FIFA-19 Data Analysis

Kaumudi Moholkar
Computer Science and Engineering
University at Buffalo
Buffalo, USA
kaumudim@buffalo.edu

Priyanka Dalit Tajane
Computer Science and Engineering
University at Buffalo
Buffalo, USA
pdtajane@buffalo.edu

***Abstract*—**This document is based on the analysis of the FIFA-19 database. FIFA-19 is a soccer video game developed by Electronic Arts and played on platforms like PC, PlayStation, Xbox. It has multiple game modes, both single and online player. To provide the users with a realistic and immersive game experience, it was created with real world players and their skills similar to the players as much as possible. The players have different skill sets along with the positions they are usually preferred by. The data acquired for FIFA-19 was around 18,000 players and about 89 attributes of each.

I. Introduction

FIFA-19 is a popular soccer simulator video game series. The main aim of this project is to gain the insights of the data like how good they are in the game, their skills, their positions in the game, ratings based on overall and international. This project is to analyze based on players' attributes, if there are any categories based on which the players can be defined by either a certain class or any similarities. The dataset consists of 89 attributes, out of which some have no importance like the photo of the flag or the real face, in all the images are of little importance. There are many valuable attributes which help segregate the data into relatable databases.

This analysis will help the team captain/skipper get a detailed picture of the combination of players which can increase their relative success factor as well as the clubs who wish to buy/draft them. Through broad football experience: the bits of knowledge gave our outcomes, along with understanding, and contextualized data empowers clients to act sagaciously when playing FIFA, picking a superior group for say Fantasy Premier association, or increment their wagering chances.

II. Background

The data is collected from Kaggle, the largest resource for analysis of data available on a public platform. The original dataset has been filtered in keeping the main aim of the project aligned by previous data scientists and analysts. For each attribute, we have a number from 0 to 100 that shows how great a player is at that quality. The features are either used for exploratory data analysis and data modelling. So, the required attributes have been sorted out based on our requirement and to answer many queries.

The tables previously used had countries and team details along with the below listed tables. However, the countries were not clearly justified as they were the nationalities of players. And thus, no team details could have been formed.

III. ER Diagram

The ER diagram is designed on 5 tables. Each player will have some set of skills and different kinds of positions while playing their game. Players will have 3 kinds of ratings, overall, potential and international.

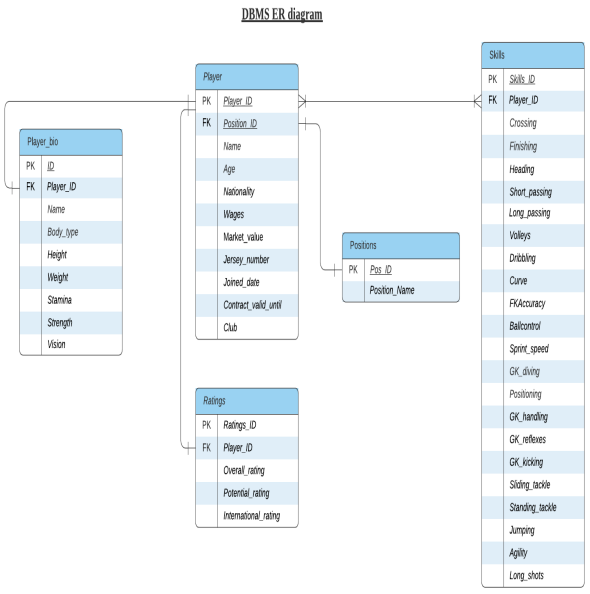

Fig.1 ER Diagram

IV. METHODS

The tables were partitioned based on different attributes they possess. The players had various features like age, nationality, club, wages to name a few. These can be formed into one table which describes the basic details of the player. Another table was formed based on their physical attributes like height, weight, stamina, strength, vision, and so on. This was the player bio table.

The positions of each player were formulated in another table as the positions table. This describes their roles and responsibilities in the team. The positions can be grouped as striker, forward, and winger. The strikers are positioned in front of forwards and wingers and close to the opposing goal. Their responsibility is to attack and score goals. So their skills like ball control, shooting, and finishing are expected to be good. Forwards are positioned behind the strikers. They are expected to receive the ball and score by passing to others or goal. They are expected to have all the skills of a striker along with passing. Wingers are positioned at the boundary to pass and create chances for strikers and forwards. They should possess skills like dribbling, passing, crossing, and acceleration. So we have tables for positions and skills. Based on the two tables many queries can be answered.

The CSV file obtained from Kaggle has been initially uploaded in the pgAdmin in the PostgreSQL server. A new database was created with the name FIFA-19. A table was created with the name fifa_details, which consists of all the 89 columns of the large CSV data. The values are inserted using the import feature available in the pgAdmin. Once we have the entire CSV handy in the database, we can write a script to just insert values from this table into multiple tables created according to the discussion above.

The first table of positions is created with primary key constraint as position id which is serially incremented as the tuples are inserted. These contain positions like striker, winger, goalkeeper, forward, mid-fielder to name a few.

	pos_id [PK] integer	position_name character varying
1	1	Left_striker
2	2	Right_striker
3	3	Striker
4	4	Left_winger
5	5	Left_forward
6	6	Center_forward
7	7	Right_forward
8	8	Right_winger
9	9	Center_attacking_midfielder
10	10	Left_attacking_midfielder

Fig 2. Positions Table

The next table consists of players' information, like their name, age, wages, nationality, market value, jersey number, and so on. These have a primary key constraint as player id which is serially incremented. This table contains a foreign key position id, which shows which player is positioned where on the soccer ground.

Query to create the players' table:

```
drop table if exists Player;
CREATE TABLE IF NOT EXISTS Player
(
id integer,
Player_ID serial NOT NULL,
Name varchar,
Age integer,
Nationality varchar,
Wages varchar,
Market_value varchar,
Jersey_number integer,
Joined_date date,
Contract_valid_until integer,
position_id integer,
CONSTRAINT Player_ID PRIMARY KEY (Player_ID),
CONSTRAINT position_id FOREIGN KEY (position_id)
REFERENCES Positions (pos_id)
);
```

Query to insert values into the players' table:

```
INSERT INTO player (name, age, nationality, wages, market_value, jersey_number, joined_date, contract_valid_until, position_id)

   SELECT f.name, f.age, f.nationality, f.wage, f.value_s, f.jersey_number, f.joined, f.contract_valid_until, PositionNumber(1,27)
FROM fifa_details f;
```

id integer	player_id [PK] integer	name character varying	age integer	nationality character varying	wages character varying	market_value character varying	jersey_number integer	club character varying	joined_date date	contract_valid_until integer	position_id integer
158023	1	L. Messi	31	Argentina	€565K	€110.5M	10	FC Barcelona	2004-07-01	2021	9
20801	2	Cristiano Ronaldo	33	Portugal	€405K	€77M	7	Juventus	2018-07-10	2022	5
190871	3	Neymar Jr	26	Brazil	€290K	€118.5M	10	Paris Saint-Germain	2017-08-03	2022	15
193080	4	De Gea	27	Spain	€260K	€72M	1	Manchester United	2011-07-01	2020	12
192985	5	K. De Bruyne	27	Belgium	€355K	€102M	7	Manchester City	2015-08-30	2023	20
183277	6	E. Hazard	27	Belgium	€340K	€93M	10	Chelsea	2012-07-01	2020	18
177003	7	L. Modrić	32	Croatia	€420K	€67M	10	Real Madrid	2012-08-01	2020	6
176580	8	L. Suárez	31	Uruguay	€455K	€80M	9	FC Barcelona	2014-07-11	2021	16
155862	9	Sergio Ramos	32	Spain	€380K	€51M	15	Real Madrid	2005-08-01	2020	9
200389	10	J. Oblak	25	Slovenia	€94K	€68M	1	Atlético Madrid	2014-07-16	2021	10

Fig 3. Player Table

Another table is created for the player's physical details like their weight, height, stamina, strength, vision to name a few. These are required to understand their strengths and weaknesses and judge how they might seem to be fit for a particular position.

```
INSERT INTO player_bio(
        player_id, name, body_type, height, weight, stamina, strength, vision)
    SELECT Distinct p.player_id, f.name, f.body_type, f.height, f.weight, f.stamina,
                f.strength, f.vision
                FROM player p
                INNER JOIN fifa_details f
                ON p.id = f.id
;
```

id [PK] integer	player_id integer	name character varying	body_type character varying	height character varying	weight character varying	stamina integer	strength integer	vision integer
1	4962	C. Deac	Normal	5'10	163lbs	84	70	69
2	395	B. Höwedes	Normal	6'2	170lbs	61	78	57
3	11086	F. Soyalp	Lean	5'9	161lbs	65	61	66
4	17047	K. Wimmer	Normal	6'2	187lbs	65	79	59
5	16834	E. Bilen	Lean	6'1	176lbs	21	61	29
6	6426	T. Guidara	Lean	5'8	150lbs	78	59	40
7	152	K. Schmeichel	Stocky	6'2	196lbs	34	64	59
8	2937	L. Gamba	Normal	5'7	154lbs	68	54	78
9	6258	A. Turgeman	Normal	5'10	163lbs	80	76	46
10	12660	V. Slivka	Lean	6'3	185lbs	55	68	58

Fig 4. Player-Bio Table

There is a table for ratings, which consists of overall rating, potential rating, and international reputation. The ratings of players are necessary for the formation of teams and choosing them in terms of their market value. The foreign key constraint is the player id which is taken from the players' table. The primary key is the rating id.

ratings_id [PK] integer	player_id integer	overall_rating integer	potential_rating integer	international_rating integer
100	1	94	94	5
101	2	94	94	5
102	3	92	93	5
103	4	91	93	4
104	5	91	92	4
105	6	91	91	4
106	7	91	91	4
107	8	91	91	5
108	9	91	91	4
109	10	90	93	3

Fig 5. Rating Table

The table of skills is created to check the skill set of each player, in terms of their position and majorly required for them. The skills include ball control, positioning, sprint speed, goalkeeper reflexes, kicking to name a few. The foreign key constraint is the player id which is taken from the players' table. The primary key is the skill id. Every player must possess a set of skills for a certain position. Then they can achieve a better rating and play to the fullest.

s_id [PK] i	player_i integer	crossir integer	finishing integer	heading integer	short_pa integer	long_pa integer	volleys integer	dribbling integer	curve integer	fkaccur integer	ballco integer	sprin integ	gk_d integ	positio integer	gk_ha integer	gk_refl integer	gk_k integ	slidi integ	standing integer
100	1	84	95	70	90	87	86	97	93	94	96	86	6	94	11	8	15	26	28
101	2	84	94	89	81	77	87	88	81	76	94	91	7	95	11	11	15	23	31
102	3	79	87	62	84	78	84	96	88	87	95	90	9	89	9	11	15	33	24
103	4	17	13	21	50	51	13	18	21	19	42	58	90	12	85	94	87	13	21
104	5	93	82	55	92	91	82	86	85	83	91	76	15	87	13	13	5	51	58
105	6	81	84	61	89	83	80	95	83	79	94	88	11	87	12	8	6	22	27
106	7	86	72	55	93	88	76	90	85	78	93	72	13	79	9	9	7	73	76
107	8	77	93	77	82	64	88	87	86	84	90	75	27	92	25	37	31	38	45
108	9	66	60	91	78	77	66	63	74	72	84	75	11	60	8	11	9	91	92
109	10	13	11	15	29	26	13	12	13	14	16	60	86	11	92	89	78	18	12

Fig 6. Skills Table

The tables are created and then the preprocessing of it is done, so the data is synchronized with all the data types and proper formatting. The updation of the column contract is valid until is done as the date is not in a synchronized format. The column is updated to only years. The data types are updated for smooth insertions. A load file is created to insert all the data, from the large table consisting of all the CSV data. Once all insertion is done the tables have around 18000 tuples in almost every table except the positions. Some of the tables have null values as some features of the player are not available in the dataset taken into consideration.

Query to update the contract valid until field:

```
UPDATE fifa_details
SET contract_valid_until = CONCAT('20',
RIGHT(contract_valid_until,2))::INTEGER
where contract_valid_until LIKE '%-%';
```

Query to alter the fifa_details table:

```
ALTER TABLE fifa_details
   ALTER COLUMN contract_valid_until TYPE
integer USING contract_valid_until::integer;
```

Query to alter the players' table:

```
ALTER TABLE player
   ALTER COLUMN contract_valid_until TYPE
integer USING contract_valid_until::integer;
```

The above is the preprocessing of the data before doing any insertions, avoiding discrepancies in data type.

Functions are created for creating random value insertion of position id into the players' table, for assigning the different positions to the table. This avoids the tedious work of allotting a position to the player manually.

Query to create function for assigning random values to position:

```
CREATE FUNCTION PositionNumber (low
integer, high integer)
RETURNS integer AS $$
BEGIN
        RETURN floor(random()*(high-low+1) +
low);
END;
$$ language 'plpgsql' STRICT;
```

V. FUNCTIONAL DEPENDENCIES

The tables created are in BCNF. Each table has a primary key and no repeating columns, thus not violating 1NF. According to 2NF, every column has a relation to the primary key i.e, each column is dependent on the primary key. There is no partial dependency, thus not violating 2NF. For example in the player table, each column is dependent on the player_id.

- player_id ---> name
- player_id ---> age
- player_id ---> nationality
- player_id ---> market_value
- player_id ---> club

The tables are in 3NF as there is no transitive dependency. For example in the player_bio table, each column is dependent on the player_id.

- player_id ---> height
- player_id ---> weight
- player_id ---> stamina
- player_id ---> strength
- player_id ---> vision

Finally, the table is in BCNF as non-prime attributes are not dependent on prime attributes

VI. QUERY EXECUTION ANALYSIS

To check the performance of any PostgreSQL query, we need to execute the query execution plan. This helps in providing an entire summary of the performance and builds a detailed report of time taken to execute each step and the cost incurred to finish it. There is a keyword to show the report, EXPLAIN. It can also be done using the button provided in pgAdmin along with multiple options like Analyze, verbose, costs, buffers to name a few. The EXPLAIN keyword syntax is

```
EXPLAIN [ ( OPTION [, ...] ) ]
YOUR_SQL_QUERY;
```

When the EXPLAIN query is executed, it provides the entire statistics including the total time taken for every plan node, number of rows are fetched.

As an example to find the best goalkeeper based on some pre set conditions we have filtered down to skills like GKReflexes, GKDiving, Positioning, GKHandling, GKKicking, Sprint Speed from the skills table, and the position is chosen from the player table. The GK was searched in the player table based on the position id and the average of the skills is calculated. If the players' skills are greater than the average value, the contract valid is less than 2021 and the market value should be around 10M. This query is executed and a series of tuples are found with the defined constraints.

To analyze the statistics of the query, EXPLAIN keyword is executed along with the query. The cost estimated was 965.7 and the rows fetched are 33000. To improve the performance of the cost indexing was performed. Indexes were created on the players' table and the positions table. The indexing was done on the player id and the position id of the player table. It was also done on the position id of the position table. The cost was drastically decreased to 761.73 and the rows fetched were 5500.

V. ACKNOWLEDGMENT

This project has required a significant amount of work on my part. That would not have been possible without Prof. Sreyasee Das Bhattacharjee encouragement and assistance, as well as the wonderful TAs (Keyan Yu and Yunnan Yu). I'd like to express my heartfelt gratitude to each and every one of them. I am deeply indebted to Professors for their constant guidance and supervision as well as for providing the necessary information regarding the project.

VI. REFERENCES

[1] Dataset link:
https://www.kaggle.com/karangadiya/fifa19

[2]https://www.sqlservercentral.com/articles/getting-a-query-execution-plan-in-postgresql

[3]https://towardsdatascience.com/can-i-form-a-dream-team-in-fifa-19-using-sql-160fd9a3f172

[4] Table structure screenshots:
https://drive.google.com/drive/folders/1sNvtDBZBHXfg2v3kCKm5QQPYeb4JQ05Q?usp=sharin