POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Systems and Methods for Big and Unstructured Data Project

Author(s): **Federico Valsecchi 274083**
Francesco Seracini 277999

Group Number: **109**

Academic Year: 2024-2025

Contents

1 | Introduction

The world of sports has always been intertwined with the advancement of technology, constantly seeking innovative ways to enhance training, gameplay, and analysis. In this regard, the United States has consistently been at the forefront, providing fans with comprehensive data and insights into every facet of the game. For this project, we have chosen to focus on the National Basketball Association (NBA) due to our shared passion for basketball and our belief in the significance of statistics in driving teams' daily improvements. Our project builds upon this data-centric approach by leveraging a comprehensive dataset that captures the multidimensional nature of basketball, including competition history and player and club statistics. We opted for MongoDB as our database solution because its non-relational structure is ideally suited to handle the diversity and volume of the data we are working with. Unlike traditional relational databases, MongoDB's flexible data model enables us to store and process various types of data without the need for a predefined schema. As a document-oriented database, MongoDB allows each game to be represented as a document with a rich and dynamic set of attributes. Furthermore, MongoDB's horizontal scalability through automatic sharding is essential for managing the large volumes of data generated in modern basketball. The performance of this technology remains optimal even as the data size increases. Our objective is to uncover meaningful patterns and insights while highlighting intriguing aspects of the game's evolution. The dataset includes detailed information on games, including results, club performance, and individual player statistics.

2 | DATASET

2.1. Dataset

The selected dataset is an extensive collection of NBA data, primarily compiled by Valeri Karpov, a software engineer who scraped data from "Basketball-Reference.com" and converted it into a MongoDB-friendly format. The dataset spans nearly 30 years of NBA regular season games, from the 1985-1986 season to the 2013-2014 season. During this period, there were 31,686 games, with each game represented as an individual document. Each document contains information on the game's result, the two teams that faced each other, and comprehensive statistics for both the teams and each player on both teams. The original dataset can be downloaded using the following link: http://bit.ly/1gAatZK.

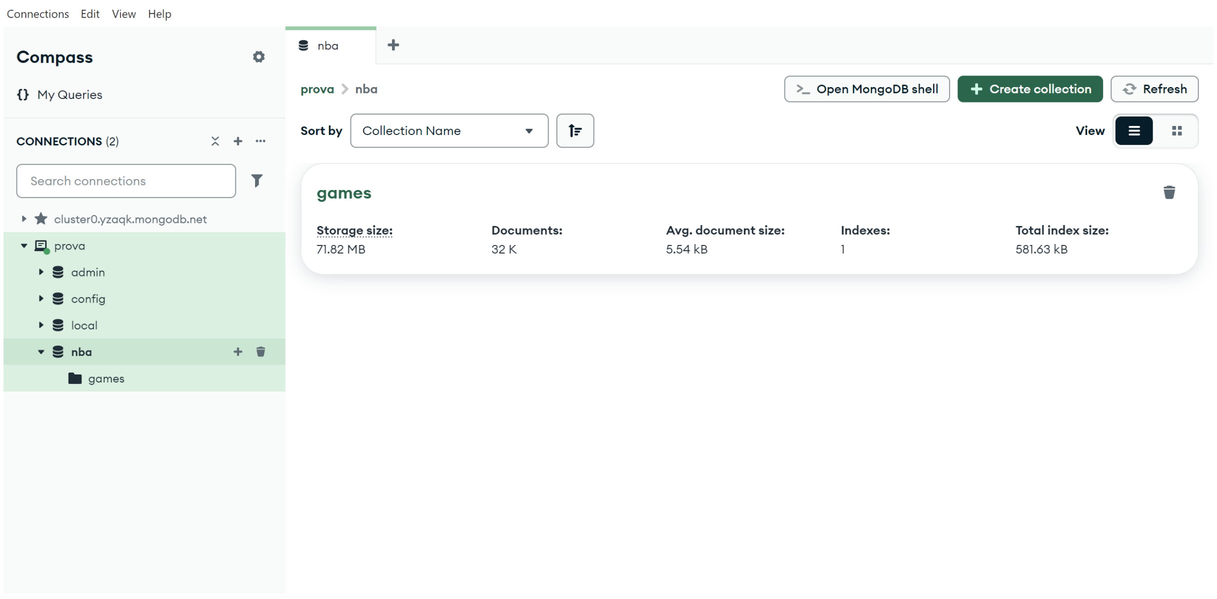
The downloaded folder contains two files :

• 'games.bson': Contains the dataset, which needs to be translated into a JSON format for use with MongoDB.

• 'games.metadata.json': Contains the necessary index to index the dataset after converting it to a JSON format.

The MongoDB database, named "nba," has a collection with the following statistics:

• 'games' collection: 31,686 documents, with an average size of 5.54 kB per document. The total size of indexes is 581.63 kB.



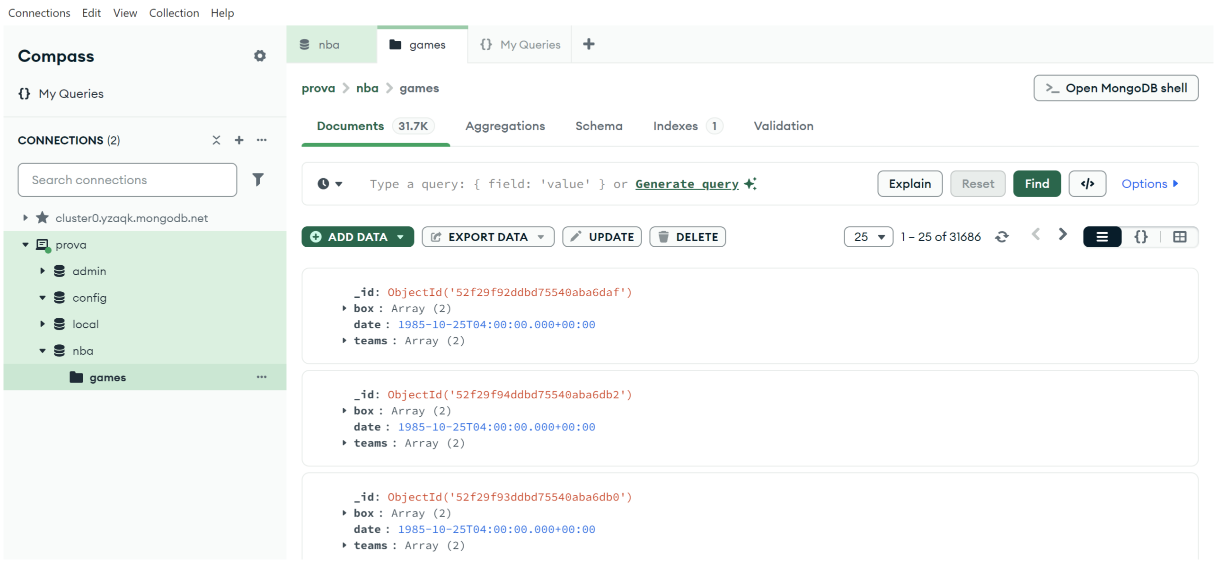
games

Storage size:	Documents:	Avg. document size:	Indexes:	Total index size:
71.82 MB	32 K	5.54 kB	1	581.63 kB

Figure 2.1: NBA Dataset.

2.2. Collections

2.2.1. Games



```
_id: ObjectId('52f29f92ddbd75540aba6daf')
▸ box : Array (2)
date : 1985-10-25T04:00:00.000+00:00
▸ teams : Array (2)

_id: ObjectId('52f29f94ddbd75540aba6db2')
▸ box : Array (2)
date : 1985-10-25T04:00:00.000+00:00
▸ teams : Array (2)

_id: ObjectId('52f29f93ddbd75540aba6db0')
▸ box : Array (2)
date : 1985-10-25T04:00:00.000+00:00
▸ teams : Array (2)
```

Figure 2.2: Games collection.

attribute	**type**	**description**
id	ObjectId	A unique identifier for the document, which also serves as a unique identifier for the game
box	Array (of Objects)	An array consisting of two objects, each representing one of the teams involved in the game. These objects contain statistics for both the team and its individual players
date	Date	Date and time when the game was played
teams	Array (of Objects)	An array containing information about the two teams that participated in the game

Table 2.1: Structure of 'Games'

attribute	**type**	**description**
players	Array (of Objects)	An array containing objects, each representing a player from the team. These objects hold detailed information and statistics for each player in the game
team	Object	An object that contains a comprehensive set of statistics and data representing the team's overall performance in the game
won	Int32	A value indicating the team's result in the game. It is 1 if the team won and 0 if they lost

Table 2.2: Structure of 'Box' inside of 'Games'

attribute	**type**	**description**
ast	Int32 (of Objects)	Number of assists made by the player
blk	Int32	Number of blocks made by the player
drb	Int32	Number of defensive rebounds made by the player
fg	Int32	Number of field goals made by the player
fg3	Int32	Number of three-point field goals made by the player
fg3-pct	String	Three-point field goals percentage
fg3a	Int32	Three-point field goals attemps
fg-pct	String	Field goals percentage
fga	Int32	Field goals attemps
ft	Int32	Free throws made by the player
ft-pct	String	Free throws percentage
fta	Int32	Free throws attemps
mp	String	Minutes played by the player
orb	Int32	Offensive rebounds made by the player
pf	Int32	Number of personal fouls
player	String	Name of the player
pts	Int32	Total points made by the player
stl	Int32	Steals made by the player
tov	Int32	Turnovers made by the player
trb	Int32	Total rebounds made by the player

Table 2.3: Structure of 'Players' inside 'Box' inside of 'Games'

attribute	**type**	**description**
ast	Int32 (of Objects)	Number of assists made by the team
blk	Int32	Number of blocks made by the team
drb	Int32	Number of defensive rebounds made by the team
fg	Int32	Number of field goals made by the team
fg3	Int32	Number of three-point field goals made by the team
fg3-pct	String	Three-point field goals percentage
fg3a	Int32	Three-point field goals attemps
fg-pct	String	Field goals percentage
fga	Int32	Field goals attemps
ft	Int32	Free throws made by the team
ft-pct	String	Free throws percentage
fta	Int32	Free throws attemps
mp	String	Minutes played by the team
orb	Int32	Offensive rebounds made by the team
tf	Int32	Number of team fouls
pts	Int32	Total points made by the team
stl	Int32	Steals made by the team
tov	Int32	Turnovers made by the team
trb	Int32	Total rebounds made by the team

Table 2.4: Structure of 'Team' inside 'Box' inside of 'Games'

attribute	**type**	**description**
name	String	Name of the team
abbreviation	String	Abbreviation of the name of the team
score	Int32	Points made by the team in that game
home	Boolean	Equal to 1 if the team plays at home, 0 otherwise
won	Int32	1 if team won, 0 otherwise

Table 2.5: Structure of 'Box' inside of 'Games'

3 | QUERIES

We have divided the queries into three sections:

- 'Teams queries': For visualizing patterns and statistics of clubs.
- 'Games queries': For examining statistics related to games.
- 'Players queries': For showcasing results on individual player statistics.

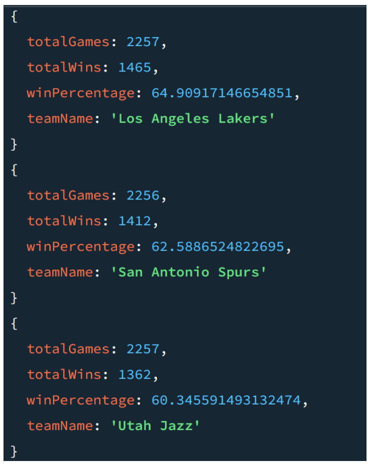
3.1. Teams queries

While individual talent is important in a sport like basketball, it is ultimately a team game, and relying on just one player to win a game is not enough. In basketball, having a well-balanced team, both offensively and defensively, is crucial. Collaboration between teammates plays a key role in achieving success on the court.

3.1.1. Teams with the most victories

To begin our analysis, we sought to identify the teams with the highest winning percentages throughout the dataset's timespan. In basketball, as in any other sport, the ultimate goal is to win more games than the competition. Therefore, we wanted to determine which teams consistently emerged victorious over the course of 30 years. We opted to focus on winning percentages rather than the absolute number of wins for a crucial reason: during the period under consideration, some teams were disbanded while others were newly formed. This disparity resulted in teams playing a varying number of games, making a direct comparison of total wins less meaningful. By calculating the percentage of games won, we can accurately assess each team's success rate, regardless of the number of games they played. This approach provides a more equitable basis for comparing the performance of teams across different eras and circumstances. The query uses unwind to split the teams array, then group to calculate total games and wins for each team. addFields computes the winPercentage, avoiding division by zero. It then sorts teams by winPercentage with sort, limits the results to the top project to display only the relevant fields.

```
db. games. aggregate([
    { $unwind: "$teams" },
    {$group: {_id: "$teams. name", totalGames: { $sum: 1},
        totalwins: { $sum: "$teams.won" } } },
    {
        $addFields: {
            winPercentage: {
                $cond: [
                    { $gt: ["$totalGames", 0] },
                    { $multiply: [{ $divide: ["$totalwins", "$totalGames"] }, 100] },
                    0
                ]
            }
        }
    },
    { $sort: { winPercentage: -1 } },
    { $limit: 5 },
    { $project: {_id: 0, teamName: "$_id", totalGames: 1, totalwins: 1,
    winPercentage: 1} }
]);
```



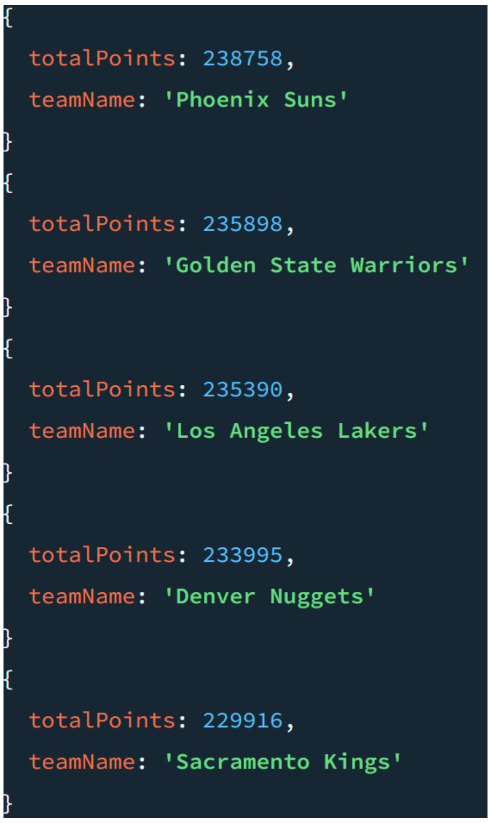
```
{
  totalGames: 2257,
  totalWins: 1465,
  winPercentage: 64.90917146654851,
  teamName: 'Los Angeles Lakers'
}
{
  totalGames: 2256,
  totalWins: 1412,
  winPercentage: 62.5886524822695,
  teamName: 'San Antonio Spurs'
}
{
  totalGames: 2257,
  totalWins: 1362,
  winPercentage: 60.345591493132474,
  teamName: 'Utah Jazz'
}
```

Figure 3.1: Result

3.1.2. Teams with the most Points Scored

After analyzing the teams with the highest win percentages, it's valuable to examine the teams with the most points scored to draw meaningful comparisons between scoring ability and overall success rate. This query unwinds the teams array to separate each team in the games and then groups by the team name to sum the total points scored by each team. The results are sorted in descending order based on the total points, and only the top 5 teams are returned. Finally, the query projects the team name and their total points.

```
db. games. aggregate ([
    { $unwind: "$teams" },
    { $group: [_id: "$teams.name", totalPoints:
        { $sum: "$teams.score" } } },
    { $sort: { totalPoints: -1 } },
    { $limit: 5 },
    { $project: {_id: 0, teamName: "$_id", totalPoints: 1 } }
]);
```



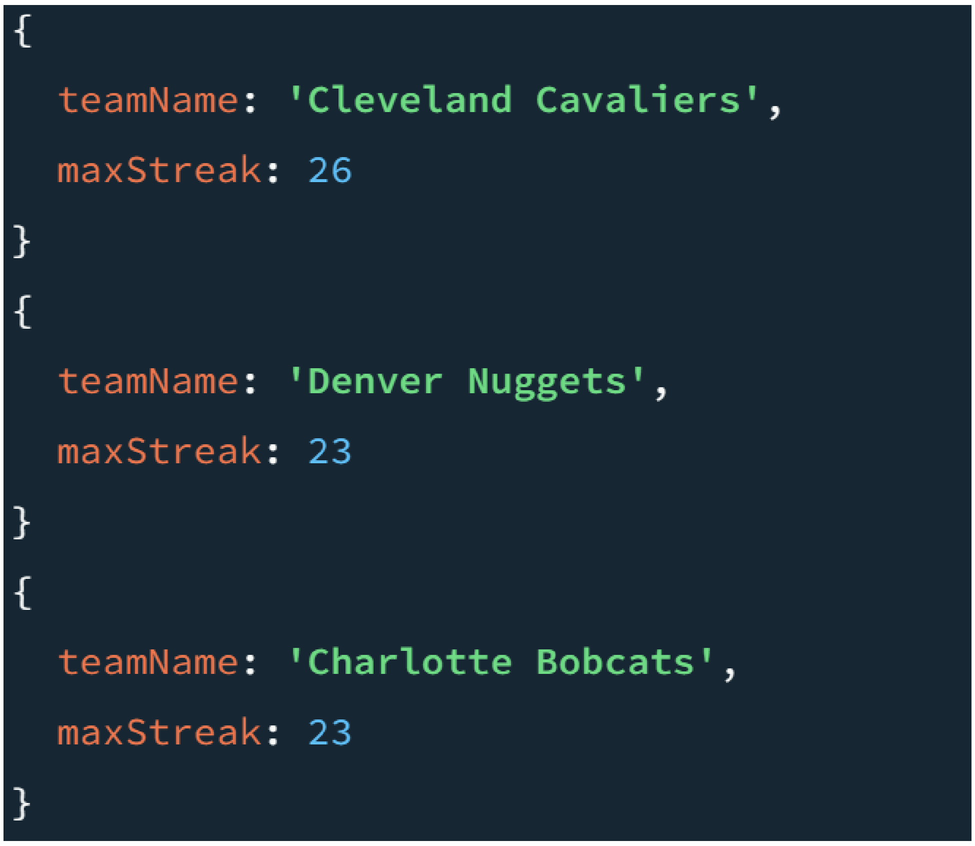
```
{
  totalPoints: 238758,
  teamName: 'Phoenix Suns'
}
{
  totalPoints: 235898,
  teamName: 'Golden State Warriors'
}
{
  totalPoints: 235390,
  teamName: 'Los Angeles Lakers'
}
{
  totalPoints: 233995,
  teamName: 'Denver Nuggets'
}
{
  totalPoints: 229916,
  teamName: 'Sacramento Kings'
}
```

Figure 3.2: Result

3.1.3. Teams with the longest loosing streak

After checking out the teams with the best winning percentages over the 30 years, we were curious to see the flip side of the coin. We wanted to find out which team had the longest losing streak in the dataset. By figuring out who this team was and how long they went without a win, we hoped to get a better idea of the tough times teams can go through. This would give us a more complete picture of the ups and downs NBA teams have faced over the years, shining a light not just on the most successful franchises, but also on those that have gone through some serious rough patches. The query splits teams, sorts games by date, groups data by team, and calculates the longest losing streak using reduce. It then sorts teams by the longest streak, limits the result to the top 3, and shows the team name and streak length.

```
nba> db. games. aggregate (L
    { $unwind: "$teams" },
    { $sort: { date: 1} },
    { $group: {_id: "$teams. name", games: { $push:
        { won: "$teams.won", date: "$date" } } } },
    {
        $addFields: {
            streaks: {
                $reduce: {
                    input: "$games",
                    initialValue: { currentStreak: 0,
                        maxStreak: 0 },
                    in: {
                        currentStreak: { $cond: [{ $eq:
                            ["$$this.won", 0] }, { $add:
                            ["$$value.currentStreak", 1] }, 0] },
                            maxStreak: { $max: ["$$value-maxStreak",
                            "$$value.currentStreak"] }
                    }
                }
            }
        }
    }
    { $sort: { "streaks.maxStreak": -1 } },
    { $limit: 3 },
        { $project: {_id: 0, teamName: "$_id",
        maxstreak: "$streaks.maxstreak" } }
    ]);
```



```
{
  teamName: 'Cleveland Cavaliers',
  maxStreak: 26
}
{
  teamName: 'Denver Nuggets',
  maxStreak: 23
}
{
  teamName: 'Charlotte Bobcats',
  maxStreak: 23
}
```

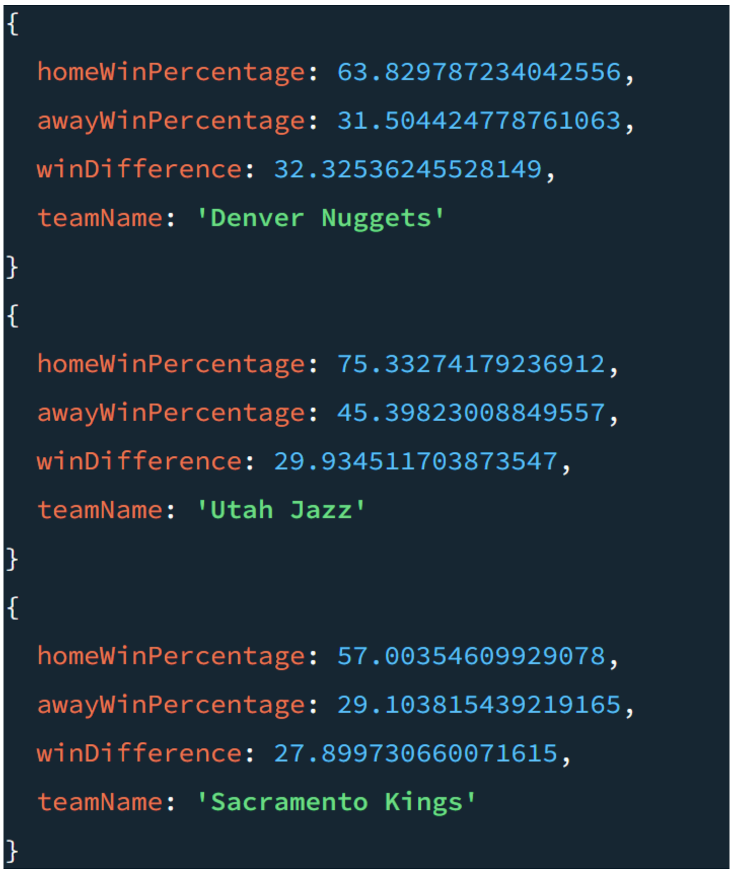
Figure 3.3: Result

3.1.4. Difference between home and away percentage wins

We decided to calculate the difference between home and away winning percentages to see if playing at home provides a significant advantage for NBA teams. This query helps us understand the impact of home court advantage on team performance. This query calculates the home and away win percentages for each team, ensuring a 0 percent if no games are played in either category. It calculates the difference between these percentages, projects the home and away win percentages along with the difference, and sorts the results in descending order by win difference.

```
    nba > db. games. aggregate ([
        { $unwind: "$teams" },
        {
```

```
        $group: {
            _id: "$teams. name",
            homeGames: { $sum: { $cond: [{ $eq: ["$teams-home",
                true] }, 1, 0] } }, homeWins: { $sum: { $cond:
                [{ $eq: ["$teams.home", true] }, "$teams-won", 0]
                } }, awayGames: { $sum: { $cond: [{ $eq:
                ["$teams.home", false]}, 1, 0] } },
                awayWins: { $sum: {} $cond: [{ $eq: ["$teams.home",
                false] }, "$teams-won", 0] } }
        }
    },
    {
        $addFields: {
            homeWinPercentage: {
                $cond: [
                    { $eq: ["$homeGames", 0] },
                    0,
                    { $multiply: [{ $divide: ["$homeWins",
                    "$homeGames"] }, 100] }
                ]
            },
            awayWinPercentage: {
                $cond: [
                    { Seq: ["$awayGames", 0] },
                    0,
                    { $multiply: [{ $divide: ["$awayWins",
                    "$awayGames"] }, 100] }
                ]
            }
        }
    },
    { $addFields: { winDifference: { $subtract:
        ["$homeWinPercentage", "$awayWinPercentage"] } } },
    {
        $project: {
            _id: 0,
            teamName: "$_id",
                homeWinPercentage: 1,
                awayWinPercentage: 1,
                winDifference: 1
            }
        },
        { $sort: { winDifference: -1 } }
    ]) ;
```



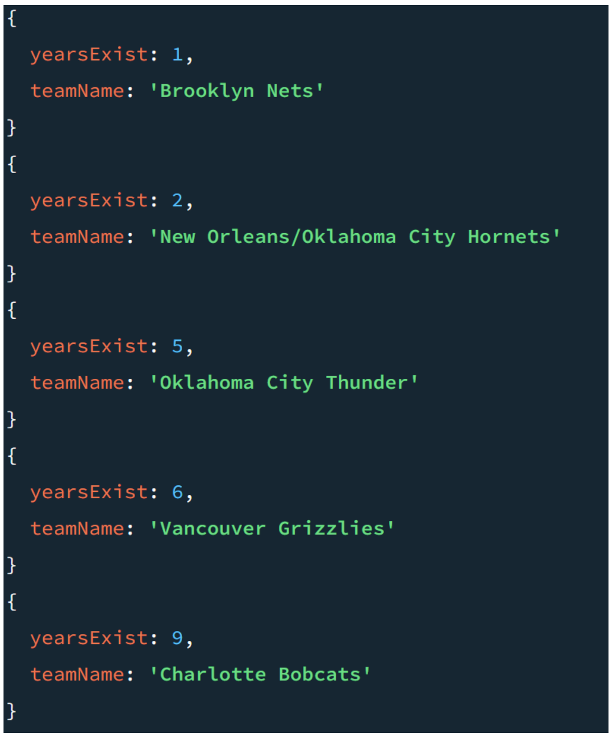
```
{
  homeWinPercentage: 63.829787234042556,
  awayWinPercentage: 31.504424778761063,
  winDifference: 32.32536245528149,
  teamName: 'Denver Nuggets'
}
{
  homeWinPercentage: 75.33274179236912,
  awayWinPercentage: 45.39823008849557,
  winDifference: 29.934511703873547,
  teamName: 'Utah Jazz'
}
{
  homeWinPercentage: 57.00354609929078,
  awayWinPercentage: 29.103815439219165,
  winDifference: 27.899730660071615,
  teamName: 'Sacramento Kings'
}
```

Figure 3.4: Result

3.1.5. Season played by each team

Examining the longevity of teams in the league provides valuable historical context, distinguishing between long-established franchises and more recent additions. This analysis helps understand the historical development of the league and each team's legacy within it. This query first unwinds the teams array to process each team separately. It then groups by team name, calculating the first and last year of each team's participation in games by extracting the year from the date field using the year operator. After that, it adds a new field, yearsExist, which represents the number of years the team has existed by subtracting the first year from the last year. The query then sorts the teams in ascending order based on their yearsExist value and finally projects the team name and the number of years they've been active. This allows you to see how long each team has existed, ordered from the least to the most years.

```
db. games. aggregate ([
    { $unwind: "$teams" } ,
    { $group: { _id: "$teams.name", firstYear: { $min:
        { $year: "$date" } }, lastYear: {$max: { $year:
        "$date"} } } },
    { $addFields: { YearsExist: { $subtract: ["$lastYear,
        "$firstYear"]b} } } },
    { $sort: { yearsExist: 1 } },
    { $project: {_id: 0, teamName: "$_id", yearsExist: 1 } }
]};
```



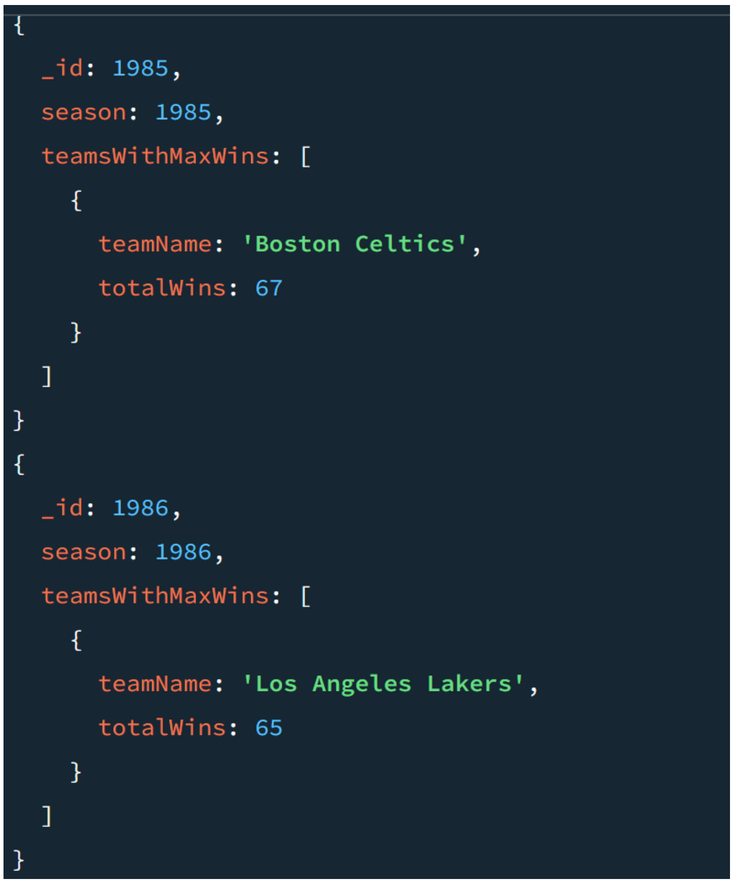
```
{
  yearsExist: 1,
  teamName: 'Brooklyn Nets'
}
{
  yearsExist: 2,
  teamName: 'New Orleans/Oklahoma City Hornets'
}
{
  yearsExist: 5,
  teamName: 'Oklahoma City Thunder'
}
{
  yearsExist: 6,
  teamName: 'Vancouver Grizzlies'
}
{
  yearsExist: 9,
  teamName: 'Charlotte Bobcats'
}
```

Figure 3.5: Result

3.1.6. Annual win leader analysis

For this analysis, we define an NBA season as running from October 1st through September 30th of the following year. We'll examine which teams achieved the highest number of regular season wins for each complete season, providing insight into year-by-year competitive dominance. First, this query determines the correct season for each game based on its date. If the game occurred in October or later, it is considered part of the current season; otherwise, it belongs to the previous season. Next, the query unwinds the teams array so that each team in each game can be processed individually. It then groups the data by season and team, counting the number of wins each team had. Afterward, it identifies the teams with the most wins for each season by finding the maximum number of wins and filtering out the teams that reached that number. Finally, the results are sorted by season in ascending order, giving the team with the most wins for each season.

```
db. games. aggregate([
    {
        $addFields: {
            seasonYear: {
                $cond: [
                    $gte: [ $month: "$date" }, 10] },
                    // If the month is October or later
                    { $year: "$date" 3, // Season in the same year
                    { $subtract: [{ $year: "$date" }, 1] }
                    // Season in the previous year
                ]
            }
        }
    },
    { $unwind: "$teams" },
    {
        $group: {
            _id: { season: "$seasonYear", team: "$teams name" },
            totalWins: { $sum: { $cond: [{ $eq: ["$teams.Won", 1]
                }, 1, 0] } }
        }
    },
    {
        $group: {
            _id: "$_id.season",
            teamsWithMaxWins: {
                $push: { teamName: "$_id. team",
                totalwins: "$totalWins" }
            }
        }
    },
    {
        $project: {
            season: "$_id",
            teamsWithMaxWins: {
                $let: {
                    vars: {
                        maxwins: { Smax: { $map: { input:
                        "$teamsWithMaxWins", as: "team", in: "$$team.
                        totalWins" } } }
                    },
                    in: {
                        $filter: {
                            input: "$teamsWithMaxWins",
                        }
                    }
                }
            }
            season: "$_id",
            teamsWithMaxWins: {
                $let: {
                    vars: {
                        maxWins: { $max: { $map:
                        { input: "$teamsWithMaxWins"
                        as: "team", in:
                        "$$team. totalWins" } } }
                    },
                    in: {
                        $filter: {
                            input: "$teamsWithMaxWins", as: "team",
                            cond: { $eq: ["$$team. totalWins",
                            "$$maxWins"] }
                        }
                    }
                    }
                }
            }
        },
        {
            $sort: { season: 1 }
        }
    ]) ;
```



```
{
  _id: 1985,
  season: 1985,
  teamsWithMaxWins: [
    {
      teamName: 'Boston Celtics',
      totalWins: 67
    }
  ]
}
{
  _id: 1986,
  season: 1986,
  teamsWithMaxWins: [
    {
      teamName: 'Los Angeles Lakers',
      totalWins: 65
    }
  ]
}
```

Figure 3.6: Result

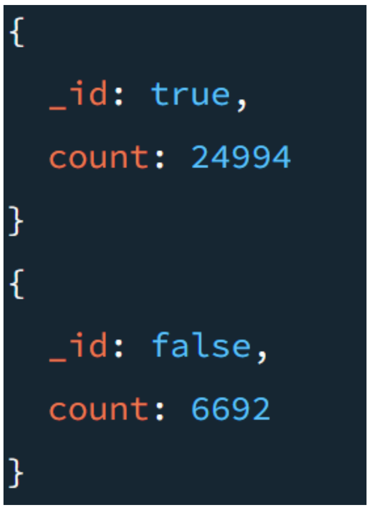
3.2. Games queries

In this section, we aim to provide the reader with some interesting statistics about individual games and analyze how they are connected to other important metrics, such as rebounds, turnovers, and other crucial statistics. The goal is to demonstrate how a player's performance in a single game can reflect or influence other aspects of the game, offering a deeper understanding of the dynamics that determine the outcome of a match. Through these statistics, we can highlight how factors like efficiency in rebounds or ball control can have a direct impact on the overall result of the game and a team's effectiveness.

3.2.1. Link between field goals made and wins

Here, we're trying to figure out how the number of field goals a team makes in a game relates to whether they win or lose. To do this, we look at each game and check which team made more field goals. Then, we count up the number of games where the team with more field goals actually won. The results show that the team with more field goals only won about 78.88 percent of the time. This makes sense when you consider that we're not distinguishing between two-pointers and three-pointers, and we're not even looking at free throws. So, while making more field goals is definitely a good thing, it's not the whole story when it comes to winning games. There are other factors at play too.

```
nba > db. games. aggregate([
    { $unwind: '$box' },
    {
        $project: {
            _id: '$_id',
            stat: {
                $cond: [
                    { $gt'$box.won'0
                    '$box. team. ast',
                    { $multiply: ['$box. team.fg', -1] }
                ]
            }
        }
    },
    { $group: {_id: '$_id', stat: { $sum: '$stat' } } },
    { $project: {_id: '$_id', winningTeamHigher:
        { $gte: ['$stat', 0] } } }, { $group:
        {_id: '$winningTeamHigher', count: { $sum: 1 } } }
]);
```



```
{
  _id: true,
  count: 24994
}
{
  _id: false,
  count: 6692
}
```

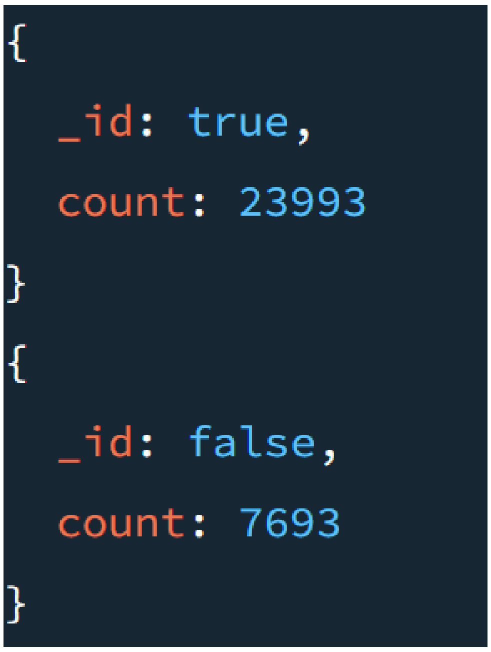
Figure 3.7: Result

3.2.2. Link between defensive rebounds and wins

Since we just saw that making more field goals doesn't always guarantee a win, let's look at another important aspect of the game: defensive rebounds. We're going to do a similar analysis here, but instead of field goals, we'll check which team got more defensive rebounds in each game. Then, we'll count how many times the team with more defensive rebounds ended up winning. The idea is that defensive rebounds are crucial because they give you possession of the ball and prevent the other team from getting second-chance points. So, teams that consistently grab more defensive boards might have an edge in terms of winning games. By looking at this, we can get a more well-rounded view of what contributes to victories, beyond just putting the ball in the hoop more times than the other team. In this case, the team with more defensive rebounds won 75 percent of the time, which is pretty close to the 78.88 percent we saw with field goals. This suggests that defensive rebounds are nearly as important as field goals when it comes to winning games.

```
nba > db. games. aggregate([
    { Sunwind: '$box' },
    {
        $project: {
            _id: '$_id',
            stat: {
                $cond: [
                    { $gt'$box.won'0
```

```
                        '$box. team. ast',
                        { $multiply: ['$box. team.drb', -1] }
                    ]
                }
            }
        },
        { $group: {_id: '$_id', stat: { $sum: '$stat' } } },
        { Sproject: {_id: '$_id', winningTeamHigher:
            { $gte: ['$stat', 0] } } }, { $group:
            {_id: '$winningTeamHigher', count: { $sum: 1 } } }
    ]);
```



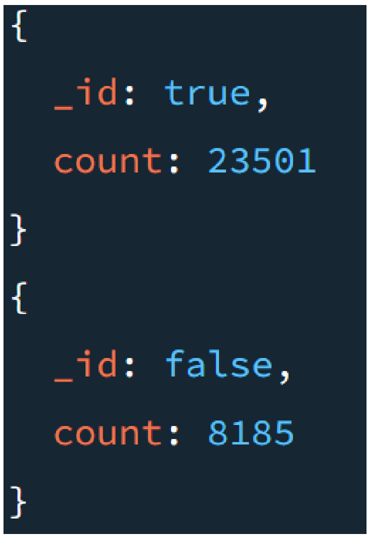
```
{
  _id: true,
  count: 23993
}
{
  _id: false,
  count: 7693
}
```

Figure 3.8: Result

3.2.3. Link between assists and wins

Continuing our exploration of key game statistics and their impact on wins, let's turn our attention to assists. Just as we did with field goals and defensive rebounds, we'll analyze each game to determine which team had more assists and then calculate the percentage of games won by the team with the higher assist count. Interestingly, the team with more assists emerged victorious in 74.2 percent of the games. This winning percentage is comparable to what we observed with defensive rebounds (75 percent) and not too far off from the impact of field goals made (78.88 percent). These findings underscore the importance of teamwork and ball movement in securing wins. Assists are a direct measure of a team's ability to work together, create scoring opportunities for one another, and execute effectively on offense. Teams that consistently rack up more assists than their opponents are likely to have a more fluid, efficient, and balanced offensive attack, which translates to a higher likelihood of winning.

```
nba > db. games. aggregate([
    { $unwind: '$box' },
    {
        $project: {
            _id: '$_id',
            stat: {
                $cond: [
                    { $gt'$box.won'0
                    '$box. team. ast',
                    { $multiply: ['$box. team.ast', -1] }
                ]
            }
        }
    },
    { $group: {_id: '$_id', stat: { $sum: '$stat' } } },
    { $project: {_id: '$_id', winningTeamHigher:
        { $gte: ['$stat', 0] } } }, { $group:
        {_id: '$winningTeamHigher', count: { $sum: 1 } } }
]);
```



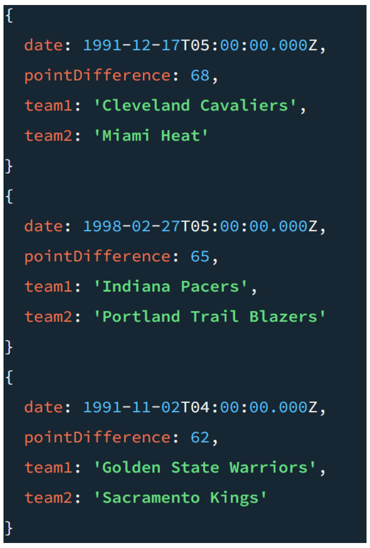
```
{
  _id: true,
  count: 23501
}
{
  _id: false,
  count: 8185
}
```

Figure 3.9: Result

3.2.4. Games with the largest point difference between the two team

After examining various trends and patterns in the data, we wanted to explore some of the more unusual or extreme cases. One area that piqued our interest was the point differential between winning and losing teams. To dive into this, we crafted a query to find the games with the largest margins of victory or defeat. This query retrieves the top 5 basketball games with the largest point difference, showing the names of the two teams, the game date, and the point difference. It calculates the absolute point difference between the two teams and sorts the results in descending order.

```
nba > db. games. aggregate([
    { $unwind: "$teams" },
    { $group: {_id: "$_id", teams: { $push: "$teams" },
    date: { $first: "$date" } } }, { $project: { teaml:
        { $arrayElemAt: ["$teams", 0] }, team2:
        { $arrayElemAt: ["$teams", 1] }, date: 1 } },
        { $addFields: { pointDifference: { $abs:
        { $subtract: ["$team.score", "$team2.score"] } } } },
    { $sort: { pointDifference: -1 } },
    { $limit: 5 },
    { $project: {_id: 0, team1: "$teami.name",
    team2: "$team2.name", date: 1, pointDifference: 1 } }
]) ;
```



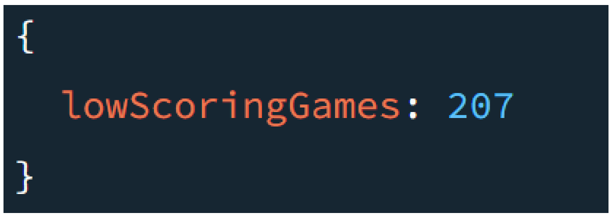
```
{
  date: 1991-12-17T05:00:00.000Z,
  pointDifference: 68,
  team1: 'Cleveland Cavaliers',
  team2: 'Miami Heat'
}
{
  date: 1998-02-27T05:00:00.000Z,
  pointDifference: 65,
  team1: 'Indiana Pacers',
  team2: 'Portland Trail Blazers'
}
{
  date: 1991-11-02T04:00:00.000Z,
  pointDifference: 62,
  team1: 'Golden State Warriors',
  team2: 'Sacramento Kings'
}
```

Figure 3.10: Result

3.2.5. Low scoring games

Studying low-scoring games from a particular year, like 1985 or 1999, offers insights into the playing style and trends of that era. By counting games where the combined score was below 170, we can gauge the prevalence of defensive battles and slower-paced play. This query helps us track the evolution of the game, as scoring patterns have changed due to rule changes, new strategies, and player skills. It also highlights the effectiveness of defenses. This query counts the number of games played in 1985 where the combined score of both teams was less than 170. It filters games by date, sums the scores of the two teams for each game, and then counts the matches meeting the condition.

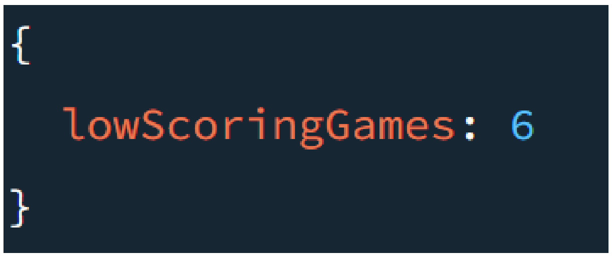
```
db. games. aggregate([
    { $match: { date: { $gte: new Date("1999-01-01T00:00:00Z"),
        $lt: new Date("2000-01-01T00:00:00Z") } } },
    { $unwind: "$teams" },
    { $group: {_id: "$_id", totalPoints: { $sum: "$teams score" } } },
    { $match: { totalPoints: { $lt: 170 } } },
    { $count: "lowScoringGames" }
]) ;
```



```
{
  lowScoringGames: 207
}
```

Figure 3.11: Result 1999

```
db. games. aggregate([
    { $match: { date: { $gte: new Date("1985-01-01T00:00:00Z"),
        $lt: new Date("1986-01-01T00:00:00Z") } } },
    { $unwind: "$teams" },
    { $group: {_id: "$_id", totalPoints: { $sum: "$teams score" } } },
    { $match: { totalPoints: { $lt: 170 } } },
    { $count: "lowScoringGames" }
]) ;
```



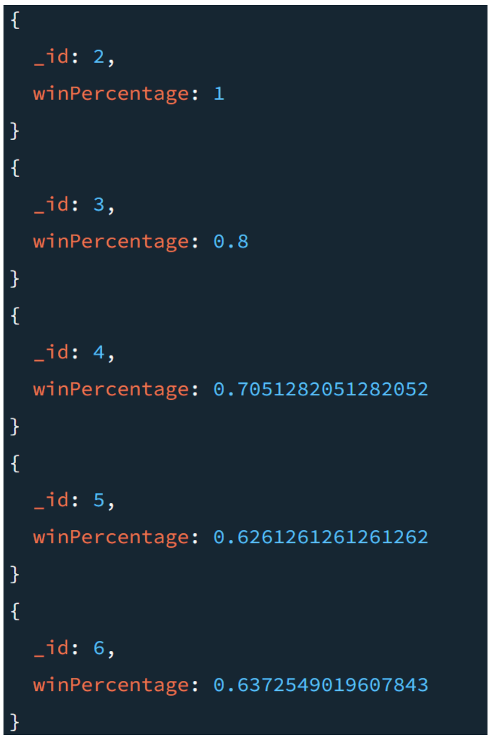
```
{
  lowScoringGames: 6
}
```

Figure 3.12: Result 1985

3.2.6. Relation between turnovers and wins

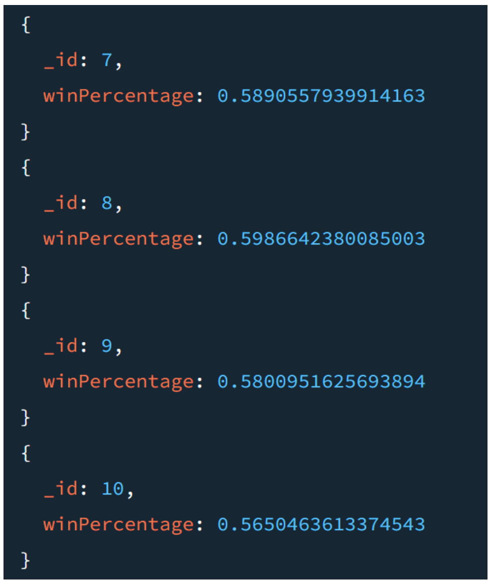
In this query, we aim to explore the relationship between the number of turnovers a team commits and their chances of winning. By calculating the percentage of wins associated with each number of turnovers, we can gain insights into how protecting the ball impacts a team's success. To accomplish this, we first group the games based on the number of turnovers committed by each team. Then, for each specific number of turnovers, we count the total games played and the number of games won by teams committing that many turnovers. Finally, we calculate the percentage of wins for each turnover count by dividing the games won by the total games played and multiplying by 100.

```
db. games. aggregate[
    { $unwind : '$box' },
    {$group : {_id: '$box. team. tov', winPercentage :
        { $avg : '$box.won' 33},
        { $sort : {_id : 1 } }
]);
```



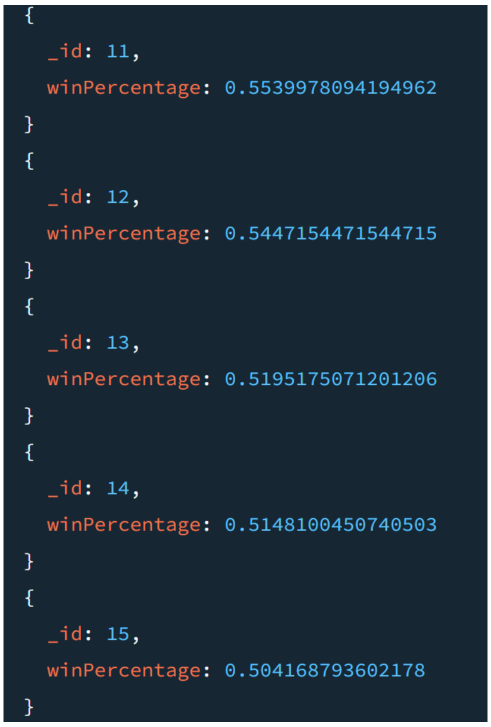
```
{
  _id: 2,
  winPercentage: 1
}
{
  _id: 3,
  winPercentage: 0.8
}
{
  _id: 4,
  winPercentage: 0.7051282051282052
}
{
  _id: 5,
  winPercentage: 0.6261261261261262
}
{
  _id: 6,
  winPercentage: 0.6372549019607843
}
```

Figure 3.13: Result part 1



```
{
  _id: 7,
  winPercentage: 0.5890557939914163
}
{
  _id: 8,
  winPercentage: 0.5986642380085003
}
{
  _id: 9,
  winPercentage: 0.5800951625693894
}
{
  _id: 10,
  winPercentage: 0.5650463613374543
}
```

Figure 3.14: Result part 2



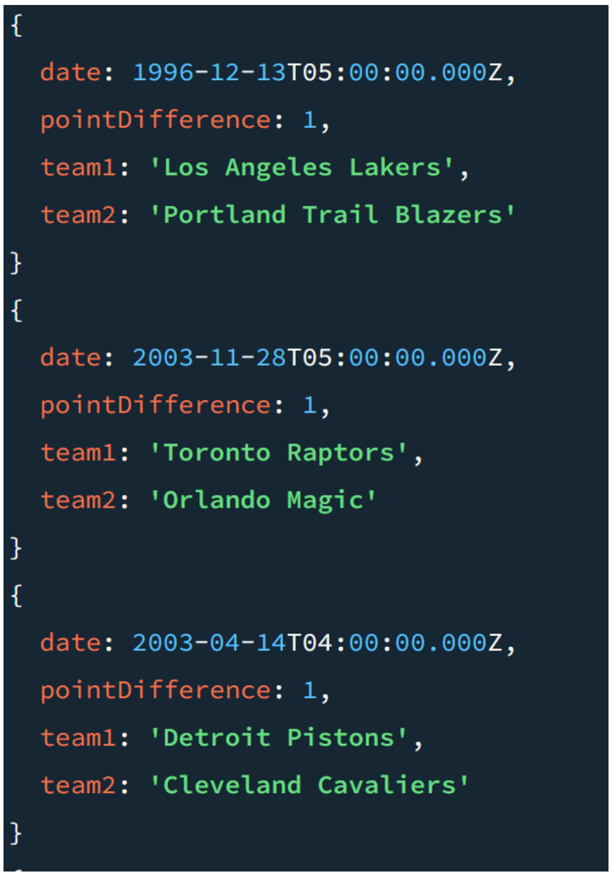
```
{
  _id: 11,
  winPercentage: 0.5539978094194962
}
{
  _id: 12,
  winPercentage: 0.5447154471544715
}
{
  _id: 13,
  winPercentage: 0.5195175071201206
}
{
  _id: 14,
  winPercentage: 0.5148100450740503
}
{
  _id: 15,
  winPercentage: 0.504168793602178
}
```

Figure 3.15: Result part 3

3.2.7. Games with one-point difference

This query finds all the games with a point difference of exactly 1. It unwinds the teams, calculates the point difference, filters for games with a 1-point difference, and then projects the relevant information such as the game date, team names, and point difference.

```
db. games. aggregate([
    { $unwind: "Steams" },
    {
        $group: {
            _id: "$_id",
            teams: { $push: "$teams" date: { $first: "$date" }
        }
    },
    {
        $addFields: {
            pointDifference: {
                $abs: { $subtract: [
                    { $arrayElemAt: ["$teams score", 01 },
                    { $arrayElemAt: ['$teams.score", 1] }
                    ]}
                }
            }
        },
        { $match: { pointDifference: 1 } },
        {$project:{
            _id:0
            date:1
            team1:{arrayElemAt: ["$teams.name", 0]},
            team2:{$arrayElemAt:["$teams.name", 1]},
            pointDifference: 1
        }}
    ]);
```



```
{
  date: 1996-12-13T05:00:00.000Z,
  pointDifference: 1,
  team1: 'Los Angeles Lakers',
  team2: 'Portland Trail Blazers'
}
{
  date: 2003-11-28T05:00:00.000Z,
  pointDifference: 1,
  team1: 'Toronto Raptors',
  team2: 'Orlando Magic'
}
{
  date: 2003-04-14T04:00:00.000Z,
  pointDifference: 1,
  team1: 'Detroit Pistons',
  team2: 'Cleveland Cavaliers'
}
```

Figure 3.16: Result

3.3. Players queries

Analyzing player statistics is crucial for identifying talents to acquire during the NBA market phase. Using a well-structured dataset like the one described, various analyses can be performed to evaluate player performance and identify the most promising ones. For example, you could analyze field goals, assists, rebounds, fouls drawn, or minutes played in specific competitions. By combining multiple metrics, creating composite performance indices, and examining trends over time, teams can make data-driven decisions to spot rising stars and assess consistency, efficiency, and impact in different game contexts.

3.3.1. Best scorers

Identifying the best scorers in the NBA is crucial for understanding the game's top offensive talents. By analyzing points per game, we can pinpoint the players who consistently put up impressive scoring numbers and have the biggest impact on their team's offensive success. This query helps highlight the stars who can take over games with their scoring prowess and serve as the go-to options for their teams. This query finds the player with the most total points by summing their scores across all games, sorting in descending order, and limiting the result to the top player.

```
db. games. aggregate([
    { $unwind: "$box" },
    { $unwind: "$box players" },
    { $group: {_id: "$box players.player", totalPoints:
        { $sum: "$box.players.pts" } } },
    { $sort: { totalPoints: -1 } },
    { $limit: 5 }
]) ;
```

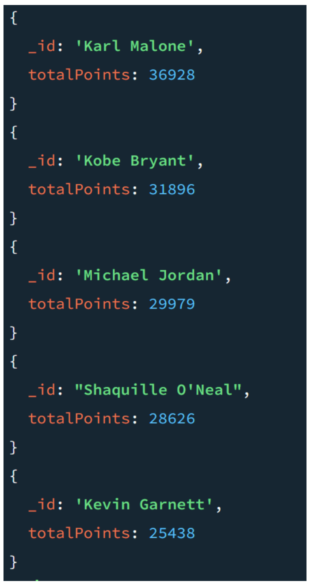
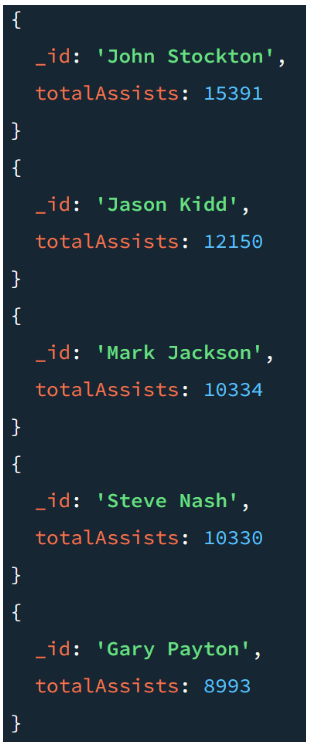

```
{
  _id: 'Karl Malone',
  totalPoints: 36928
}
{
  _id: 'Kobe Bryant',
  totalPoints: 31896
}
{
  _id: 'Michael Jordan',
  totalPoints: 29979
}
{
  _id: "Shaquille O'Neal",
  totalPoints: 28626
}
{
  _id: 'Kevin Garnett',
  totalPoints: 25438
}
```

Figure 3.17: Result

3.3.2. Players with the most assists

Scoring is not the only important thing in basketball and stars can also be the one that not score the most points. Discovering the best playmakers in the NBA is essential for recognizing the players who excel at creating scoring opportunities for their teammates. By examining assists per game, we can identify the players who have a knack for setting up others and keeping the offense running smoothly. This query sheds light on the floor generals who can elevate their team's performance with their vision, passing skills, and ability to make everyone around them better. This query identifies the player with the most total assists by summing their assists across all games, sorting in descending order, and returning the top player.

```
db. games. aggregate([
    { $unwind: "$box" },
    { $unwind: "$box players" },
    { $group: {_id: "$box players.player", totalAssists:
        { $sum: "$box.players.ast" } } },
    { $sort: { totalAssists: -1 } },
    { $limit: 5 }
]) ;
```



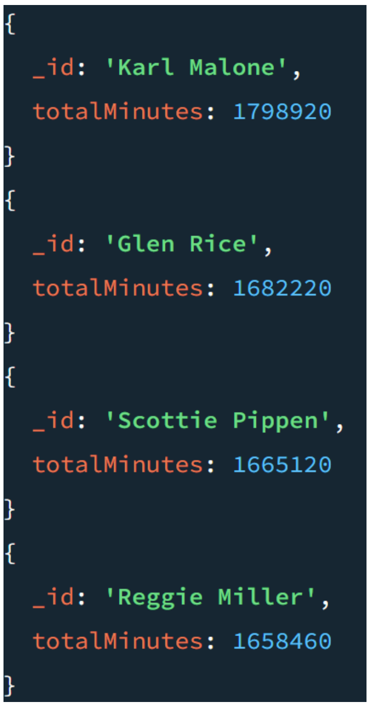
```
{
  _id: 'John Stockton',
  totalAssists: 15391
}
{
  _id: 'Jason Kidd',
  totalAssists: 12150
}
{
  _id: 'Mark Jackson',
  totalAssists: 10334
}
{
  _id: 'Steve Nash',
  totalAssists: 10330
}
{
  _id: 'Gary Payton',
  totalAssists: 8993
}
```

Figure 3.18: Result

3.3.3. Minutes played in a span of time

It could be also very fascinating to explore which players were relied upon most by their coaches over a specific period. This query identifies the player with the highest total minutes played between 1990 and 2000 by converting the mp field into seconds, summing their total playtime across all games, and ranking them accordingly.

```
db. games. aggregate([]
    { $match: { date: { $gte: new Date("1990-01-01T00:00:00Z"),
        $lt: new Date("2000-01-01T00:00:00z") } } },
    { $unwind: "$box" },
    { $unwind: "$box-players" },
    {
        $addFields: {
            minutesPlayed: {
                $sum: [
                    { {multiply: [ { $toInt: { $arrayElemAt: [ $split:
                        ["$box.players.mp", ":"] }, 0] } }, 60 ] },
                    $toInt: { $arrayElemAt: [ $split:
                          ["$box. players mp", ":"] }, 1] } }
                  ]
              }
          }
      },
      { $group: {_id: "$box-players.player", totalMinutes:
          { $sum: "$minutesPlayed" } } }, { $sort: { totalMinutes: -1 } },
      { $limit: 1 }
  ]);
```



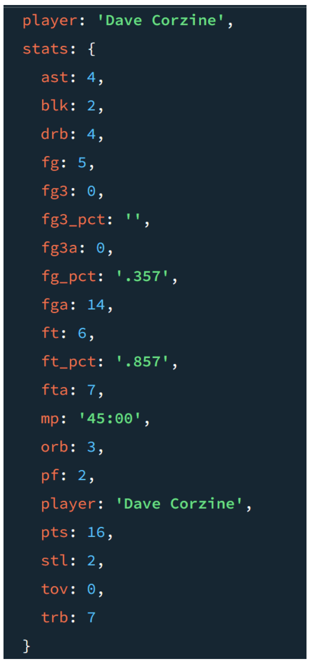
```
{
  _id: 'Karl Malone',
  totalMinutes: 1798920
}
{
  _id: 'Glen Rice',
  totalMinutes: 1682220
}
{
  _id: 'Scottie Pippen',
  totalMinutes: 1665120
}
{
  _id: 'Reggie Miller',
  totalMinutes: 1658460
}
```

Figure 3.19: Result

3.3.4. Players in a specific match

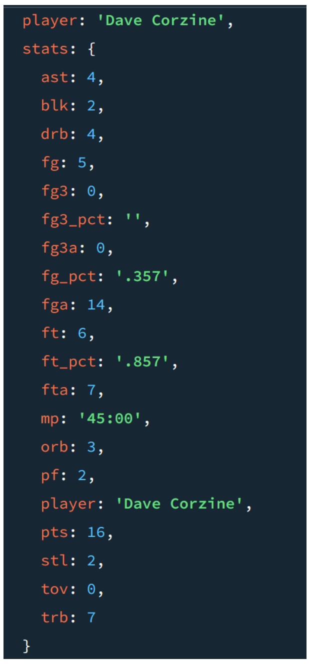
This query retrieves all players who appeared in the game with the specified id (here we have chosen "52f2a34fddbd75540aba74dd"). It first filters the game using the match stage. Then, it unwinds both the box array (which contains information about the teams) and the players array (which holds individual player stats). Finally, it projects the player's name along with all their statistics from the players array, omitting the id field for a cleaner result.

```
  db. games. aggregate([
      { $match: {_id: ObjectId ObjectId("52f2a34fddbd75540aba74dd") } },
        { $unwind: "$box" },
        { $unwind: "$box players" },
        { $project: {_id: _id: player: "$box.players.player",
            stats: "$box players" } }
    ]);
```



```
player: 'Dave Corzine',
stats: {
  ast: 4,
  blk: 2,
  drb: 4,
  fg: 5,
  fg3: 0,
  fg3_pct: '',
  fg3a: 0,
  fg_pct: '.357',
  fga: 14,
  ft: 6,
  ft_pct: '.857',
  fta: 7,
  mp: '45:00',
  orb: 3,
  pf: 2,
  player: 'Dave Corzine',
  pts: 16,
  stl: 2,
  tov: 0,
  trb: 7
}
```

Figure 3.20: Result part 1



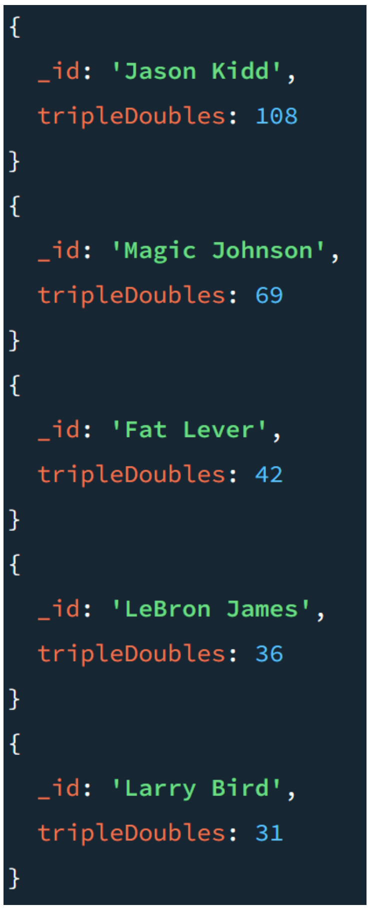
```
player: 'Dave Corzine',
stats: {
  ast: 4,
  blk: 2,
  drb: 4,
  fg: 5,
  fg3: 0,
  fg3_pct: '',
  fg3a: 0,
  fg_pct: '.357',
  fga: 14,
  ft: 6,
  ft_pct: '.857',
  fta: 7,
  mp: '45:00',
  orb: 3,
  pf: 2,
  player: 'Dave Corzine',
  pts: 16,
  stl: 2,
  tov: 0,
  trb: 7
}
```

Figure 3.21: Result part 2

3.3.5. Players with more triple doubles

A triple-double in the NBA is an impressive achievement, where a player records double-digit numbers in three of five key stats: points, rebounds, assists, steals, and blocks. Most commonly, it's at least 10 points, 10 rebounds, and 10 assists in one game. Getting a triple-double highlights a player's versatility and all-around impact. It shows they can excel in multiple aspects of the game at once, making big contributions in scoring, rebounding, and setting up their teammates. This query looks for triple-doubles, where a player has at least 10 assists, 10 rebounds, and 10 points in a game. It first filters the games for those criteria, then groups by player and counts the number of triple-doubles each player has achieved. The results are sorted by the number of triple-doubles in descending order, and the top 10 players are returned.

```
db. games. aggregate ([
    { $unwind: "$box" },
    { $unwind: "$box players" },
    {
        $match: {
            $and: [
                { "box players.ast": { $gte: 10 } },
                { "box-players.trb": { $gte: 10 } },
                { "box-players-pts": { $gte: 10 } }
            ]
        }
    },
    { $group: {_id: "$box.players.player", tripleDoubles:
        { $sum: 1 } } },
    { $sort: { tripleDoubles: -1 } },
    { $limit: 10 }
]) ;
```



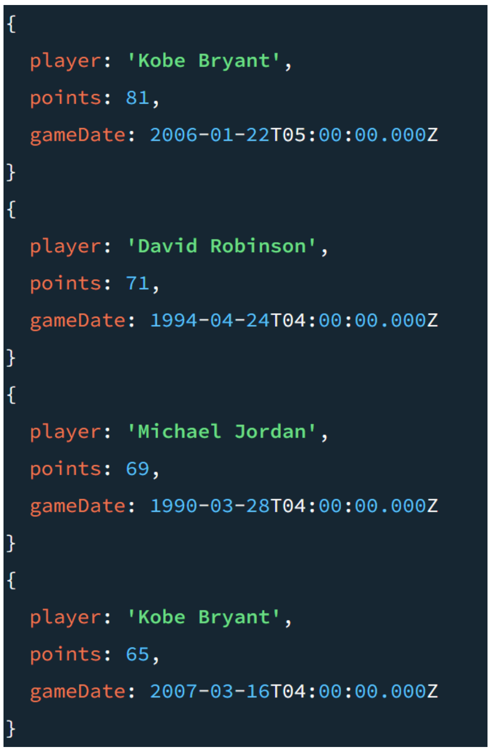
```
{
  _id: 'Jason Kidd',
  tripleDoubles: 108
}
{
  _id: 'Magic Johnson',
  tripleDoubles: 69
}
{
  _id: 'Fat Lever',
  tripleDoubles: 42
}
{
  _id: 'LeBron James',
  tripleDoubles: 36
}
{
  _id: 'Larry Bird',
  tripleDoubles: 31
}
```

Figure 3.22: Result

3.3.6. Best scoring performances

This query finds the best individual scoring performances by sorting the players' points (pts) in descending order. It then limits the result to the top 10 players with the highest points, projecting the player's name, points, and the date of the game.

```
db. games. aggregate(L
    { $unwind: "$box" },
    { $unwind: "$box-players" },
    { $sort: { "box.players.pts": -1 } },
    { $limit: 10 },
    { $project: {_id: 0, player: "$box-players player",
        points: "$box.players.pts", gameDate: "$date" } }
]);
```



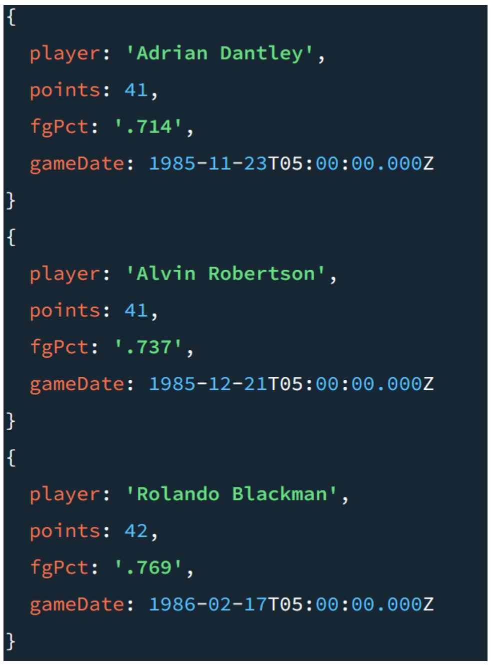
```
{
  player: 'Kobe Bryant',
  points: 81,
  gameDate: 2006-01-22T05:00:00.000Z
}
{
  player: 'David Robinson',
  points: 71,
  gameDate: 1994-04-24T04:00:00.000Z
}
{
  player: 'Michael Jordan',
  points: 69,
  gameDate: 1990-03-28T04:00:00.000Z
}
{
  player: 'Kobe Bryant',
  points: 65,
  gameDate: 2007-03-16T04:00:00.000Z
}
```

Figure 3.23: Result

3.3.7. Efficient performances

This query finds games where a player scored over 40 points with a field goal percentage above 70 percent. These performances showcase a player's ability to score efficiently at a high volume. By analyzing these games, we can identify players who excel at efficient scoring and explore trends across eras. This query highlights the value of shooting accuracy and its impact on offensive excellence in the NBA. This query finds players who scored over 40 points and have a field goal percentage greater than 70 percent. It converts the fg-pct string to a number using toDouble and filters accordingly. The results show the player's name, points, field goal percentage, and game date.

```
db. games. aggregate([
    { $unwind: "$box" },
    { $unwind: "$box-players" },
    {
        $match: {
            $and: [
                { "box-players.pts": { $gt: 40 } },
                $expr: { $gt: [ { $toDouble: "$box-players.fg_pct" },
                    0.71] } }
                ]
            }
        },
        { $project: {_id: 0, player: "$box-players-player",
            points: "$box-players.pts", fgPct:
            "$box.players.fg_pct", gameDate: "$date" } }
    ]);
```



```
{
  player: 'Adrian Dantley',
  points: 41,
  fgPct: '.714',
  gameDate: 1985-11-23T05:00:00.000Z
}
{
  player: 'Alvin Robertson',
  points: 41,
  fgPct: '.737',
  gameDate: 1985-12-21T05:00:00.000Z
}
{
  player: 'Rolando Blackman',
  points: 42,
  fgPct: '.769',
  gameDate: 1986-02-17T05:00:00.000Z
}
```

Figure 3.24: Result

4 | EXTRA

4.1. How certain stats impact on the outcome of the match

By leveraging graphs, we aim to provide a visual representation of how different performance metrics, such as three-point scored, defensive rebounds, blocks, and other factors, correlate with a team's success in individual games, in particular how the number of points/defensive rebounds/etc secure a certain win percentage.

The visualizations allow us to see patterns and trends that may not be immediately apparent from raw numbers alone. For example, they can help us identify which metrics have the strongest influence on determining the outcome of a game. By breaking down these connections, we can gain a deeper insight into how certain aspects of gameplay contribute to building a winning strategy.

This approach not only helps in understanding the significance of individual statistics but also demonstrates the complex interplay between offensive and defensive metrics and their collective impact on team performance. Through this analysis, we hope to shed light on how these factors interact and contribute to a team's overall success on the court.

4.1.1. Result of the analysis

As mentioned earlier in the queries chapter, the most impactful statistics for securing a win are, unsurprisingly, field goals (though perhaps not as decisive as one might initially assume) in particular when a team performs more than 15 three-point a game the probability to win the game is almost 0.8 and Defensive Rebounds. Turnovers also play an important role in determining the outcome of a game, their influence is generally less pronounced compared to the first two metrics.

Three-pointField goals directly contribute to the score, making them a key driver of success, but the nuances of the game reveal that other factors, like defensive rebounds, are equally, if not more, critical in maintaining possession and limiting the opponent's scoring opportunities. Turnovers, which reflect ball control, are also significant but tend to have a less impact, depending on the dynamics of the match and . This hierarchy of importance provides valuable insights into how different aspects of the game interact to shape the final outcome.

Blocks and Wins

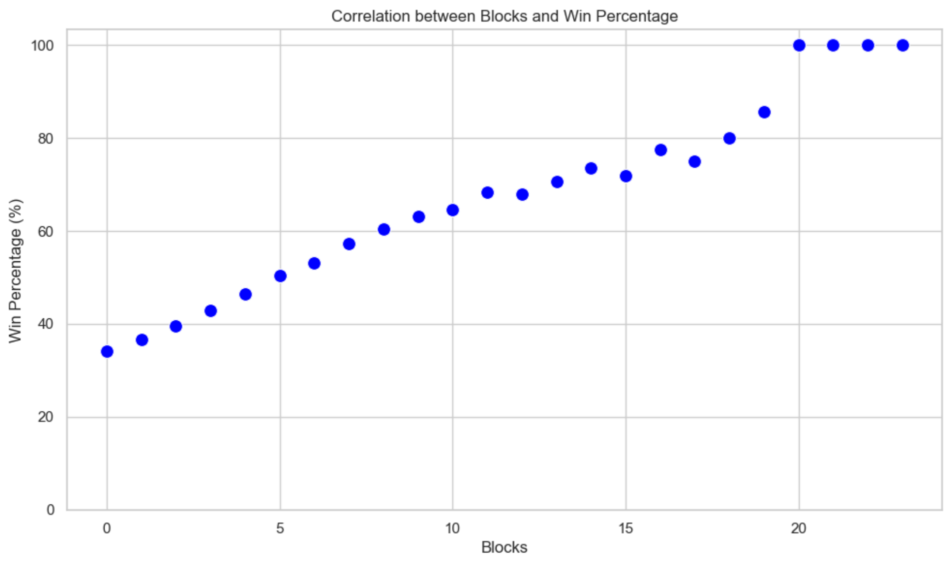


Figure 4.1: How Blocks impact a game

Turnovers and Wins

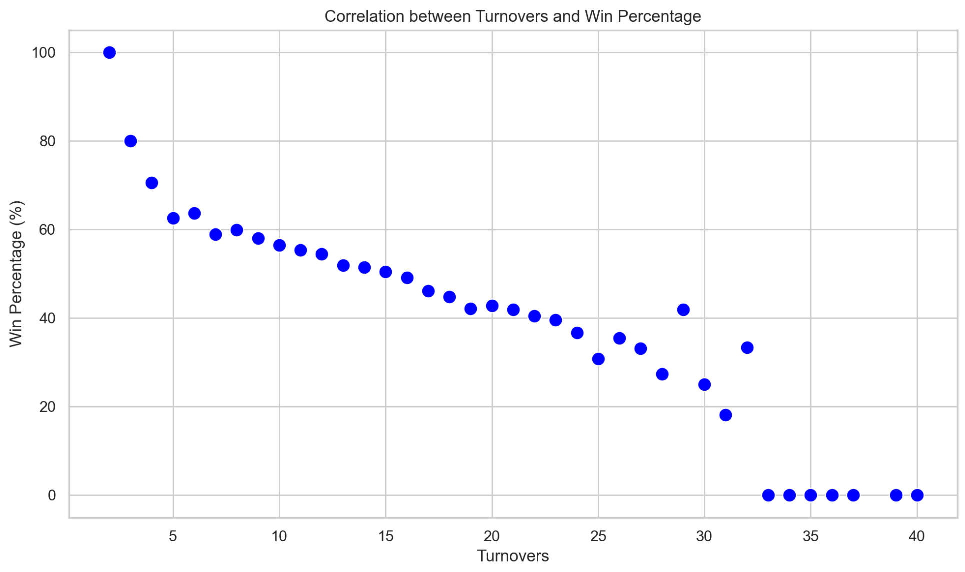


Figure 4.2: How Turnovers impact a game

Defensive Rebounds and Wins

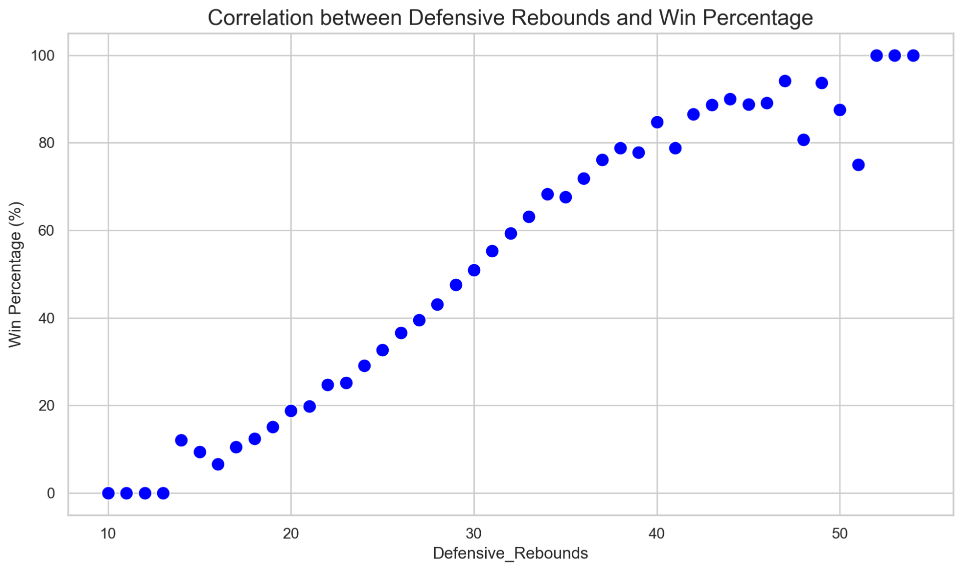


Figure 4.3: How Defensive Rebounds impact a game

Three-Point Field Goals and Wins

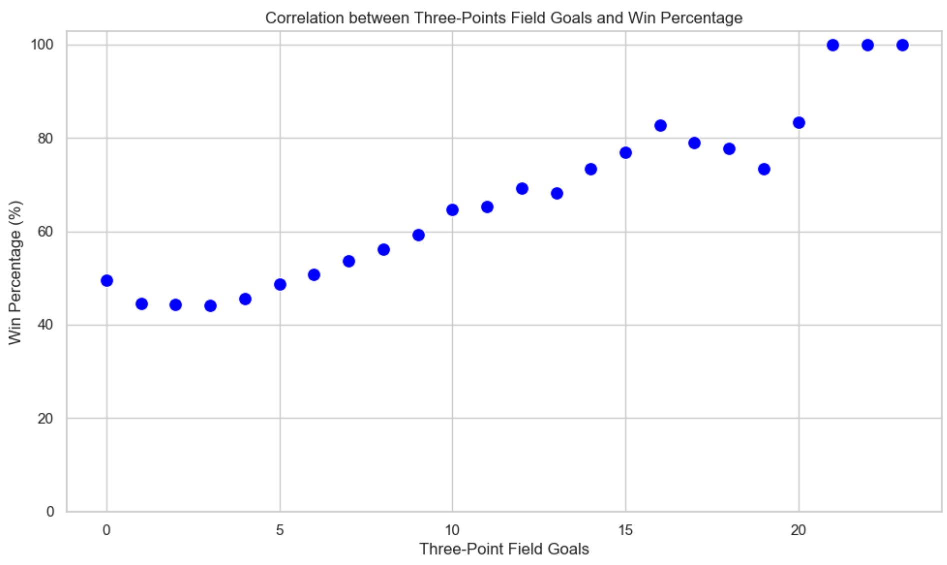
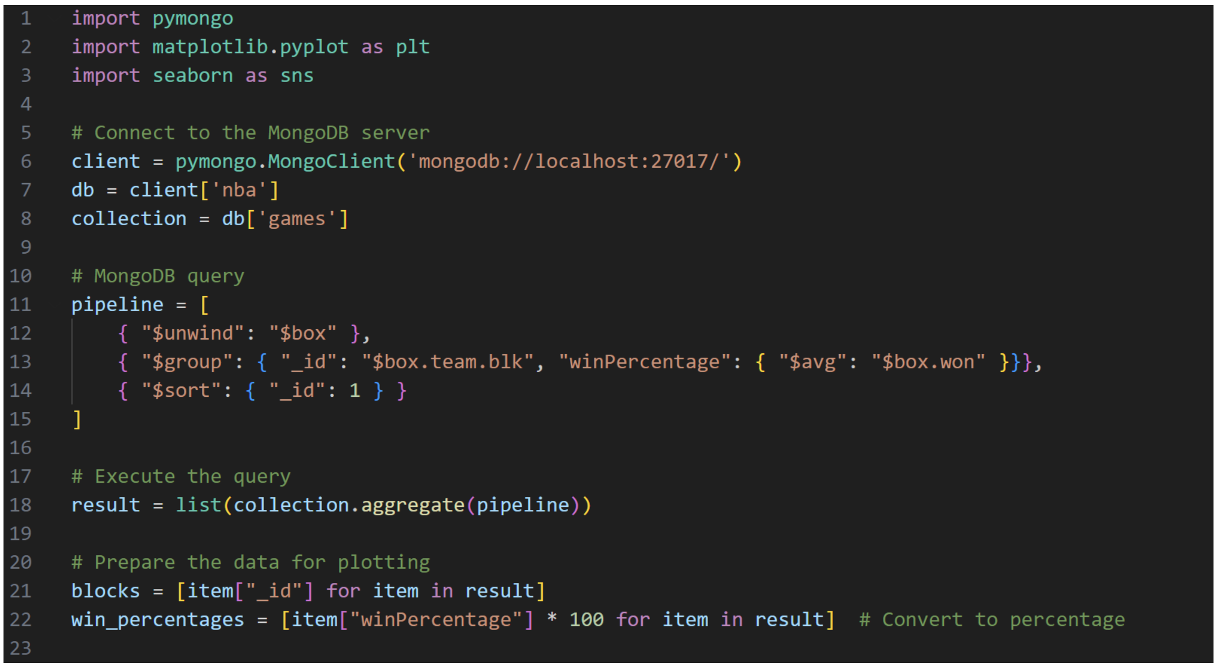


Figure 4.4: How Three-Point Field Goals impact a game

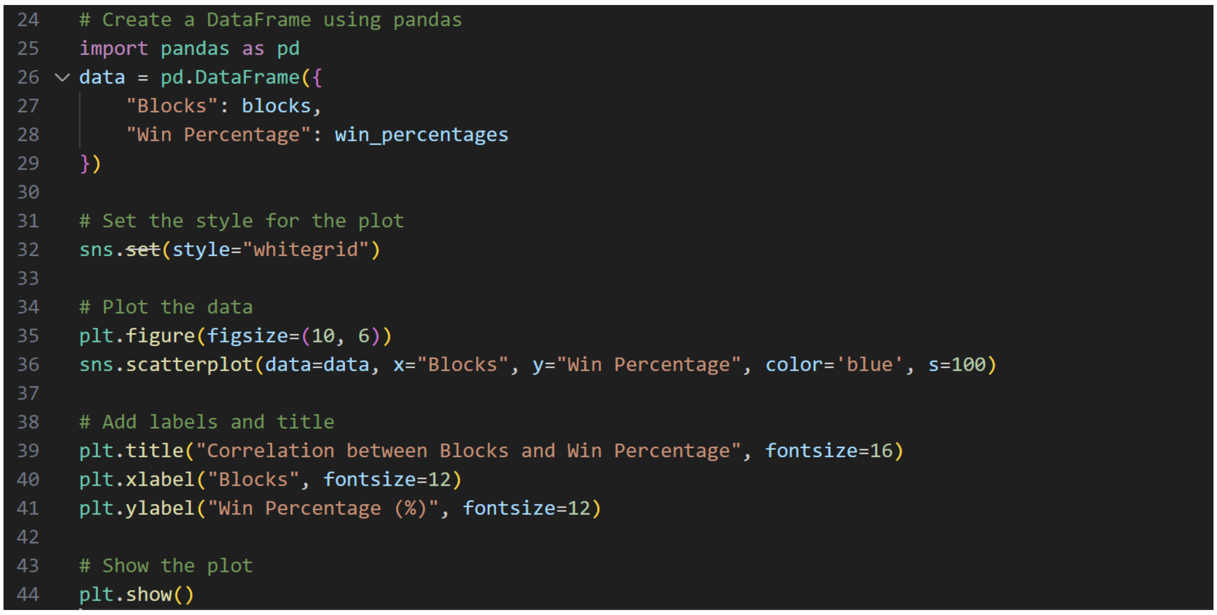
4.1.2. Python Code

Blocks Python code



```python
import pymongo
import matplotlib.pyplot as plt
import seaborn as sns

# Connect to the MongoDB server
client = pymongo.MongoClient('mongodb://localhost:27017/')
db = client['nba']
collection = db['games']

# MongoDB query
pipeline = [
    { "$unwind": "$box" },
    { "$group": { "_id": "$box.team.blk", "winPercentage": { "$avg": "$box.won" }}},
    { "$sort": { "_id": 1 } }
]

# Execute the query
result = list(collection.aggregate(pipeline))

# Prepare the data for plotting
blocks = [item["_id"] for item in result]
win_percentages = [item["winPercentage"] * 100 for item in result]  # Convert to percentage

```

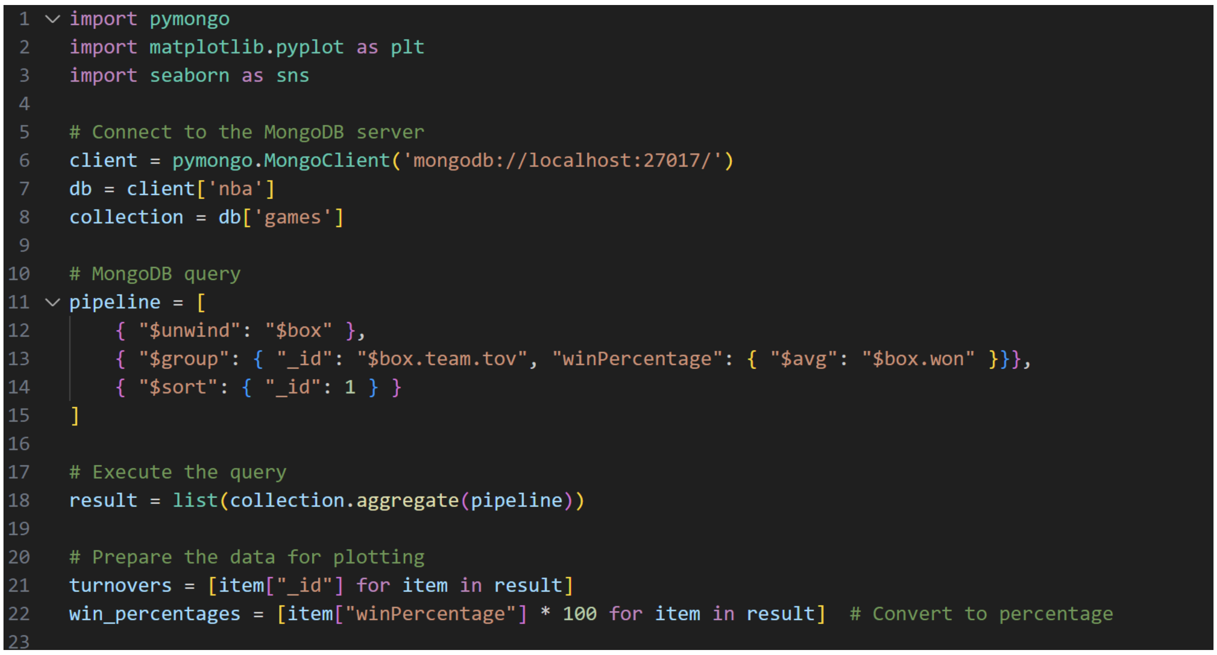
Figure 4.5: Blocks Python part 1



```python
# Create a DataFrame using pandas
import pandas as pd
data = pd.DataFrame({
    "Blocks": blocks,
    "Win Percentage": win_percentages
})

# Set the style for the plot
sns.~~set~~(style="whitegrid")

# Plot the data
plt.figure(figsize=(10, 6))
sns.scatterplot(data=data, x="Blocks", y="Win Percentage", color='blue', s=100)

# Add labels and title
plt.title("Correlation between Blocks and Win Percentage", fontsize=16)
plt.xlabel("Blocks", fontsize=12)
plt.ylabel("Win Percentage (%)", fontsize=12)

# Show the plot
plt.show()
```

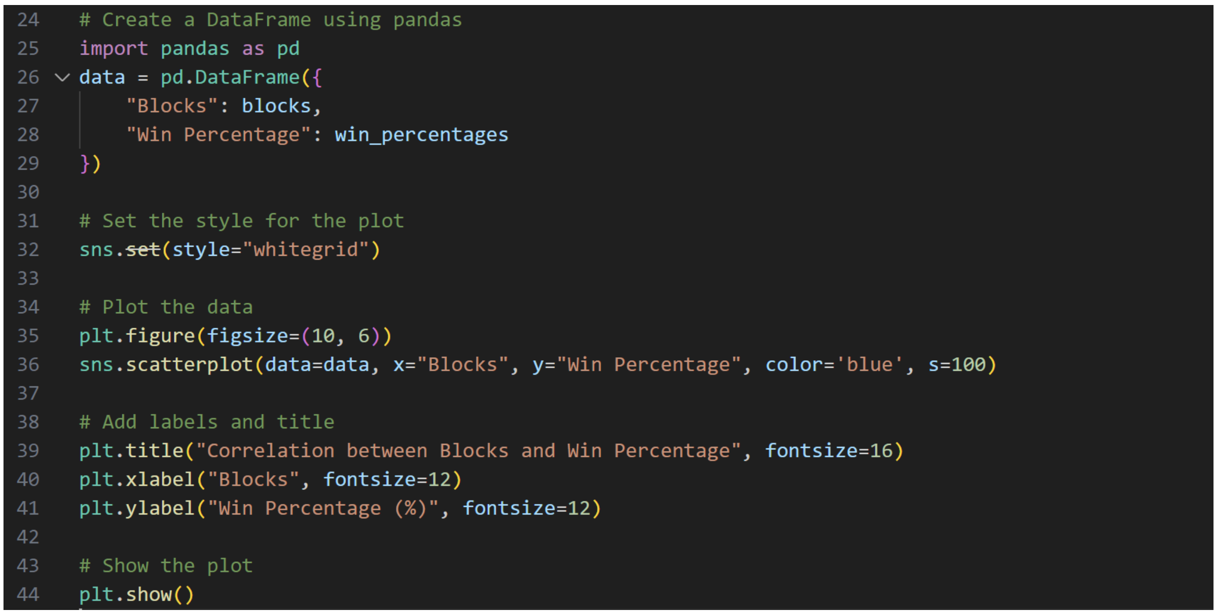
Figure 4.6: Blocks Python part 2

Turnovers Python code



```python
import pymongo
import matplotlib.pyplot as plt
import seaborn as sns

# Connect to the MongoDB server
client = pymongo.MongoClient('mongodb://localhost:27017/')
db = client['nba']
collection = db['games']

# MongoDB query
pipeline = [
    { "$unwind": "$box" },
    { "$group": { "_id": "$box.team.tov", "winPercentage": { "$avg": "$box.won" }}},
    { "$sort": { "_id": 1 } }
]

# Execute the query
result = list(collection.aggregate(pipeline))

# Prepare the data for plotting
turnovers = [item["_id"] for item in result]
win_percentages = [item["winPercentage"] * 100 for item in result]  # Convert to percentage

```

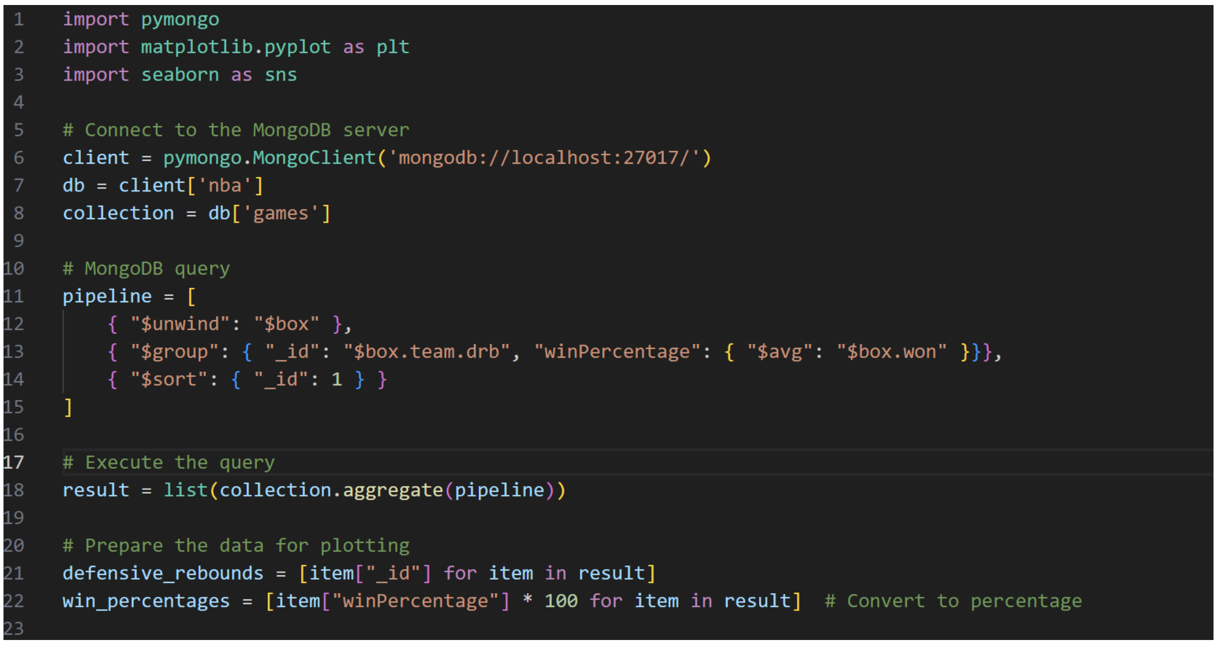
Figure 4.7: Turnovers Python part 1



```python
# Create a DataFrame using pandas
import pandas as pd
data = pd.DataFrame({
    "Blocks": blocks,
    "Win Percentage": win_percentages
})

# Set the style for the plot
sns.set(style="whitegrid")

# Plot the data
plt.figure(figsize=(10, 6))
sns.scatterplot(data=data, x="Blocks", y="Win Percentage", color='blue', s=100)

# Add labels and title
plt.title("Correlation between Blocks and Win Percentage", fontsize=16)
plt.xlabel("Blocks", fontsize=12)
plt.ylabel("Win Percentage (%)", fontsize=12)

# Show the plot
plt.show()
```

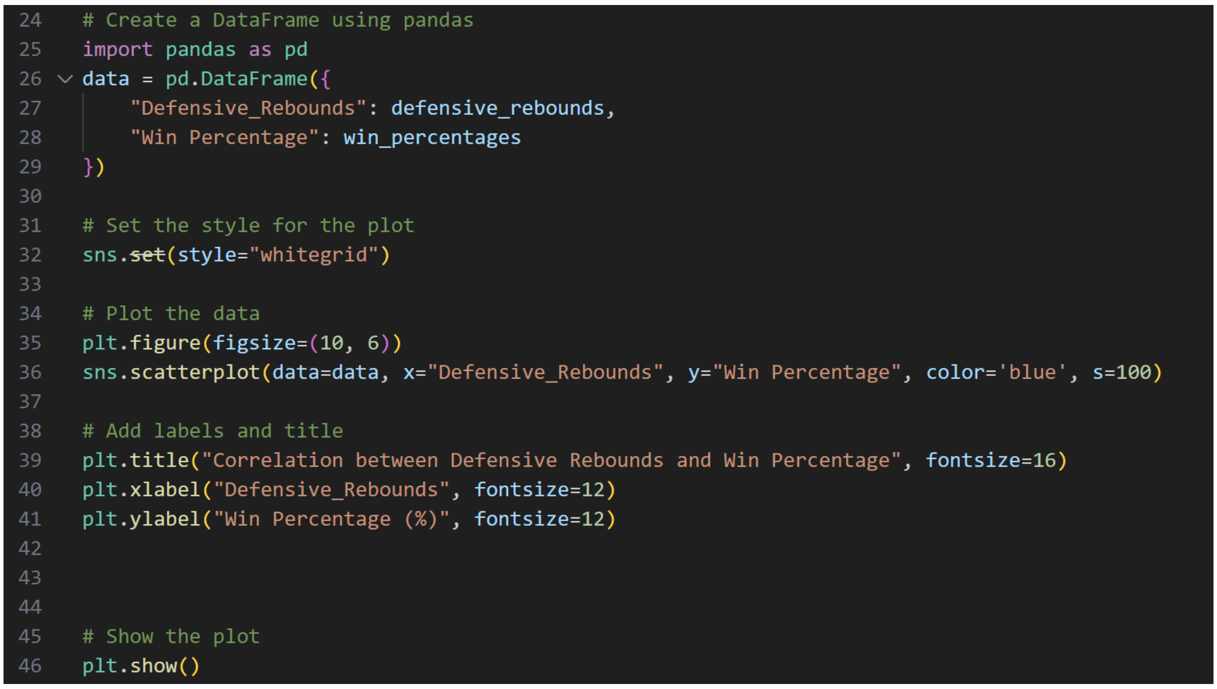
Figure 4.8: Turnovers Python part 2

Defensive Rebounds Python code



```python
import pymongo
import matplotlib.pyplot as plt
import seaborn as sns

# Connect to the MongoDB server
client = pymongo.MongoClient('mongodb://localhost:27017/')
db = client['nba']
collection = db['games']

# MongoDB query
pipeline = [
    { "$unwind": "$box" },
    { "$group": { "_id": "$box.team.drb", "winPercentage": { "$avg": "$box.won" }}},
    { "$sort": { "_id": 1 } }
]

# Execute the query
result = list(collection.aggregate(pipeline))

# Prepare the data for plotting
defensive_rebounds = [item["_id"] for item in result]
win_percentages = [item["winPercentage"] * 100 for item in result]  # Convert to percentage

```

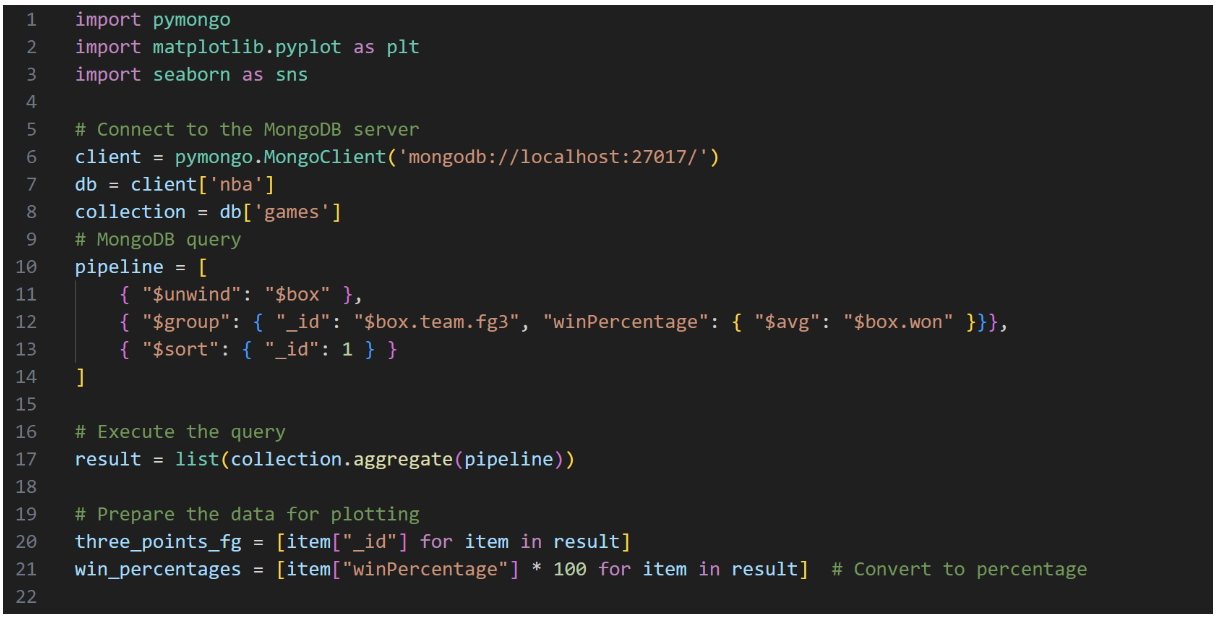
Figure 4.9: Defensive Rebounds Python part 1



```python
# Create a DataFrame using pandas
import pandas as pd
data = pd.DataFrame({
    "Defensive_Rebounds": defensive_rebounds,
    "Win Percentage": win_percentages
})

# Set the style for the plot
sns.set(style="whitegrid")

# Plot the data
plt.figure(figsize=(10, 6))
sns.scatterplot(data=data, x="Defensive_Rebounds", y="Win Percentage", color='blue', s=100)

# Add labels and title
plt.title("Correlation between Defensive Rebounds and Win Percentage", fontsize=16)
plt.xlabel("Defensive_Rebounds", fontsize=12)
plt.ylabel("Win Percentage (%)", fontsize=12)



# Show the plot
plt.show()
```

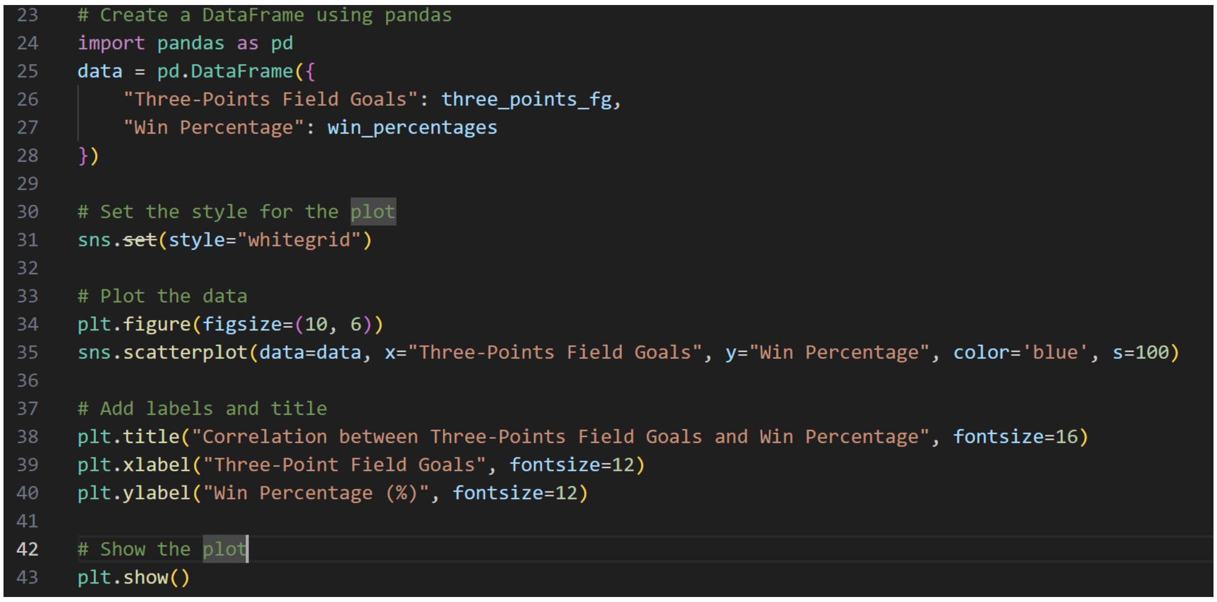
Figure 4.10: Defensive Rebounds Python part 2

Three-Point Field Goals Python code



```python
import pymongo
import matplotlib.pyplot as plt
import seaborn as sns

# Connect to the MongoDB server
client = pymongo.MongoClient('mongodb://localhost:27017/')
db = client['nba']
collection = db['games']
# MongoDB query
pipeline = [
    { "$unwind": "$box" },
    { "$group": { "_id": "$box.team.fg3", "winPercentage": { "$avg": "$box.won" }}},
    { "$sort": { "_id": 1 } }
]

# Execute the query
result = list(collection.aggregate(pipeline))

# Prepare the data for plotting
three_points_fg = [item["_id"] for item in result]
win_percentages = [item["winPercentage"] * 100 for item in result]  # Convert to percentage

```

Figure 4.11: Three-Point Field Goals Python part 1



```python
# Create a DataFrame using pandas
import pandas as pd
data = pd.DataFrame({
    "Three-Points Field Goals": three_points_fg,
    "Win Percentage": win_percentages
})

# Set the style for the plot
sns.~~set~~(style="whitegrid")

# Plot the data
plt.figure(figsize=(10, 6))
sns.scatterplot(data=data, x="Three-Points Field Goals", y="Win Percentage", color='blue', s=100)

# Add labels and title
plt.title("Correlation between Three-Points Field Goals and Win Percentage", fontsize=16)
plt.xlabel("Three-Point Field Goals", fontsize=12)
plt.ylabel("Win Percentage (%)", fontsize=12)

# Show the plot
plt.show()
```

Figure 4.12: Three-Point Field Goals Python part 2

4.2. Evolution of the Three-Point throw

In this section through visual representations, we aim to illustrate how scoring patterns have evolved over the years, particularly highlighting the gradual decrease in the total points scored during games.

Additionally, we explore the impact of the three-point shot, introduced during the 1984/1985 season. Initially, the three-point shot was underutilized and not particularly effective, with teams hesitant to incorporate it as a core part of their strategy. However, over the years, its usage has steadily increased, reflecting advancements in strategy, player skill, and the evolving dynamics of the game.

As of today (2024), nearly half of all shot attempts come from beyond the three-point line, demonstrating how integral it has become to modern basketball. This shift has not only changed the way the game is played but has also contributed to a transformation in offensive strategies and scoring trends over time. The graphs in this section provide a clear visualization of these trends, offering insights into the significant role the three-point shot now plays in shaping the game.

4.2.1. Result of the analysis

As we can observe from the graphs, the league has undergone significant changes in its approach to gameplay over the years. For a substantial period, the game pivoted towards a slower pace, with a heightened focus on the defensive phase. This trend emphasized strategies aimed at limiting opponents' scoring opportunities rather than maximizing offensive output.

However, in more recent years (2024), the game has begun to shift back toward prioritizing the offensive phase, with teams increasingly emphasizing scoring and faster-paced gameplay. This resurgence of offensive strategies has brought about a renewed dynamism to the sport.

Additionally, the introduction of the three-point shot during the 1984/1985 season marked a pivotal moment in basketball history. Initially, the three-point shot was sparingly used, as players lacked the proficiency to shoot consistently from long range. Over time, however, this has changed dramatically. In recent years, the league has seen the emergence of a new breed of players who specialize in shooting from beyond the arc. These players have not only mastered long-range shooting but have also integrated it into their teams' core offensive strategies.

Today, the three-point shot has become a dominant feature of the game, with teams frequently relying on it as a key scoring tool. The data reflects this evolution, showing how the three-point throw has transitioned from a rarely used option to a fundamental aspect of modern basketball. This shift underscores the ongoing evolution of the league and the innovative strategies teams employ to succeed.

Three-Point Field Goal attempted

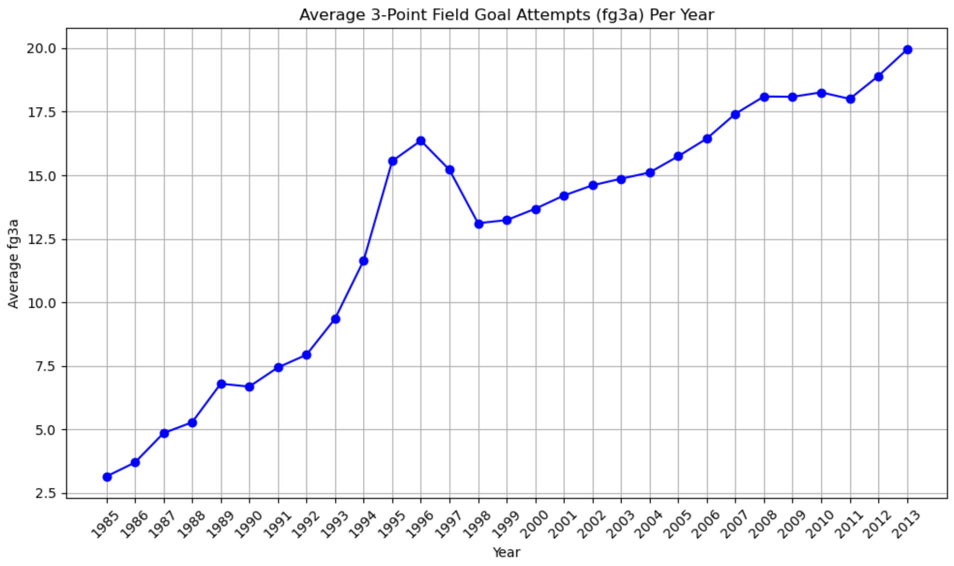


Figure 4.13: Three-Point Field Goals attempted per year

Average Points made

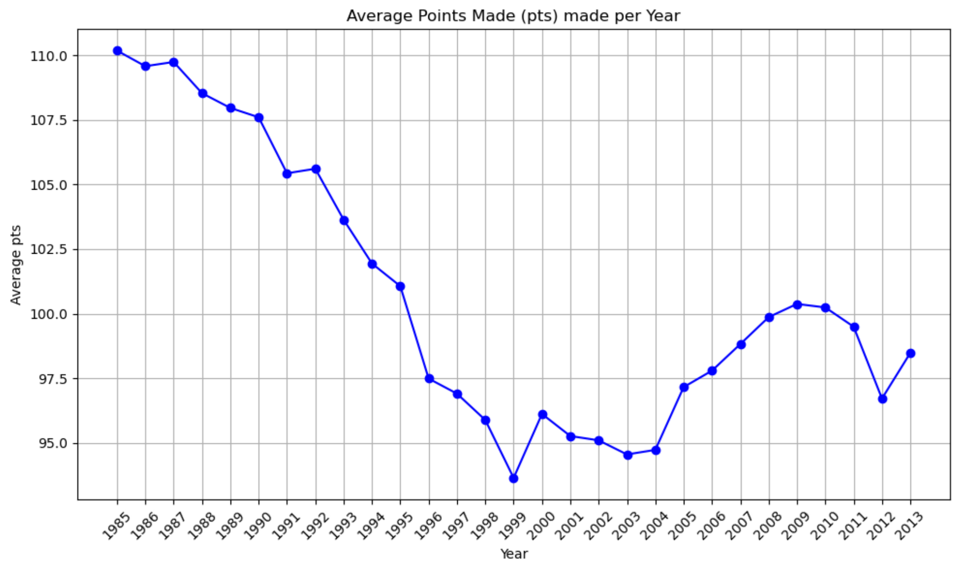
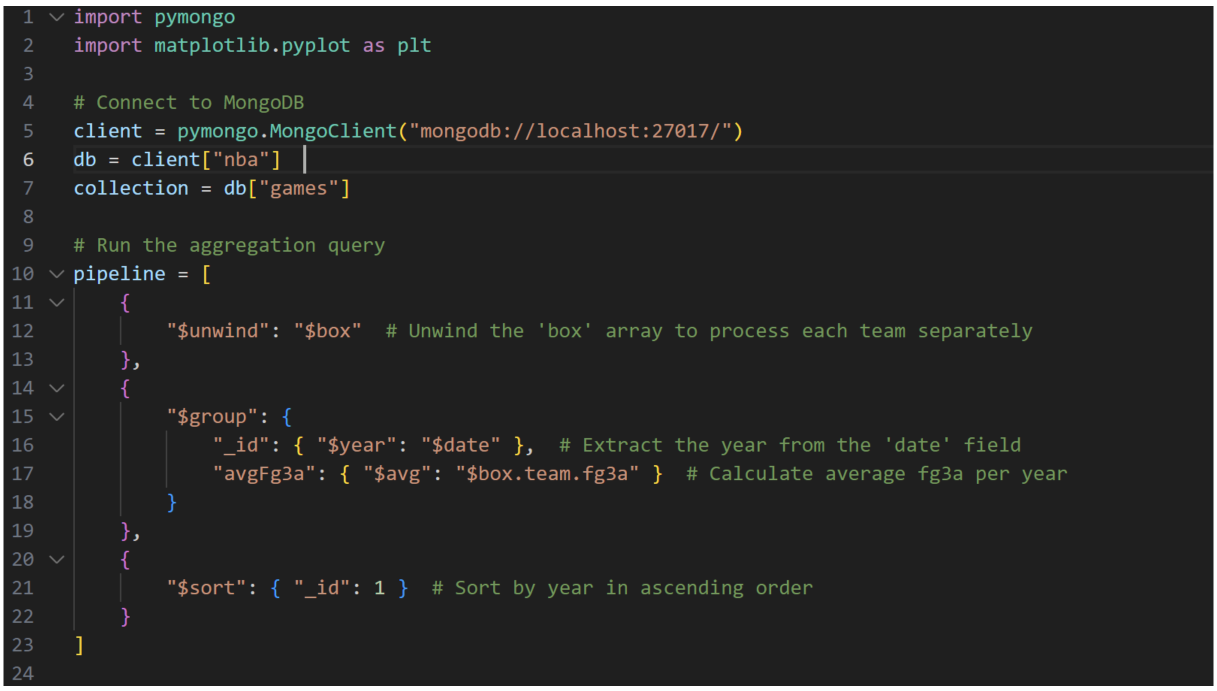


Figure 4.14: Average Points per year

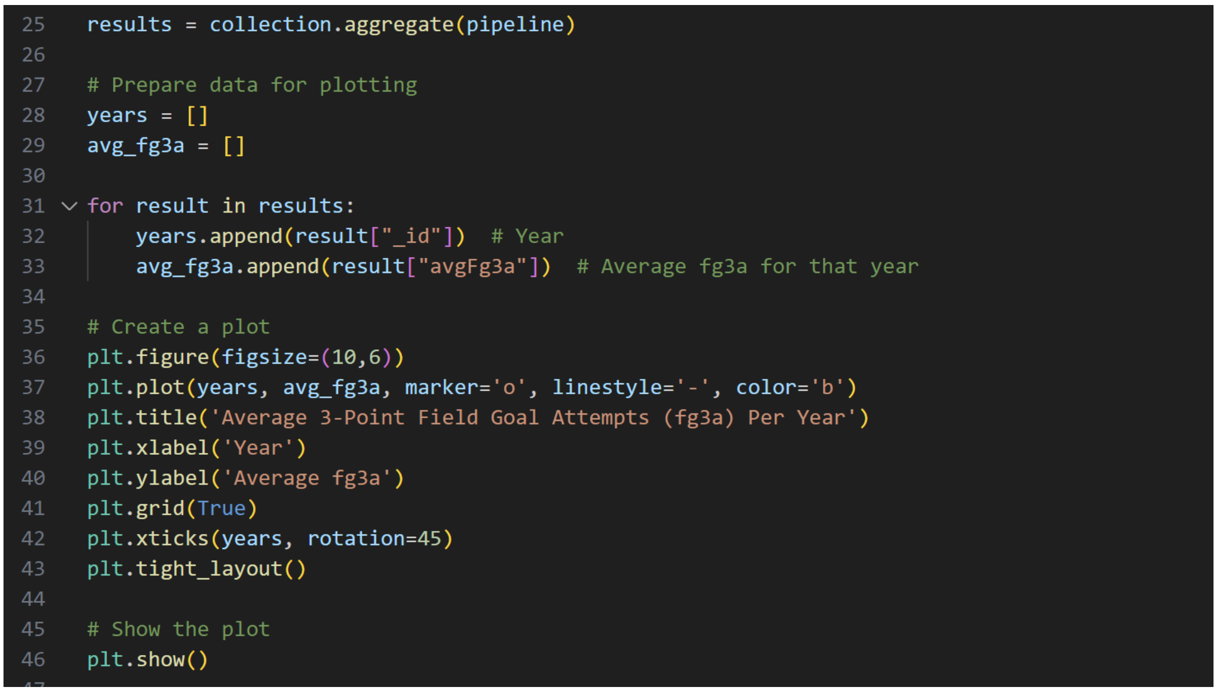
4.2.2. Python Code

Three-Point Field Goal attempted Python code



```
import pymongo
import matplotlib.pyplot as plt

# Connect to MongoDB
client = pymongo.MongoClient("mongodb://localhost:27017/")
db = client["nba"]
collection = db["games"]

# Run the aggregation query
pipeline = [
    {
        "$unwind": "$box"  # Unwind the 'box' array to process each team separately
    },
    {
        "$group": {
            "_id": { "$year": "$date" },  # Extract the year from the 'date' field
            "avgFg3a": { "$avg": "$box.team.fg3a" }  # Calculate average fg3a per year
        }
    },
    {
        "$sort": { "_id": 1 }  # Sort by year in ascending order
    }
]

```

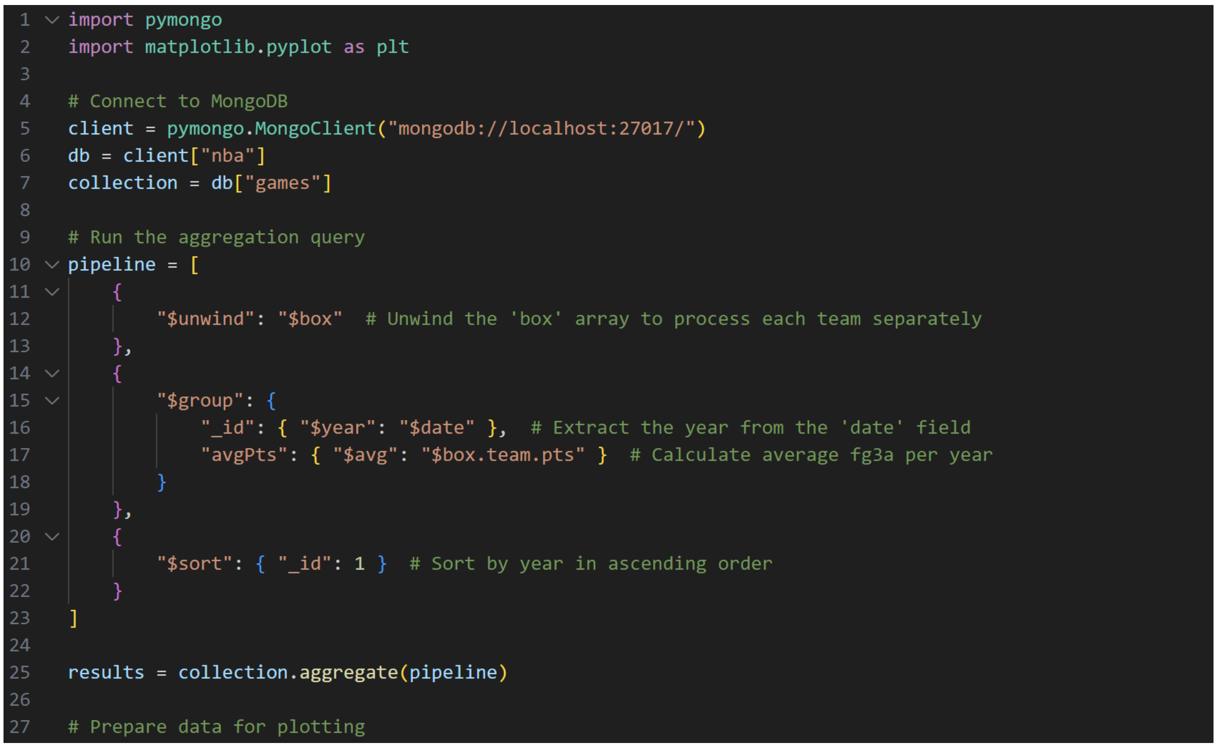
Figure 4.15: Three-Point Field Goal attempted Python part 1



```python
results = collection.aggregate(pipeline)

# Prepare data for plotting
years = []
avg_fg3a = []

for result in results:
    years.append(result["_id"])  # Year
    avg_fg3a.append(result["avgFg3a"])  # Average fg3a for that year

# Create a plot
plt.figure(figsize=(10,6))
plt.plot(years, avg_fg3a, marker='o', linestyle='-', color='b')
plt.title('Average 3-Point Field Goal Attempts (fg3a) Per Year')
plt.xlabel('Year')
plt.ylabel('Average fg3a')
plt.grid(True)
plt.xticks(years, rotation=45)
plt.tight_layout()

# Show the plot
plt.show()
```

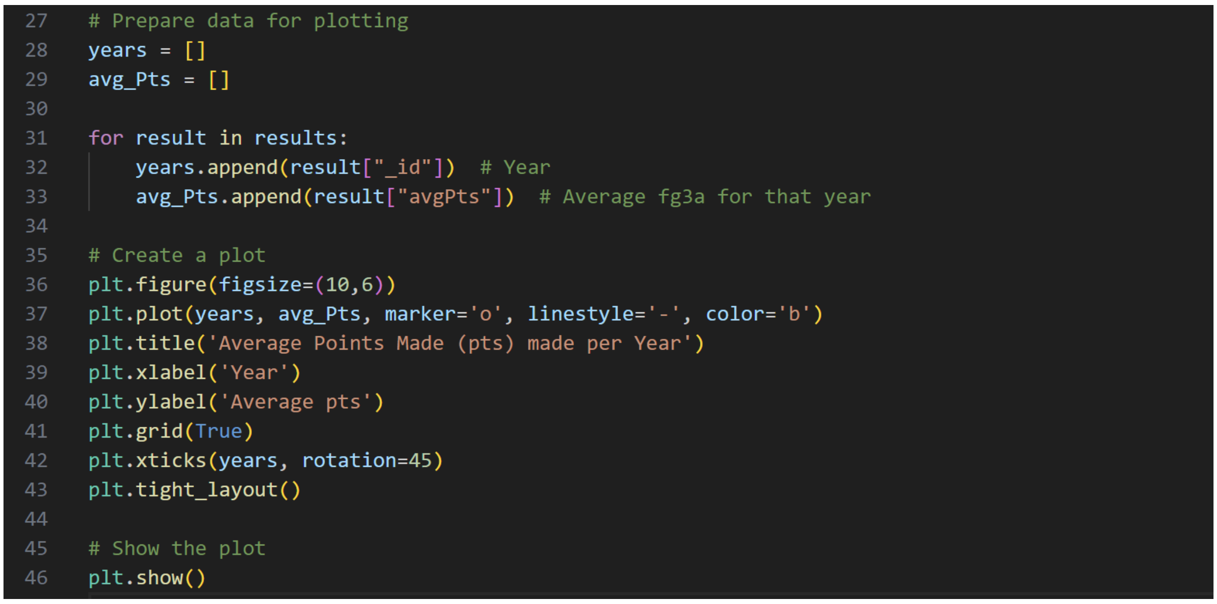
Figure 4.16: Three-Point Field Goal attempted Python part 2

Average Points made per year Python code



```python
import pymongo
import matplotlib.pyplot as plt

# Connect to MongoDB
client = pymongo.MongoClient("mongodb://localhost:27017/")
db = client["nba"]
collection = db["games"]

# Run the aggregation query
pipeline = [
    {
        "$unwind": "$box"  # Unwind the 'box' array to process each team separately
    },
    {
        "$group": {
            "_id": { "$year": "$date" },  # Extract the year from the 'date' field
            "avgPts": { "$avg": "$box.team.pts" }  # Calculate average fg3a per year
        }
    },
    {
        "$sort": { "_id": 1 }  # Sort by year in ascending order
    }
]

results = collection.aggregate(pipeline)

# Prepare data for plotting
```

Figure 4.17: Average Points made per year Python part 1



```python
# Prepare data for plotting
years = []
avg_Pts = []

for result in results:
    years.append(result["_id"])  # Year
    avg_Pts.append(result["avgPts"])  # Average fg3a for that year

# Create a plot
plt.figure(figsize=(10,6))
plt.plot(years, avg_Pts, marker='o', linestyle='-', color='b')
plt.title('Average Points Made (pts) made per Year')
plt.xlabel('Year')
plt.ylabel('Average pts')
plt.grid(True)
plt.xticks(years, rotation=45)
plt.tight_layout()

# Show the plot
plt.show()
```

Figure 4.18: Average Points made per year Python part 2

List of Figures

List of Tables