98point6 Data Engineer Assignment

Preparation:

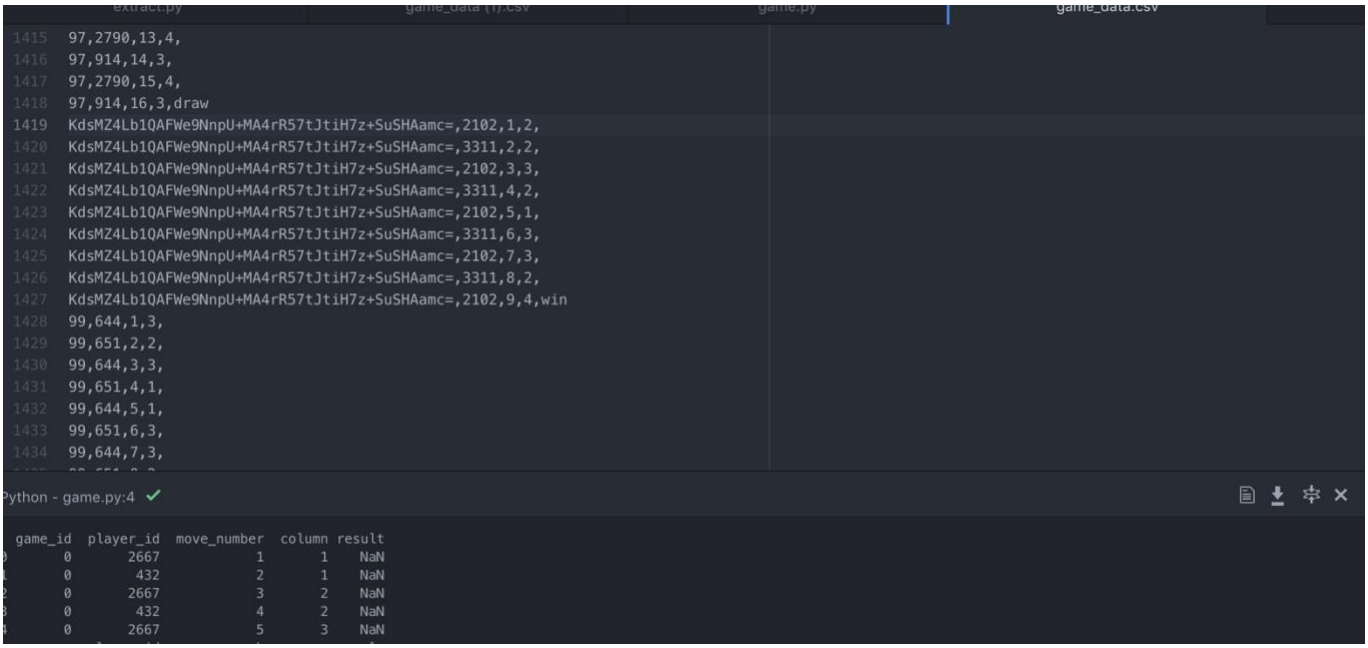
- Understand the schema
- View player data to be extracted using REST API
- Read and Re-read the requirements
- Design the cleanest and most efficient solution
- Weed out the anomalies in data
- Store data
- Report the findings

Technology Stack:

- Programming (ETL): Python
- Persistent Storage: Postgres local instance
- Analytics: SQL

Steps:

- Extraction
    - game.py to read csv
    - extract.py to write player data into csv
- Transform
    - game.py to clean the data
        - remove corrupt game_id
        - replace it with average of previous and next game
        - storing the clean data in csv



Invalid game_id

- Load
    - Create local postgres instance
    - Create game table for games
    - Create player table for players
- Analytics
    - Design SQL query for all 3 solutions

game.py

```
import pandas as pd

# read csv
df=pd.read_csv('game_data.csv')

# describe df
#print(df.describe())
```

```
# remove non-int values from game-id
df['game_id']=pd.to_numeric(df['game_id'], errors='coerce', downcast='signed')

# replace na with average of previous and next game_id
df['game_id'] = (df['game_id'].ffill()+df['game_id'].bfill())/2
df['game_id'] = df['game_id'].bfill().ffill()

# cast as int
df['game_id']=df['game_id'].astype(int)
# renaming column to col to avoid conflict in postgres
df.rename(columns={'column': 'col'}, inplace=True)

df.to_csv('game_data_proc.csv', index=False)
```

extract.py

```
import requests
import json
import csv

# create csv in write mode
f = csv.writer(open("country.csv", "wb+"))
f.writerow(["player_id", "country", "gender"])

# read from pages
for page in range(500):
    url = 'https://x37sv76kth.execute-api.us-west-
1.amazonaws.com/prod/users?page=%s'%page
    data = requests.get(url).json()

    # write to csv
    for row in data:
        f.writerow([row['id'],row['data']['nat'],row['data']['gender']])
```

Postgresql database connection

mac-90676:98point6 aashrayyadav$ /Applications/Postgres.app/Contents/Versions/11/bin/psql -p5432 "aashrayyadav"

Postgres game table creation and population

```
                                                                ^
aashrayyadav=# CREATE TABLE game ( id SERIAL, game_id int, player_id int, move
nt, col int, result varchar);
CREATE TABLE
aashrayyadav=# /dt
aashrayyadav-# \dt
          List of relations
 Schema | Name | Type  |    Owner
--------+------+-------+-------------
 public | game | table | aashrayyadav
(1 row)

aashrayyadav-#
```

game table in database

```
        ^
aashrayyadav=# COPY game(game_id,player_id,move,col,result) FROM '/Users/aashray
yadav/desktop/98point6/game_data_proc.csv' DELIMITER ',' CSV HEADER;
COPY 145723
aashrayyadav=#
```

Copying data from csv to the table

```
COPY 145723
aashrayyadav=# select * from game;
   id   | game_id | player_id | move | col | result
--------+---------+-----------+------+-----+--------
      1 |       0 |      2667 |    1 |   1 |
      2 |       0 |       432 |    2 |   1 |
      3 |       0 |      2667 |    3 |   2 |
      4 |       0 |       432 |    4 |   2 |
      5 |       0 |      2667 |    5 |   3 |
      6 |       0 |       432 |    6 |   1 |
      7 |       0 |      2667 |    7 |   2 |
      8 |       0 |       432 |    8 |   4 |
      9 |       0 |      2667 |    9 |   1 |
     10 |       0 |       432 |   10 |   2 |
     11 |       0 |      2667 |   11 |   3 |
     12 |       0 |       432 |   12 |   4 |
     13 |       0 |      2667 |   13 |   4 |
     14 |       0 |       432 |   14 |   3 |
     15 |       0 |      2667 |   15 |   3 |
     16 |       0 |       432 |   16 |   4 | draw
     17 |       1 |      3857 |    1 |   4 |
     18 |       1 |       351 |    2 |   1 |
     19 |       1 |      3857 |    3 |   2 |
     20 |       1 |       351 |    4 |   1 |
```

Resulting game table

Postgres player table creation and population

```
aashrayyadav=#
aashrayyadav=# create table player(player_id int primary key, country varchar, gender varchar);
CREATE TABLE
aashrayyadav=# \dt
           List of relations
 Schema |  Name  | Type  |    Owner
--------+--------+-------+-------------
 public | game   | table | aashrayyadav
 public | player | table | aashrayyadav
(2 rows)

aashrayyadav=#
```

Player table in postgres

```
aashrayyadav=# COPY player(player_id,country,gender) FROM '/Users/aashrayyadav/desktop/96point6_th/country.csv' DELIMITER ',' CSV HEADER;
COPY 5000
aashrayyadav=# select * from player;
 player_id | country | gender
-----------+---------+--------
         0 | IE      | male
         1 | ES      | female
         2 | GB      | male
         3 | CH      | male
         4 | TR      | male
         5 | TR      | female
         6 | NZ      | male
         7 | AU      | female
         8 | CA      | male
         9 | FR      | male
        10 | NL      | female
        11 | ES      | female
        12 | ES      | female
        13 | IR      | female
        14 | AU      | female
        15 | ES      | female
        16 | ES      | male
        17 | FR      | male
        18 | ES      | female
        19 | FR      | female
```

Data in player table

Questions and Answers

1. Out of all the games, what is the percentile rank of each column used as the first move in a game? That is, when the first player is choosing a column for their first move, which column most frequently leads to that player winning the game?

Intermediate query helps us map each game-player winning combination and first move to subsequent column

```
aashrayyadav=# select a.game_id, a.player_id, a.col from game a, game b where a.game_id=b.game_id and a.player_id=b.player_id and b.result='win' and a.move=1 group by a.game_id,a.player_id, a.col;
 game_id | player_id | col
---------+-----------+-----
       1 |      3857 |   4
       2 |      1123 |   3
       5 |      2165 |   1
      16 |      2153 |   3
      21 |       836 |   2
      23 |      1376 |   3
      25 |      4738 |   2
      26 |      3729 |   2
      33 |      3888 |   3
      35 |      3745 |   1
      40 |      2963 |   3
      42 |      3809 |   3
      44 |      1035 |   4
      49 |      1068 |   1
      50 |      1012 |   1
      51 |      4525 |   2
      52 |      3832 |   4
      59 |      3311 |   1
      71 |      4969 |   2
      76 |      1421 |   2
      81 |      1429 |   1
      82 |      4119 |   2
      89 |      2102 |   3
      98 |      2102 |   2
     103 |       393 |   1
     104 |       995 |   1
     105 |      3778 |   3
     120 |      1225 |   1
     123 |      2759 |   4
     125 |      3409 |   4
```

Final query plots the count of cols corresponding to first move resulting in that player winning the game

```
select a.game_id, a.player_id, a.col from game a, game b where
a.game_id=b.game_id and a.player_id=b.player_id and b.result='win' and
a.move=1 group by a.game_id,a.player_id, a.col;
```

```
aashrayyadav=# select count(c.cols), c.cols from (select a.game_id, a.player_id, a.col
up by a.game_id,a.player_id, a.col) c group by c.cols order by count desc;
 count | cols
-------+------
   794 |    4
   776 |    1
   656 |    2
   649 |    3
(4 rows)

aashrayyadav=#
```

As we can see, column 4 has resulted in the most number of wins for the first player.

2. How many games has each nationality participated in?

First, assimilate information based on game-player information across nationalities. Each game would be played between two players as shown below with their respective nationalities.

Next, group the counts of these values based on nationalities and display in descending order.

```
aashrayyadav=# select c.country, count(c.country) as count from (select game_id, a.player_id, country from game a, player b where a.player_id=b.player_id group by game_id, a.player_id, country) as c group
 by c.country;
 country | count
---------+-------
 DK      |  1190
 AU      |  1205
 BR      |  1087
 DE      |  1166
 FI      |  1308
 IE      |  1505
 ES      |  1289
 TR      |  1354
 US      |  1433
 NZ      |  1098
 IR      |  1071
 NL      |  1323
 CH      |  1447
 CA      |  1296
 FR      |  1139
 GB      |  1089
(16 rows)

aashrayyadav=# select c.country, count(c.country) as count from (select game_id, a.player_id, country from game a, player b where a.player_id=b.player_id group by game_id, a.player_id, country) as c group
 by c.country order by count desc;
 country | count
---------+-------
 IE      |  1505
 CH      |  1447
 US      |  1433
 TR      |  1354
 NL      |  1323
 FI      |  1308
 CA      |  1296
 ES      |  1289
 AU      |  1205
 DK      |  1190
 DE      |  1166
 FR      |  1139
 NZ      |  1098
 GB      |  1089
 BR      |  1087
 IR      |  1071
(16 rows)

aashrayyadav=#
```

As can be seen, IE has the most players playing 9dt with IR having the least.

Final query

```
select c.country, count(c.country) as count from (select game_id, a.player_id, country from game a, player b where a.player_id=b.player_id group by game_id, a.player_id, country) as c group by c.country order by count desc;
```

3. Marketing wants to send emails to players that have only played a single game. The email will be customized based on whether or not the player won, lost, or drew the game. Which players should receive an email, and with what customization?

With the first query, we try to find no of games played by each player and the associated result

```
select count(game_id) as games, player_id, result  from game group by player_id, result;
```

```
aashrayyadav=# select count(game_id) as games, player_id, result  from game group by player_id, result;
 games | player_id | result
-------+-----------+--------
     3 |      2227 | draw
     2 |      4109 | draw
     3 |       146 | draw
     2 |      1723 | win
    30 |      4064 |
     3 |       472 | win
     3 |      1955 | draw
     1 |      2719 | draw
     1 |       472 | draw
    15 |      3005 |
    36 |      1507 |
    33 |      3755 |
    22 |      1037 |
    40 |      3866 |
    11 |      4843 |
    38 |      2306 |
     1 |      2866 | win
     6 |       843 |
    23 |      2147 |
    36 |      3390 |
     2 |      1305 | draw
     2 |       517 | draw
     2 |      3440 | win
    38 |      2379 |
     1 |       705 | win
     1 |      1056 | win
     1 |      4053 | win
     1 |      4639 | draw
    37 |      4603 |
    21 |      3004 |
     3 |      4547 | draw
    14 |        39 |
    14 |      3624 |
     1 |      3636 | draw
    67 |       411 |
     3 |      2195 | draw
    35 |       715 |
     2 |      3278 | win
     3 |      3986 | draw
    52 |       972 |
     1 |      2781 | draw
     1 |      2522 | draw
    44 |       711 |
    52 |      3378 |
     1 |      4061 | win
     7 |       230 |
     1 |       231 | win
```

With second, query, we find all players who have played just one game

```
select count(game_id) as games, player_id, result  from game group by player_id, result having count(game_id)=1;
```

```
aashrayyadav=# select count(game_id) as games, player_id, result  from game group by player_id, result having count(game_id)=1;;
 games | player_id | result
-------+-----------+--------
     1 |      2719 | draw
     1 |       472 | draw
     1 |      2866 | win
     1 |       705 | win
     1 |      1056 | win
     1 |      4053 | win
     1 |      4639 | draw
     1 |      3636 | draw
     1 |      2781 | draw
     1 |      2522 | draw
     1 |      4061 | win
     1 |       231 | win
     1 |       341 | win
     1 |       263 | draw
     1 |      1388 | win
     1 |      1725 | draw
     1 |      4518 | draw
     1 |       644 | draw
     1 |      3817 | draw
     1 |      4181 | draw
     1 |      1704 | win
     1 |      1093 | win
     1 |      4955 | draw
     1 |      2138 | draw
     1 |      1586 | draw
     1 |      2817 | win
     1 |      1231 | draw
     1 |      3940 | draw
     1 |      3449 | win
     1 |       119 | win
     1 |      2658 | win
     1 |       715 | win
     1 |      4286 | draw
     1 |      4074 | win
     1 |      1513 | win
     1 |       542 | win
     1 |      1341 | draw
     1 |      4972 | draw
     1 |      3833 | draw
     1 |      4024 | win
     1 |      1298 | draw
     1 |      3634 | win
     1 |       524 | draw
     1 |      3653 | win
     1 |      3068 | win
     1 |      4538 | draw
     1 |      4083 | draw
     1 |      1950 | draw
     1 |      3596 | win
```

Building on the previous query, the third query gives counts of each result

```
select count(c.games) as count, c.result from (select count(game_id) as games, player_id, result from game group by player_id, result having count(game_id)=1) c group by result;
```

```
aashrayyadav=# select count(c.games) as count, c.result from (select count(game_id) as games, player_id, result from game group by
 count | result
-------+--------
  1797 | draw
  1749 | win
(2 rows)

aashrayyadav=#
```

We have **1797 draws** and **1749 wins**. The email campaign should try to target to bring these players back to the contest.

Since they have won or drawn, we can either send a generic message to all players congratulating them on their past performance and asking them to return.

Another approach could be design two templates for players who drew and won.

For players who **won** -> ask them to continue their winning streak and improve their ranks, level up

For players who **drew** -> you got so close to winning. Come back and claim your first victory at 9dt!

Note:

- For Q3, we can further classify these values across genders which is why I have stored it in the player table, thereby resulting in more targeted messaging.
- I have used SQL subqueries over joins to increase the readability of the queries
- You will be able to locate the source code, processed csv files and screenshots in the submission