AMAZON PRIME RECOMMENDATIONS USE CASE STUDY REPORT

Group 25

Student Names: Srinidhi Aduri and Surbhi Virendra Wahie

Introduction:

The Problem:

Users of amazon are increasingly dissatisfied with the recommendations provided by amazon prime video. the data is driven by overall popularity which leads to poor personalization, leading to an increase in customer churn rate.

The Goal:

We aim to solve this problem by using Amazon's in-house data capabilities. Using Amazon's rich database, we aim to improve prime user's movie recommendation system by making use of the user's Amazon music database.

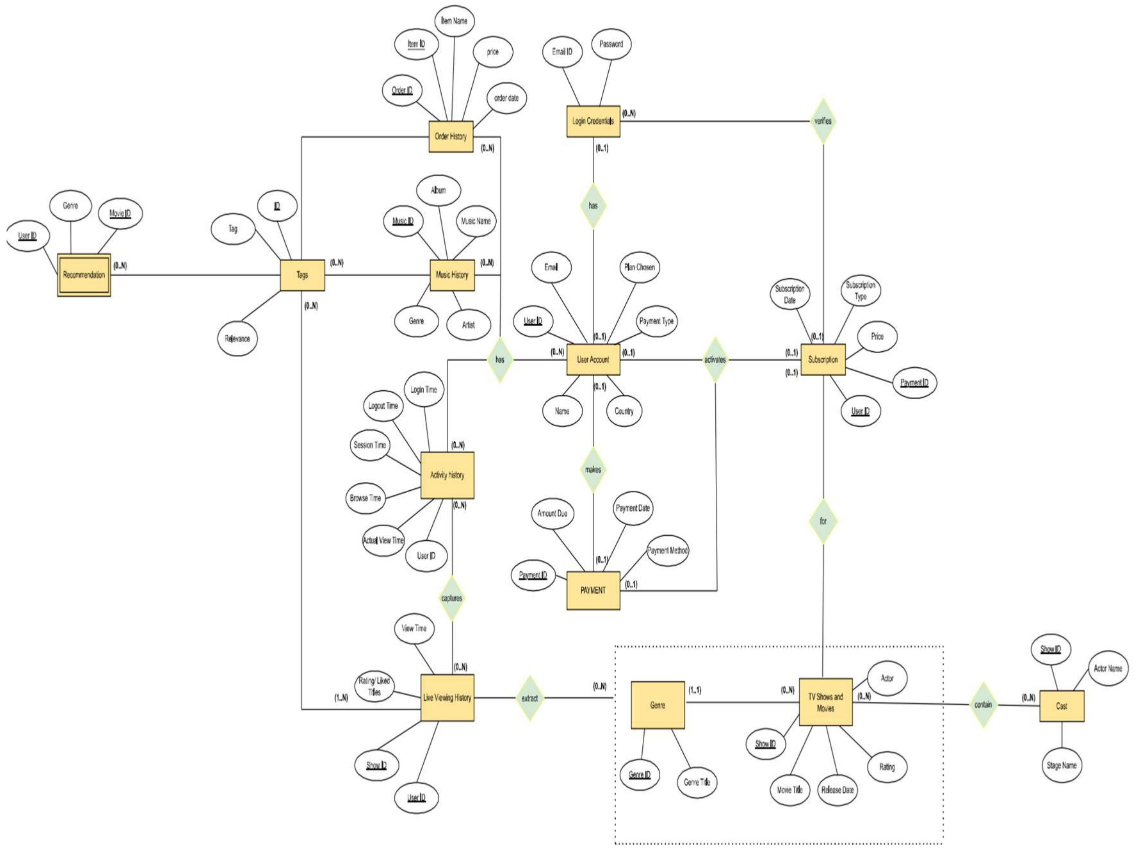
The Requirement:

For popular video streaming platforms, there exist taggers that watch through the whole movie, analyze it on various parameters and add a tag that describes the personality traits of the characters – like quirky, or pretentious.

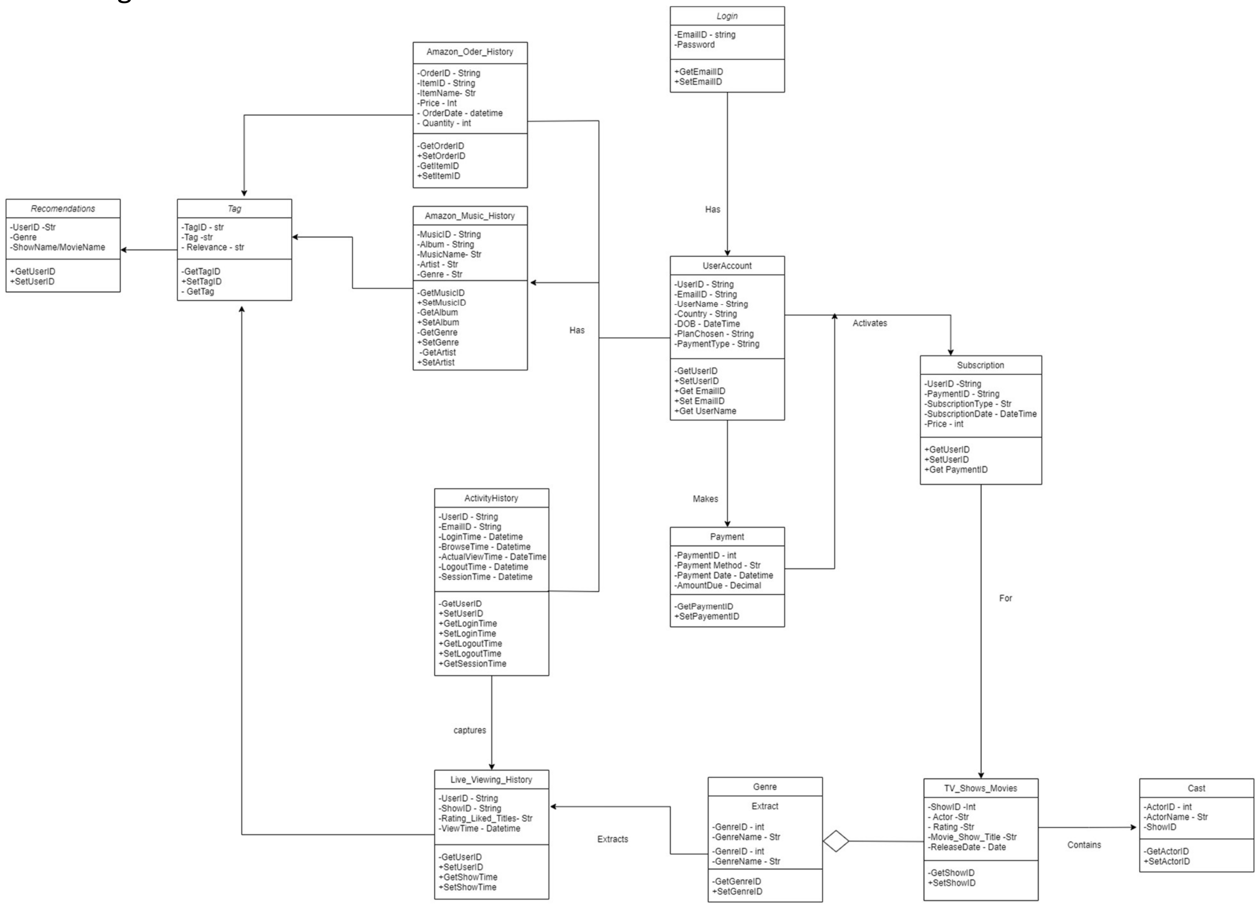
To reach our goal, we need to generate tags for music that the same user is listing to. This can be done by scraping YouTube comments and studying users' mood that classify contents by genre and microgenre, time-period, plot conclusion, and mood, like that of a movie tag.

<u>Conceptual data Modelling</u>

EER Diagram:



UML Diagram:



<u>Mapping Conceptual Model to Relational Model</u>

Relational Model:

USERACCOUNT (USER   ID, NAME, EMAILID, PAYMENTID, COUNTRY, PLANCHOSEN,

PAYMENTTYPE)
EMAILID: FOREIGN KEY FROM LOGINCREDENTIALS – NOT NULL PAYMENTID: FOREIGN KEY FROM PAYMENT – NULL ALLOWED

LOGINCREDENTIALS (EMAILID, PASSWORD)
PAYMENT (PAYMENTID, PAYMENTMETHOD, AMOUNT, PAYMENTDATE)
SUBSCRIPTION (USERID, PAYMENTID, SUBSCRIPTIONTYPE, PRICE,

SUBSCRIPTIONDATE)
USERID: FOREIGN KEY FROM USERACCOUNT – NOT NULL PAYMENTID – FOREIGN KEY FROM PAYMENT

ACTIVITYHISTORY (USERID, LOGINTIME, LOGOUTTIME, SESSIONTIME,

BROWSETIME, ACTUALTIME) (DO WE NEED A PRIMARY KEY SPECIFICALLY FOR

ACTIVITY HISTORY?)
USERID: FOREIGN KEY FROM USERACCOUNT
LIVEVIEWINGHISTORY (SHOWID, USERID, VIEWTIME, RATINGS/LIKES)

GENRE (GENREID, GENRETITLE)
TVSHOWSANDMOVIES (SHOWID, MOVIETITLE, RELEASEDATE, RATING)
SHOWGENRE (GENREID, SHOWID) (DO WE WANT A SHOW TO BE TAGGED TO MULTIPLE GENRES?)
CAST (SHOWID, ACTORNAME, STAGENAME)

*ORDERDETAILS (ORDERID, ITEMID, ITEMNAME, ITEMPRICE, ORDERDATE)

ORDERHISTORY (USERID, ORDERID)

ORDERID: FOREIGN KEY FROM ORDERDETAILS

*MUSICDATA (MUSICID, MUSICNAME, GENRE, ARTIST, ALBUM)

MUSICHISTORY (USERID, MUSICID)

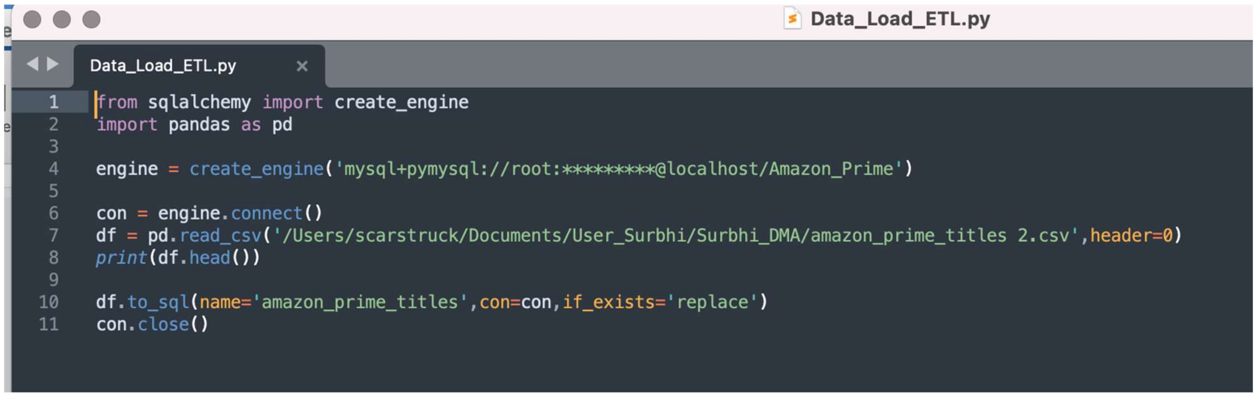
USERID: FOREIGN KEY FROM USERACCOUNT MUSICID: FOREIGN KEY FROM MUSICHISTORY

TAGS (ID, TAG, RELEVANCE)

RECOMMENDATION (USERID, MOVIEID, GENRE) USERID: FOREIGN KEY FROM USERACCOUNT
MOVIEID: FOREIGN KEY FROM MUSICDATA

Implementation of a relational Model via MySQL:

We have sourced the data from Kaggle. The source file was in csv, excel file format and used SqlAlchemy package to impot data into MySql database using Python.



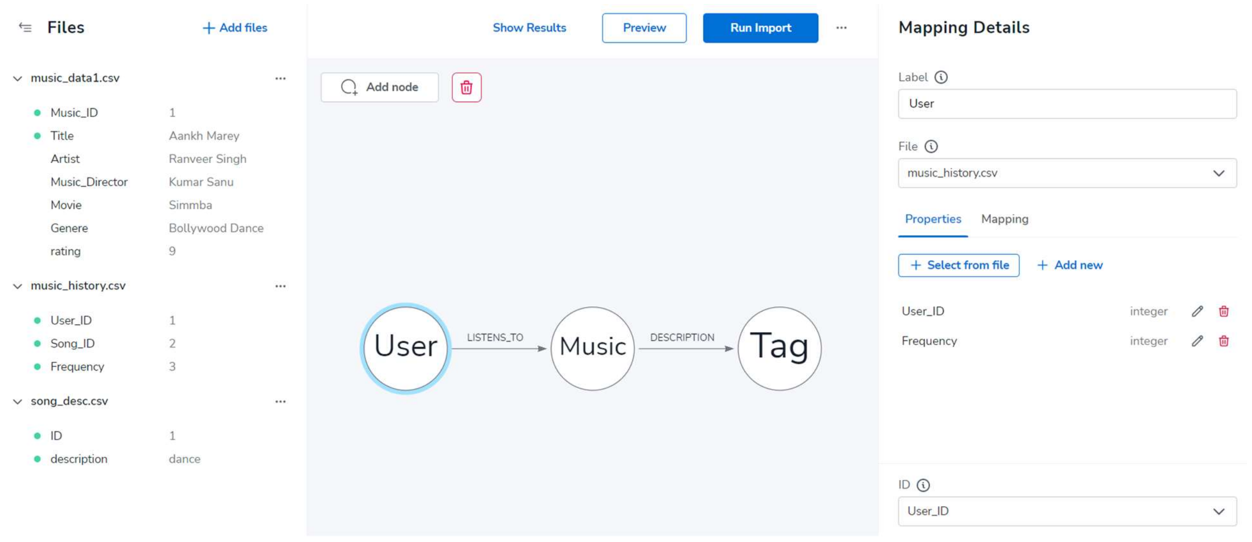
```python
from sqlalchemy import create_engine
import pandas as pd

engine = create_engine('mysql+pymysql://root:*********@localhost/Amazon_Prime')

con = engine.connect()
df = pd.read_csv('/Users/scarstruck/Documents/User_Surbhi/Surbhi_DMA/amazon_prime_titles 2.csv',header=0)
print(df.head())

df.to_sql(name='amazon_prime_titles',con=con,if_exists='replace')
con.close()
```

Database Name: Amazon_Prime

```sql
CREATE TABLE `amazon_prime_titles` (
  `index` bigint DEFAULT NULL,
  `show_id` text,
  `type` text,
  `title` text,
  `director` text,
  `cast` text,
  `country` text,
  `date_added` text,
  `release_year` double DEFAULT NULL,
  `rating` text,
  `duration` text,
  `listed_in` text,
  `description` text,
  KEY `ix_amazon_prime_titles_index` (`index`)
) ;
```

Implementation of a relational Model via NoSQL:

Imported a couple csv files into Neo4j and created relations between them.



Files + Add files

music_data1.csv

Music_ID	1
Title	Aankh Marey
Artist	Ranveer Singh
Music_Director	Kumar Sanu
Movie	Simmba
Genere	Bollywood Dance
rating	9

music_history.csv

User_ID	1
Song_ID	2
Frequency	3

song_desc.csv

ID	1
description	dance

Mapping Details

Label: User

File: music_history.csv

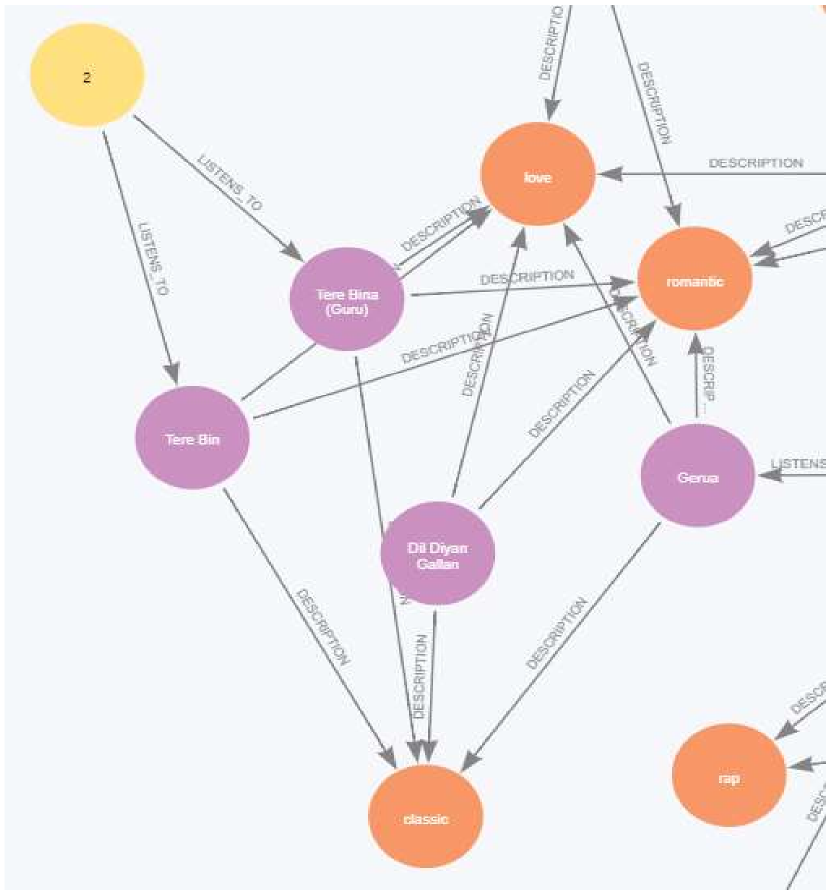
Properties Mapping

+ Select from file + Add new

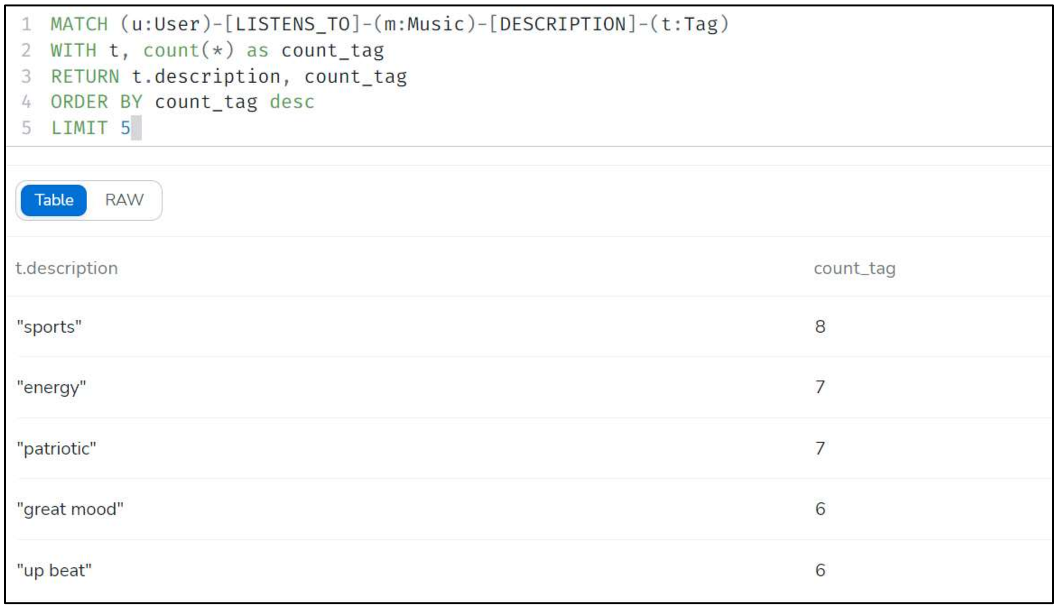
User_ID	integer
Frequency	integer

ID: User_ID

Here is the magnified version of how our cypher database tables looks like:



Query for using the tables in the database to get the list of most listened genres.

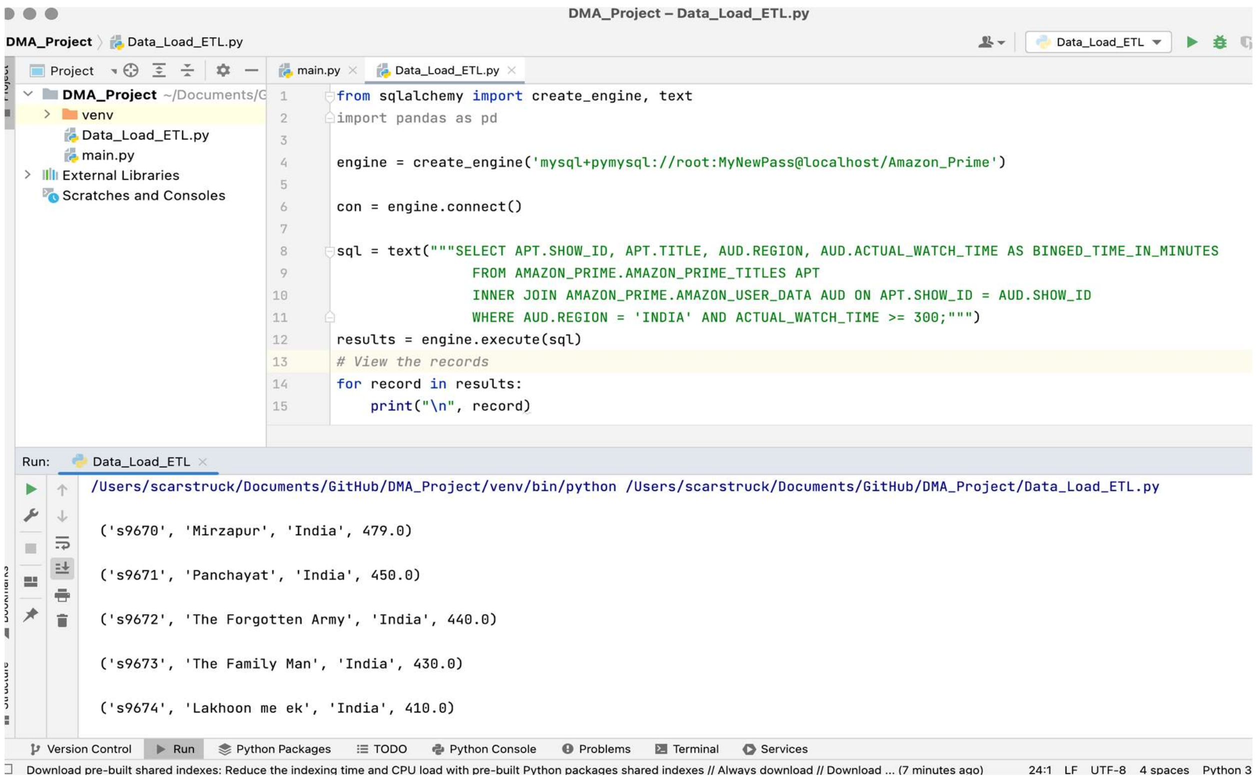


```
MATCH (u:User)-[LISTENS_TO]-(m:Music)-[DESCRIPTION]-(t:Tag)
WITH t, count(*) as count_tag
RETURN t.description, count_tag
ORDER BY count_tag desc
LIMIT 5
```

t.description	count_tag
"sports"	8
"energy"	7
"patriotic"	7
"great mood"	6
"up beat"	6

<u>Database access via Python:</u>

Using the connection code to connect to the MySQL database. Imported SqlAlchemy, created connection, passed the database configurations to create engine, and used the engine to interact with MySQL. Below is the screen capture demonstrating the retrieval of desired data from our Amazon_Prime database and simple analytics based upon the data.



```
from sqlalchemy import create_engine, text
import pandas as pd

engine = create_engine('mysql+pymysql://root:MyNewPass@localhost/Amazon_Prime')

con = engine.connect()

sql = text("""SELECT APT.SHOW_ID, APT.TITLE, AUD.REGION, AUD.ACTUAL_WATCH_TIME AS BINGED_TIME_IN_MINUTES
                FROM AMAZON_PRIME.AMAZON_PRIME_TITLES APT
                INNER JOIN AMAZON_PRIME.AMAZON_USER_DATA AUD ON APT.SHOW_ID = AUD.SHOW_ID
                WHERE AUD.REGION = 'INDIA' AND ACTUAL_WATCH_TIME >= 300;""")
results = engine.execute(sql)
# View the records
for record in results:
    print("\n", record)
```

```
/Users/scarstruck/Documents/GitHub/DMA_Project/venv/bin/python /Users/scarstruck/Documents/GitHub/DMA_Project/Data_Load_ETL.py

 ('s9670', 'Mirzapur', 'India', 479.0)

 ('s9671', 'Panchayat', 'India', 450.0)

 ('s9672', 'The Forgotten Army', 'India', 440.0)

 ('s9673', 'The Family Man', 'India', 430.0)

 ('s9674', 'Lakhoon me ek', 'India', 410.0)
```

Download pre-built shared indexes: Reduce the indexing time and CPU load with pre-built Python packages shared indexes // Always download // Download ... (7 minutes ago)   24:1 LF UTF-8 4 spaces Python 3

Summary and Recommendation

As existing users of Amazon prime video, we have developed an understanding of the level of personalization the platform needs when compared to other major players in the streaming services. Through our analysis, we found a way to recommend to customers and identify niche patterns in user behavior trends using Amazon's in-house cross-domain datasets, making the process **cost-effective.**

By improving recommendations based on persona, we ensure that the customer's watch time is increased and browse time is decreased, leading to better engagement and reduction in churn rate