malID
GROUP BY g.Genres
ORDER BY AverageScore DESC;
```

- Query to Find Studios with the Most Released Episodes:

```
SELECT s.Studios, SUM(d.Episodes) AS TotalEpisodes
FROM anime.Studios s
JOIN anime.animeStudios ast ON s.sID = ast.sID
JOIN anime.descriptions d ON ast.malID = d.malID
GROUP BY s.Studios
ORDER BY TotalEpisodes DESC
```

- Retrieve Anime with Specific Genre (e.g., 'Action'):

```
SELECT d.*, g.Genres
FROM anime.descriptions d
JOIN anime.animeGenres ag ON d.malID = ag.malID
JOIN anime.Genres g ON ag.gID = g.gID
WHERE g.Genres = 'Action';
```

- Query to Retrieve Anime with a Certain Producer:

```
SELECT d.*, p.Producers
FROM anime.descriptions d
JOIN anime.animeProducers ap ON d.malID = ap.malID
JOIN anime.Producers p ON ap.pID = p.pID
WHERE p.Producers = 'Madhouse';
```

4. Preparing the video

Preparing the video by trimming and joining clips to form a single video.

5. Peer Review Table

Name	**Grade**
Praneet Patnaik	A
Anton J Vikranth	A
Pavankumar Managoli	A

Although we divided our responsibilities and tasks, our collective contributions were important in addressing every aspect of the project, enhancing my overall understanding of the concepts.