

Entrega 3



Francisco Ricardo Teixeira Silva
a103807



José António Almeida Vieira
a103899



Leonel José Coelho Pinheiro
a103716



Margarida dos Santos Ferreira
a100590

Índice

MYSQL	4
1.1 Dataset escolhido, descrição e links	4
1.2.1 Modelo Relacional.....	5
1.3 Dicionário de Dados - NBA Games Data (2004 – 2020)	6
1.4 Questões Analíticas.....	12
1.5 Respostas às questões analíticas	13
Cassandra.....	15
a. Carregamento em MYSQL – Processo de OUTFILE	15
b. Criação da tabela em Cassandra	15
c. Exportação de dados para o Cassandra.....	15
d. Query de resposta a cada questão analítica.....	15
Pergunta 1:.....	16
Pergunta 2:.....	17
Pergunta 3:.....	18
Pergunta 4:.....	19
Neo4j.....	20
a. Modelo de grafos de base de dados com os devidos relacionamentos	20
b. Carregamento dos dados e código utilizado	21
c. Resposta às questões analíticas.....	23
Pergunta 1 (Atualizada):	23
Pergunta 2 (Atualizada):	24
Pergunta 3 (Atualizada)	25
Pergunta 4 (Atualizada):	26
Comparação de respostas MySQL, Cassandra e Neo4j.....	27
Comparação entre as tecnologias MySQL, Cassandra e Neo4j	27
Conclusão	28

Introdução

No contexto da disciplina de Base de Dados II, o presente trabalho propõe-se a explorar e aplicar conceitos fundamentais relacionados à modelagem relacional e conceitual de dados. Para tal, optamos por utilizar um conjunto de dados centrado em NBA, uma liga renomeada mundialmente no cenário desportivo. Assim, esta entrega final visa a elaboração de um relatório final onde, em grupo, nos foi solicitado a realização de modelo relacional e conceptual e ainda descrição detalhada do processo de criação de tabelas de dados, a subsequente importação dos mesmos, bem como a execução de consultas e as suas queries em três distintos sistemas de gestão de base de dados (SGBD): **MySQL**, **Cassandra** e **Neo4j**.

O MySQL trata-se de um SGBD relacional amplamente adotado na indústria, conhecido pela sua robustez, confiabilidade e eficiência no processamento de transações. Em contraste, o Cassandra é uma base de dados distribuído altamente escalável, projetado para atender às exigências de alta disponibilidade e desempenho em ambientes distribuídos. Por fim, o Neo4j, trata-se de um SGBD orientado a grafos, particularmente eficaz na modelagem e consulta de relacionamentos complexos entre dados.

Neste relatório, procuramos explorar diversas abordagens e funcionalidades no contexto do armazenamento e manipulação de dados. Para tal, serão apresentados exemplos práticos e análises comparativas, enfatizando as diferenças de funcionalidades e desempenho entre os sistemas mencionados acima.

MYSQL

No desenvolvimento deste trabalho, iremos utilizar o MYSQL. Trata-se de um sistema de gestão de base de dados relacional amplamente utilizado, uma vez que é de confiança, com um desempenho eficiente e fácil integração com várias aplicações. Desenvolvido como um código aberto, o MYSQL utiliza a linguagem SQL para gerir e consultar dados, permitindo encontrar soluções robustas.

1.1 Dataset escolhido, descrição e links

Dataset escolhido: NBA Games Data (2004 – 2020)

Descrição: O conjunto de dados NBA Games Data fornece uma compilação abrangente de informações relacionadas a todos os jogos da NBA da temporada de 2004 até dezembro de 2020. Este dataset, disponível no Kaggle, consiste em vários arquivos CSV, em que cada um aborda um aspeto específico da liga, incluindo resultados de jogos, informações sobre equipas, estatísticas de jogadores e estatísticas históricas relevantes. Estes arquivos oferecem insights detalhados para análises e explorações profundas sobre o período mencionado.

Data Explorer

Version 10 (112.9 MB)

- games.csv
- games_details.csv
- players.csv
- ranking.csv
- teams.csv

Link: <https://www.kaggle.com/datasets/nathanlauga/nba-games?select=teams.csv>

1.2.1 Modelo Relacional

Games (game_date_est , game_id , game_status_text, home_team_id, visitor_team_id, season, PTS_home, FG_PCT_home, FT_PCT_home, FG3_PCT_home, AST_home, REB_home, PTS_away, FG_PCT_away, FT_PCT_away, FG3_PCT_away, AST_away, REB_away, home_team_wins)

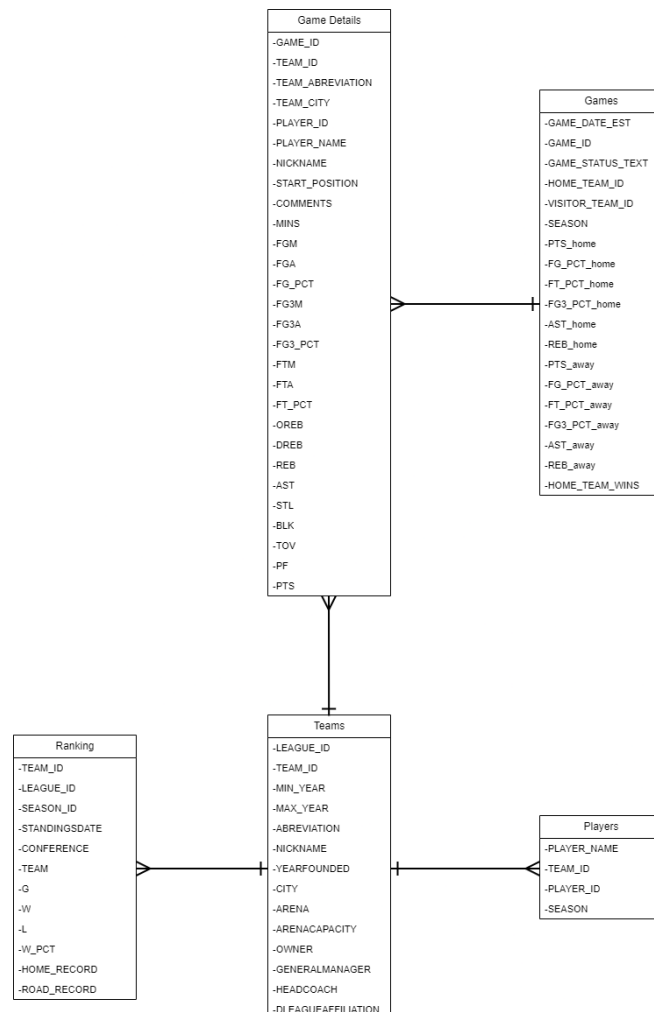
Teams (league_id, team_id, min_year, max_year, abbreviation, nickname, yearfounded, city, arena, arena_capacity, owner, general_manager, headcoach, dleagueaffiliation)

Players (player_name, team_id, player_id, season)

Ranking (team_id, league_id, season_id, standingsdate, conference, team, G, W, L, W_PCT, home_record, road_record)

Game Details (game_id, team_id, team_abbreviation, team_city, player_id, player_name, nickname, start_position, comments, mins, FGM, FGA, FG_PCT, FG3M, FG3A, FG3_PCT, FTM, AST, STL, BLK, TO, PF, PTS)

1.2.2 Modelo Conceptual



1.3 Dicionário de Dados - NBA Games Data (2004 – 2020)

Games:

Todos os jogos da temporada de 2004 até à data da última atualização e alguns detalhes como o número de pontos, etc.

Nome	Descrição	Tipo
GAME_DATE_EST	Data do jogo	Date
GAME_ID	Identificação única do jogo	Int
HOME_TEAM_ID	Identificação única da equipa que joga como visitante	Int
VISITOR_TEAM_ID	Identificação única da equipa que joga como visitante	Int
SEASON	Temporada em que ocorreu certo jogo	Int
PTS_home	Pontos da equipa visitada	Int
FG_PCT_home	Percentagem de arremessos da equipa visitada	Decimal (7,5)
FT_PCT_home	Percentagem de lançamentos livres da equipa visitada	Decimal (7,5)
FG3_PCT_away	Percentagem de lançamentos de três pontos da equipa da casa	Decimal (7,5)
AST_home	Assistências da equipa da casa	Int
REB_home	Ressaltos da equipa da casa	Int
PTS_away	Número de pontos marcados pela equipa visitante	Int

FG_PCT_away	Percentagem de lançamentos de quadra da equipa visitante	Decimal (7,5)
FT_PCT_away	Percentagem de lances livres convertidos pela equipa visitante	Decimal (7,5)
FG3_PCT_away	Percentagem de lançamentos de três pontos da equipa visitante	Decimal (7,5)
AST_away	Assistências da equipa visitante	Int
REB_away	Ressaltos da equipa visitante	Int
HOME_TEAM_WINS	Equipa da casa venceu a partida	Int

Game Details:

Detalhes relativos aos jogos de NBA bem como, às estatísticas dos jogadores.

Nome	Descrição	Tipo
GAME_ID	Identificação única do jogo	Int
TEAM_ID	Identificação única da equipa	Int
TEAM_ABBREVIATION	Abreviação do nome da equipa	Varchar(5)
TEAM_CITY	Cidade onde o jogo foi disputado	Varchar(20)
PLAYER_ID	Identificação única do jogador	Int
PLAYER_NAME	Nome do jogador	Varchar(40)
NICKNAME	Apelido do jogador	Varchar(20)

START_POSITION	Posição na equipa titular (caso esteja vazio, este começou o jogo no banco)	Varchar(4)
COMMENTS	Comentário sobre o facto de o jogador ter ou não jogado um determinado jogo	Varchar(40)
MINS	Minutos jogados	Varchar(35)
FGM	Cestos convertidos	Decimal (7,5)
FGA	Tentativas de cestos	Decimal (7,5)
FG_PCT	Percentual de acerto de cestos	Decimal (7,5)
FG3M	Cestos de três pontos convertidos	Decimal (7,5)
FG3A	Tentativas de três pontos	Decimal (7,5)
FG3_PCT	Percentagem de acerto de três pontos	Decimal (7,5)
FTM	Lances livres convertidos	Decimal (7,5)
FTA	Tentativas de lances livres	Decimal (7,5)
FT_PCT	Percentagem de acerto de lances livres	Decimal (7,5)
OREB	Ressaltos ofensivos	Decimal (7,5)
DREB	Ressaltos defensivos	Decimal (7,5)
REB	Total de ressaltos	Decimal (7,5)

AST	Total de assistências	Decimal (7,5)
STL	Roubos de bola	Decimal (7,5)
BLK	Bloqueios de bola	Decimal (7,5)
TO	Perdas da posse de bola	Decimal (7,5)
PF	Número de faltas (por jogador)	Decimal (7,5)
PTS	Número de pontos (por jogador)	Decimal (7,5)

Ranking: Posição relativa de uma equipa em relação às outras em termos de desempenho. Pode ser na classificação geral da liga ou em categorias específicas.

Nome	Descrição	Tipo
TEAM_ID	Número único para cada equipa de NBA	Int
LEAGUE_ID	Identificação única para a liga de NBA	Int
STANDINGSTATE	Data que as estatísticas de classificação são registadas	Varchar (15)
SEASON_ID	Identificação única para cada temporada de NBA	Int
CONFERENCE	A conferência à qual a equipa pertence	Varchar (15)
TEAM	Nome da equipa	Varchar (30)
G	Número de jogos jogados pela equipa	Int

W	Número de jogos ganhos pela equipa	Int
L	Número de jogos perdidos pela equipa	Int
W_PCT	A percentagem de vitórias da equipa	Decimal (7,5)
HOME_RECORD	Registo do desempenho da equipa em jogos disputados em casa	Varchar (10)
ROAD_RECORD	Registo do desempenho da equipa em jogos disputados fora de casa	Varchar (10)

Players:

Jogadores de NBA no período definido.

Nome	Descrição	Tipo
Player_name	Nome do jogador de basquetebol, que funciona como a identificação única do atleta	Varchar (40)
Team_id	Identificação única da equipa à qual o jogador está associado durante uma temporada	Int
Player_id	Identificador único atribuído a cada jogador de NBA, tratando-se de um código exclusivo, de forma a que não haja confusão com nomes iguais.	Int
Season	Temporada específica a que os dados se referem	Int

Teams:

Detalhes relativos a cada uma das equipas que participaram na NBA.

Nome	Descrição	Tipo
LEAGUE_ID	Identificação única da liga	Int
TEAM_ID	Identificação única da equipa	Int
MIN_YEAR	Primeiro ano em que a equipa esteve na NBA	Int
MAX_YEAR	Último ano em que a equipa esteve na NBA	Int
ABBREVIATION	Abreviação do nome da equipa	Varchar(5)
NICKNAME	Apelido da equipa	Varchar(20)
YEARFOUNDED	Ano de fundação	Int
CITY	Nome da cidade da equipa	Varchar(20)
ARENA	Nome da arena/estádio da equipa	Varchar(40)
ARENACAPACITY	Capacidade da arena/estádio	Int
OWNER	Proprietário da equipa (último proprietário)	Varchar(40)
GENERALMANAGER	Gerente geral	Varchar(40)

HEADCOACH	Treinador principal	Varchar(40)
DLEAGUEAFFILIATION	Afiliação à liga (Cada equipa da NBA pode ter uma equipa afiliada na G-League)	Varchar(40)

1.4 Questões Analíticas

Pergunta 1: Qual foi a equipa cujo ano de fundação é superior a 1950, apresentou um maior número de vitórias em 2022?

Pergunta 2: Qual foi a média de ressaltos do LeBron James na temporada de 2019?

Pergunta 3: Quais foram os três jogadores que tiveram o melhor desempenho em termos de pontuação (PTS), rebotes (REB) e assistências (AST) em jogos realizados na temporada de 2020?

Pergunta 4: Quantos jogos realizados em Los Angeles tiveram mais do que 200 pontos e 40 rebounds?

1.5 Respostas às questões analíticas

Pergunta 1: Qual foi a equipa cujo ano de fundação é superior a 1950, apresentou um maior número de vitórias em 2022?

```
SELECT t.TEAM_ID, t.NICKNAME, r.W
FROM Teams t
JOIN Ranking r ON t.TEAM_ID = r.TEAM_ID
WHERE t.YEARFOUNDED > 1950 AND r.SEASON_ID = 22022
ORDER BY r.W DESC
LIMIT 1;
```

Resultado:

	TEAM_ID	NICKNAME	W
1	1,610,612,739	Cavaliers	22

Pergunta 2: Qual foi a média de ressaltos do LeBron James na temporada de 2019?

```
SELECT AVG(gd.REB) AS MEDIA_RESSALTOS
FROM Game_Details gd
JOIN Players p ON gd.PLAYER_ID = p.PLAYER_ID AND gd.TEAM_ID = p.TEAM_ID
JOIN Games g ON gd.GAME_ID = g.GAME_ID
WHERE p.PLAYER_NAME = 'LeBron James'
AND g.SEASON = 2019;
```

Resultado:

	MEDIA_RESSALTOS
1	7.888888888

Pergunta 3: Quais foram os três jogadores que tiveram o melhor desempenho em termos de pontuação (PTS), rebotes (REB) e assistências (AST) em jogos realizados na temporada de 2020?

```
SELECT p.PLAYER_NAME, SUM(gd.PTS) AS TOTAL_PONTOS, SUM(gd.REB) AS TOTAL_REBOTES, SUM(gd.AST) AS TOTAL_ASSISTENCIAS
FROM Players p
JOIN Game_Details gd ON p.PLAYER_ID = gd.PLAYER_ID AND p.TEAM_ID = gd.TEAM_ID
JOIN Games g ON gd.GAME_ID = g.GAME_ID
WHERE g.SEASON = 2020
GROUP BY p.PLAYER_NAME
ORDER BY TOTAL_PONTOS DESC, TOTAL_REBOTES DESC, TOTAL_ASSISTENCIAS DESC
LIMIT 3;
```


Resultado:

	PLAYER_NAME	TOTAL_PONTOS	TOTAL_REBOTES	TOTAL_ASSISTENCIAS
1	Stephen Curry	23,749	4,037	4,191
2	Damian Lillard	17,328	2,520	4,608
3	Giannis Antetokounmpo	16,800	6,741	3,290

Pergunta 4: Quantos jogos realizados em Los Angeles tiveram mais do que 200 pontos e 40 rebounds?

```
SELECT COUNT(*) AS TOTAL_JOGOS
FROM Games g
JOIN Teams home_team ON g.HOME_TEAM_ID = home_team.TEAM_ID
JOIN Teams visitor_team ON g.VISITOR_TEAM_ID = visitor_team.TEAM_ID
JOIN (
    SELECT GAME_ID, (PTS_home + PTS_away) AS TOTAL_PONTOS,
           (REB_home + REB_away) AS TOTAL_REBOUNDS
    FROM Games
) AS TotalPontosRebounds ON g.GAME_ID = TotalPontosRebounds.GAME_ID
WHERE (home_team.CITY = 'Los Angeles' OR visitor_team.CITY = 'Los Angeles')
AND (TotalPontosRebounds.TOTAL_PONTOS > 200 AND TotalPontosRebounds.TOTAL_REBOUNDS > 40);
```

Resposta:

	 123 TOTAL_JOGOS	
1	2,053	

Cassandra

No desenvolvimento deste trabalho, iremos utilizar o Cassandra.

Trata-se de um sistema de gestão de base de dados projetado para lidar com grandes volumes de dados em várias máquinas sem um único ponto de falha. Desenvolvido para ser amplamente escalável e disponível, fornece um bom desempenho para aplicações que precisam de armazenar e aceder a grandes quantidades de dados de forma rápida.

a. Carregamento em MYSQL – Processo de OUTFILE

Através do processo de OUTFILE, procedemos ao carregamento de dados iniciais da NBA num banco de dados MySQL, de maneira a exportar os dados de um arquivo externo para uma tabela do MySQL.

b. Criação da tabela em Cassandra

Nesta etapa, desenvolvemos um esquema de base de dados no Cassandra e criamos as tabelas necessárias para armazenar os dados de NBA de forma prática e eficiente.

c. Exportação de dados para o Cassandra

Nesta etapa, procedemos à exportação dos dados previamente carregados no MySQL para o Cassandra, garantindo a consistência e integridade dos dados durante o processo de transferência.

d. Query de resposta a cada questão analítica

Pergunta 1: Qual foi a equipa cujo ano de fundação é superior a 1950 que apresentou um maior número de vitórias em 2022?

a. Processo de Outfile

```
-- Questao analitica 1
SELECT Ranking.W, Teams.TEAM_ID, Teams.NICKNAME, Teams.YEARFOUNDED, Ranking.SEASON_ID
FROM Teams
JOIN Ranking ON Teams.TEAM_ID = Ranking.TEAM_ID
WHERE Teams.YEARFOUNDED > 1950 AND Ranking.SEASON_ID = 2022
ORDER BY Ranking.W DESC
INTO OUTFILE '/var/lib/mysql/csv/nbaquestao1.csv'
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n';
```

b. Criação da tabela 1

```
●CREATE TABLE IF NOT EXISTS nbaquestao1 (
  W INT,
  TEAM_ID INT,
  NICKNAME VARCHAR,
  YEARFOUNDED INT,
  SEASON_ID INT,
  PRIMARY KEY ((SEASON_ID, W), TEAM_ID)
);
```

c. Exportação de dados para o Cassandra

```
COPY nbaquestao1 (W, TEAM_ID, NICKNAME, YEARFOUNDED, SEASON_ID)
FROM '/var/lib/cassandra/csv/nbaquestao1.csv' WITH DELIMITER=',';
```

d. Query de resposta

```
-- 1.Qual foi a equipa cujo ano de fundação é superior a 1950, apresentou um maior número de vitórias em 2022?
SELECT TEAM_ID, NICKNAME, W
FROM nbaquestao1
LIMIT 1;
```

e. Resultados da query

Grd	team_id	nickname	w
1	1,610,612,739	Cavaliers	22

Pergunta 2: Qual foi a média de ressaltos de LeBron James na temporada de 2019?

a. Processo de Outfile

```
-- Questao analitica 2
SET @row_number = 0;

SELECT (@row_number:=@row_number + 1), gd.REB, p.PLAYER_NAME, g.SEASON
FROM Game_Details gd
JOIN Players p ON gd.PLAYER_ID = p.PLAYER_ID AND gd.TEAM_ID = p.TEAM_ID
JOIN Games g ON gd.GAME_ID = g.GAME_ID
INTO OUTFILE '/var/lib/mysql/csv/nbaquestao2.csv'
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n';
```

b. Criação da tabela 2

```
CREATE TABLE IF NOT EXISTS nbaquestao2 (
  NEW_ID INT,
  REB FLOAT,
  PLAYER_NAME VARCHAR,
  SEASON INT,
  PRIMARY KEY (NEW_ID)
);
```

c. Exportação de dados para o Cassandra

```
COPY nbaquestao2 (NEW_ID, REB, PLAYER_NAME, SEASON)
FROM '/var/lib/cassandra/csv/nbaquestao2.csv' WITH DELIMITER=',';
```

d. Query de resposta

```
-- 2.Qual a média de ressaltos do LeBron James na temporada de 2019?
-- Calcular o total de ressaltos
SELECT SUM(REB) AS total_reb
FROM nbaquestao2
WHERE PLAYER_NAME = 'LeBron James' AND SEASON = 2019
allow filtering;

-- Contar o numero de jogos
SELECT COUNT(*) AS num_jogos
FROM nbaquestao2
WHERE PLAYER_NAME = 'LeBron James' AND SEASON = 2019
allow filtering;
```

e. Resultados da query

SELECT SUM(REB) AS total_reb FROM nbaquestao2 WHERE	
Grid	123 total_reb
1	1,562

SELECT COUNT(*) AS num_jogos FROM nbaquestao2 WHERE	
Grid	123 num_jogos
1	198

Uma vez que não é possível utilizar o avg (media) no Cassandra, utilizamos duas queries separadas e obtivemos dois resultados, apresentados ao lado. Assim, $1562/198 = 7.8$

Pergunta 3: Quais foram os três jogadores que tiveram o melhor desempenho em termos de pontuação (PTS), rebotes (REB) e assistências (AST) em jogos realizados na temporada de 2020?

a. Processo de Outfile

```
-- Questao analitica 3
SELECT p.PLAYER_NAME, SUM(gd.PTS) AS TOTAL_PONTOS, SUM(gd.REB) AS TOTAL_REBOTES, SUM(gd.AST) AS TOTAL_ASSISTENCIAS
FROM Players p
JOIN Game_Details gd ON p.PLAYER_ID = gd.PLAYER_ID AND p.TEAM_ID = gd.TEAM_ID
JOIN Games g ON gd.GAME_ID = g.GAME_ID
WHERE g.SEASON=2020
GROUP BY p.PLAYER_NAME
ORDER BY TOTAL_PONTOS DESC, TOTAL_REBOTES DESC, TOTAL_ASSISTENCIAS DESC
LIMIT 3
INTO OUTFILE '/var/lib/mysql/csv/nbaquestao3.csv'
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n';
```

b. Criação da tabela 3

```
CREATE TABLE IF NOT EXISTS nbaquestao3 (
  PLAYER_NAME VARCHAR,
  TOTAL_PONTOS FLOAT,
  TOTAL_REBOTES FLOAT,
  TOTAL_ASSISTENCIAS FLOAT,
  PRIMARY KEY (PLAYER_NAME)
);
```

c. Exportação de dados para o Cassandra

```
COPY nbaquestao3 (PLAYER_NAME, TOTAL_PONTOS, TOTAL_REBOTES, TOTAL_ASSISTENCIAS)
FROM '/var/lib/cassandra/csv/nbaquestao3.csv' WITH DELIMITER=',' AND HEADER=FALSE;
```

d. Query de resposta

```
-- 3.Quais foram os três jogadores que tiveram o melhor desempenho em termos de pontuação (PTS),
-- rebotes (REB) e assistências (AST) em jogos realizados na temporada de 2020?
SELECT PLAYER_NAME, TOTAL_PONTOS, TOTAL_ASSISTENCIAS, TOTAL_REBOTES
FROM nbaquestao3;
```

e. Resultado da query

	player_name	total_pontos	total_assistencias	total_rebotes
1	Damian Lillard	17,328	4,608	2,520
2	Giannis Antetokounmpo	16,800	3,290	6,741
3	Stephen Curry	23,749	4,191	4,037

Pergunta 4: Quantos jogos realizados em Los Angeles tiveram mais do que 200 pontos e 40 rebounds?

a. Processo de Outfile

```
●-- Questao analitica 4
SET @row_number = 0;

●SELECT (@row_number:=@row_number + 1), TotalPontosRebounds.TOTAL_PONTOS, TotalPontosRebounds.TOTAL_REBOUNDS
FROM Games g
JOIN Teams home_team ON g.HOME_TEAM_ID = home_team.TEAM_ID
JOIN Teams visitor_team ON g.VISITOR_TEAM_ID = visitor_team.TEAM_ID
JOIN (
  SELECT GAME_ID, (PTS_home + PTS_away) AS TOTAL_PONTOS,
  (REB_home + REB_away) AS TOTAL_REBOUNDS
  FROM Games
) AS TotalPontosRebounds ON g.GAME_ID = TotalPontosRebounds.GAME_ID
WHERE (home_team.CITY = 'Los Angeles' OR visitor_team.CITY = 'Los Angeles')
INTO OUTFILE '/var/lib/mysql/csv/nbaquestao4.csv'
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n';
```

b. Criação da tabela 4

```
●CREATE TABLE nbaquestao4 (
  ROW_ID INT,
  HOME_CITY VARCHAR,
  VISITOR_CITY VARCHAR,
  TOTAL_PONTOS INT,
  TOTAL_REBOUNDS INT,
  PRIMARY KEY (ROW_ID)
);
```

c. Exportação de dados para o Cassandra

```
●COPY nbaquestao4 (ROW_ID, TOTAL_PONTOS, TOTAL_REBOUNDS)
FROM '/var/lib/cassandra/csv/nbaquestao4.csv' WITH DELIMITER=',' AND HEADER=FALSE;
```

d. Query de resposta

```
●-- 4.Quantos jogos realizados em Los Angeles tiveram mais do que 200 pontos e 40 rebounds?
SELECT COUNT(ROW_ID) AS TOTAL_JOGOS
FROM nbaquestao4
WHERE TOTAL_PONTOS > 200 AND TOTAL_REBOUNDS > 40
allow filtering;
```

e. Resultado da query

```
HT SELECT COUNT(ROW_ID) AS TOTAL_JOGOS FROM nbaquest
```

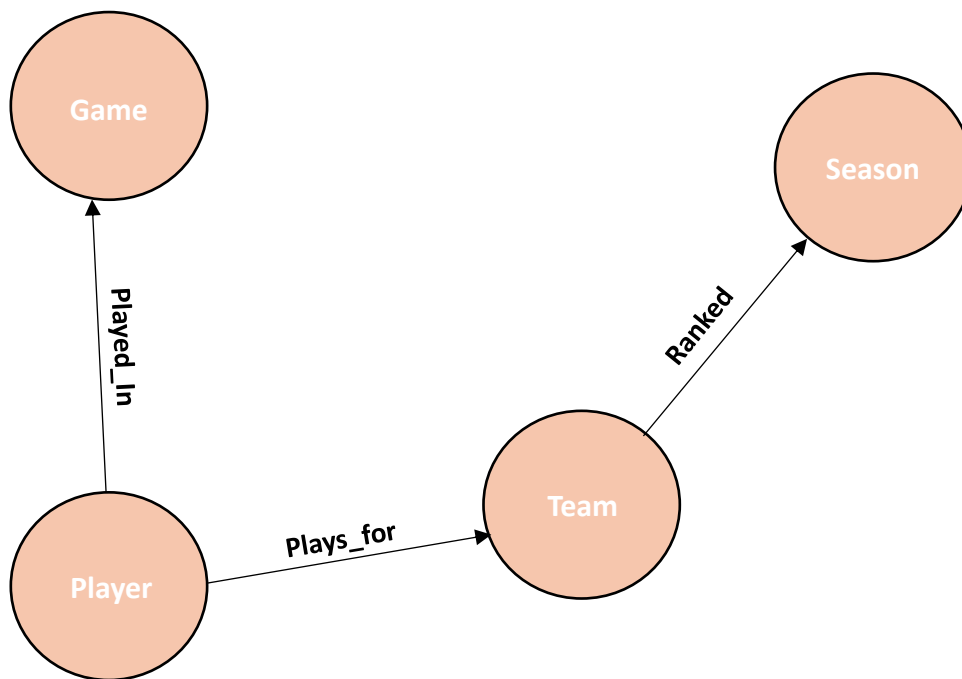
	total_jogos	
1	2,053	

Neo4j

No desenvolvimento deste trabalho, iremos utilizar o Neo4j.

Trata-se de um sistema de gestão de base de dados orientado a grafos projetado para lidar com grandes volumes de dados, representando-os através de nós, arestas e propriedades. Desenvolvido para ser altamente escalável e disponível, o Neo4j apresenta um bom desempenho para aplicações que precisam de armazenar e aceder a dados complexos e interconectados de forma rápida e eficiente.

a. Modelo de grafos de base de dados com os devidos relacionamentos



b. Carregamento dos dados e código utilizado

Para o carregamento dos dados, começamos por criar três nós, Game (Jogo), Team (Equipa) e Player (Jogador), carregando os respectivos dados das tabelas games, teams e players. No caso do nó Player, usamos o MERGE para evitar valores duplicados e estabelecer a relação PLAYS_FOR entre os nós Player e Team. Após isso criamos os INDEX para cada um desses nós.

```
LOAD CSV WITH HEADERS FROM 'file:///games.csv' AS row
FIELDTERMINATOR ','
CREATE (:Game {
    game_date_est: row.GAME_DATE_EST,
    game_id: toInteger(row.GAME_ID),
    home_team_id: toInteger(row.HOME_TEAM_ID),
    visitor_team_id: toInteger(row.VISITOR_TEAM_ID),
    season: toInteger(row.SEASON),
    pts_home: toInteger(row.PTS_home),
    pts_away: toInteger(row.PTS_away),
    home_team_wins: toInteger(row.HOME_TEAM_WINS)
});

LOAD CSV WITH HEADERS FROM 'file:///teams.csv' AS row
FIELDTERMINATOR ','
CREATE (:Team {
    league_id: toInteger(row.LEAGUE_ID),
    team_id: toInteger(row.TEAM_ID),
    min_year: toInteger(row.MIN_YEAR),
    max_year: toInteger(row.MAX_YEAR),
    abbreviation: row.ABBREVIATION,
    nickname: row.NICKNAME,
    year_founded: toInteger(row.YEARFOUNDED),
    city: row.CITY,
    arena: row.ARENA,
    arena_capacity: toInteger(row.ARENACAPACITY),
    owner: row.OWNER
});

LOAD CSV WITH HEADERS FROM 'file:///players.csv' AS row
FIELDTERMINATOR ','
MERGE (t:Team {team_id: toInteger(row.TEAM_ID)})
CREATE (p:Player {
    player_name: row.PLAYER_NAME,
    player_id: toInteger(row.PLAYER_ID),
    season: toInteger(row.SEASON)
})
CREATE (p)-[:PLAYS_FOR]->(t);

// Index
CREATE INDEX game_id_index FOR (g:Game) ON (g.game_id);
CREATE INDEX team_id_index FOR (t:Team) ON (t.team_id);
CREATE INDEX player_id_index FOR (p:Player) ON (p.player_id);
```

Depois da criação de INDEX, criamos uma relação RANKED entre os nós Team e Season que também criamos. Usamos outra vez o MERGE e carregamos os dados da tabela ranking para os atributos da relação RANKED e também do nó Season.

```
LOAD CSV WITH HEADERS FROM 'file:///ranking_part1.csv' AS row
FIELDTERMINATOR ';'
MERGE (t:Team {team_id: toInteger(row.TEAM_ID)})
CREATE (t)-[:RANKED {
    league_id: toInteger(row.LEAGUE_ID),
    season_id: toInteger(row.SEASON_ID),
    standings_date: row.STANDINGS_DATE,
    conference: row.CONFERENCE,
    team: row.TEAM,
    games: toInteger(row.G),
    wins: toInteger(row.W),
    losses: toInteger(row.L)
}]->(s:Season {season_id: toInteger(row.SEASON_ID)});
```

Por fim criamos mais um relacionamento, neste caso o PLAYED_IN entre Player e Game. Usamos os MATCH para conectar as linhas do csv games_details a dados já presentes na base de dados e assim evitar valores duplicados. Os dados da tabela foram carregados para os atributos da relação PLAYED_IN.

```
LOAD CSV WITH HEADERS FROM 'file:///games_details_part1.csv' AS row
FIELDTERMINATOR ';'
MATCH (g:Game {game_id: toInteger(row.GAME_ID)})
MATCH (t:Team {team_id: toInteger(row.TEAM_ID)})
MATCH (p:Player {player_id: toInteger(row.PLAYER_ID)})
CREATE (p)-[:PLAYED_IN {
    team_city: row.TEAM_CITY,
    name: row.PLAYER_NAME,
    reb: toFloat(row.REB),
    ast: toFloat(row.AST),
    pts: toFloat(row.PTS)
}]->(g);
```

O carregamento das duas últimas tabelas teve de ser dividido, visto que o servidor Neo4j “caía” ao tentar carregar estas tabelas de maior dimensão. No caso da tabela ranking foi dividida em duas partes enquanto a games_details teve de ser dividida em vinte e uma partes.

c. Resposta às questões analíticas

Pergunta 1 (Atualizada): Qual foi a equipa cujo ano de fundação é superior a 1950, apresentou um maior número de vitórias em 2022?

```
1 MATCH (t:Team)-[r:RANKED]→(:Season {season_id: 22022})
2 WHERE t.year_founded > 1950
3 RETURN DISTINCT t.team_id, t.nickname, r.wins
4 ORDER BY r.wins DESC
5 LIMIT 2;
```

	t.team_id	t.nickname	r.wins
1	1610612749	"Bucks"	22
2	1610612739	"Cavalliers"	22

```
-- 1.Qual foi a equipa cujo ano de fundação é superior a 1950, apresentou um maior número de vitórias em 2022?
SELECT DISTINCT t.TEAM_ID, t.NICKNAME, r.W
FROM Teams t
JOIN Ranking r ON t.TEAM_ID = r.TEAM_ID
WHERE t.YEARFOUNDED > 1950 AND r.SEASON_ID = 22022
ORDER BY r.W DESC
LIMIT 2;
```

SELECT DISTINCT t.TEAM_ID, t.NICKNAME, r.W FROM Team

	TEAM_ID	NICKNAME	W
1	1,610,612,739	Cavaliers	22
2	1,610,612,749	Bucks	22

Nota: Neste caso, a pergunta é a mesma, mas tivemos de atualizar pois na realidade existem duas equipas com o mesmo número de vitórias em 2022.

Pergunta 2 (Atualizada): Qual foi a média de ressaltos do LeBron James na temporada de 2021?

```
1 MATCH (p:Player {player_name: 'LeBron James'})-[j:PLAYED_IN]-(g:Game)
2 WHERE g.season = 2021
3 RETURN AVG(toFloat(j.reb)) AS media_ressaltos;
```

media_ressaltos	
1	7.181818181818181

```
-- 2.Qual a média de ressaltos do LeBron James na temporada de 2021?
SELECT AVG(gd.REB) AS MEDIA_RESSALTOS
FROM Game_Details gd
JOIN Players p ON gd.PLAYER_ID = p.PLAYER_ID AND gd.TEAM_ID = p.TEAM_ID
JOIN Games g ON gd.GAME_ID = g.GAME_ID
WHERE p.PLAYER_NAME = 'LeBron James'
AND g.SEASON = 2021;
```

SELECT AVG(gd.REB) AS MEDIA_RESSALTOS FROM Game_Details gd	
Grid	123 MEDIA_RESSALTOS
1	7.181818181

Pergunta 3 (Atualizada): Quais foram os três jogadores que tiveram o melhor desempenho em termos de pontuação (PTS), rebotes (REB) e assistências (AST) em jogos realizados na temporada de 2021?

```

1 MATCH (p:Player)-[j:PLAYED_IN]-(g:Game)
2 WHERE g.season = 2021
3 RETURN p.player_name AS nome_jogador,
4        SUM(j.pts) AS total_pontos,
5        SUM(j.reb) AS total_ressaltos,
6        SUM(j.ast) AS total_assistencias
7 ORDER BY total_pontos DESC, total_ressaltos DESC, total_assistencias DESC
8 LIMIT 1;

```

	nome_jogador	total_pontos	total_ressaltos	total_assistencias
1	"Stephen Curry"	25630.0	5148.0	5973.0

```

-- 3.Qual o jogador que teve o melhor desempenho em termos de pontuação (PTS),
-- rebotes (REB) e assistências (AST) em jogos realizados na temporada de 2021?
SELECT p.PLAYER_NAME, SUM(gd.PTS) AS TOTAL_PONTOS, SUM(gd.REB) AS TOTAL_REBOTES, SUM(gd.AST) AS TOTAL_ASSISTENCIAS
FROM Players p
JOIN Game_Details gd ON p.PLAYER_ID = gd.PLAYER_ID AND p.TEAM_ID = gd.TEAM_ID
JOIN Games g ON gd.GAME_ID = g.GAME_ID
WHERE g.SEASON = 2021
GROUP BY p.PLAYER_NAME
ORDER BY TOTAL_PONTOS DESC, TOTAL_REBOTES DESC, TOTAL_ASSISTENCIAS DESC
LIMIT 1;

```

⚙️ `SELECT p.PLAYER_NAME, SUM(gd.PTS) AS TOTAL_PONTOS` | Enter a SQL expression to filter results (use Ctrl+Space)

	ABC PLAYER_NAME	123 TOTAL_PONTOS	123 TOTAL_REBOTES	123 TOTAL_ASSISTENCIAS
1	Stephen Curry	25,630	5,148	5,973

Pergunta 4 (Atualizada): Quantos jogos realizados em Los Angeles em 2021 que tiveram mais do que 200 pontos e 40 assistências?

```

1 MATCH (g:Game)
2 WHERE g.season = 2021
3   AND g.pts_home + g.pts_away > 200
4 MATCH (home:Team {team_id: g.home_team_id}), (visitor:Team {team_id: g.visitor_team_id})
5 WHERE home.city = 'Los Angeles' OR visitor.city = 'Los Angeles'
6 WITH g
7 MATCH (p:Player)-[j:PLAYED_IN]->(g)
8 WITH g, SUM(j.ast) AS total_assistencias
9 WHERE total_assistencias > 40
10 RETURN COUNT(DISTINCT g.game_id) AS total_jogos;

```

total_jogos	
1	146

```

-- 4.Quantos jogos realizados em Los Angeles tiveram mais do que 200 pontos e 40 assistências?
SELECT COUNT(DISTINCT GAME_ID) AS num_games
FROM (
  SELECT GA.GAME_ID
  FROM (
    SELECT GD.GAME_ID,
           SUM(GD.AST) AS total_assists,
           (G.PTS_home + G.PTS_away) AS total_points
    FROM Game_Details GD
    JOIN Games G ON GD.GAME_ID = G.GAME_ID
    WHERE G.SEASON = 2021
    GROUP BY GD.GAME_ID, G.PTS_home, G.PTS_away
  ) AS GA
  JOIN Games G ON GA.GAME_ID = G.GAME_ID
  JOIN Teams TH ON G.HOME_TEAM_ID = TH.TEAM_ID
  JOIN Teams TV ON G.VISITOR_TEAM_ID = TV.TEAM_ID
  WHERE GA.total_points > 200
    AND GA.total_assists > 40
    AND (TH.CITY = 'Los Angeles' OR TV.CITY = 'Los Angeles')
  ) AS FG;

```

SELECT COUNT(GAME_ID) AS num_g

Grid	123 num_games
1	139

Nota: Estas últimas três questões tiveram de ser modificadas, relativamente aos anos, uma vez que o servidor Neo4j “caía”, constantemente, a tentar carregar tabelas maiores e, por isso, os dados mais fiáveis eram apenas de 2020/2021 para cima, sendo que esta última pergunta foi a única que não conseguimos obter o mesmo resultado.

Comparação de respostas MySQL, Cassandra e Neo4j:

Como podemos observar, nas duas primeiras tecnologias, MySQL e Cassandra, não tivemos necessidade de atualizar as questões, visto que ambas até foram usadas “juntas”, e conseguimos obter os mesmos resultados.

No caso do neo4j, tivemos de modificar as questões devido aos motivos anteriormente referidos, tendo de usar dados mais “recentes”. No entanto, após as modificações em MySQL e Neo4j, conseguimos obter os mesmos resultados em ambas as tecnologias, exceto a questão 4.

Em conclusão, todas as respostas deram iguais, exceto a pergunta 4.

Comparação entre as tecnologias MySQL, Cassandra e Neo4j

Ao longo do desenvolvimento destes trabalhos práticos, deparamo-nos com a necessidade de implementar diferentes tecnologias de base de dados tais como, MySQL, Cassandra e Neo4j, existindo entre todas elas algumas semelhanças e disparidades. Cada uma destas tecnologias oferece soluções distintas para o armazenamento e gestão de dados, adaptando-se a diferentes tipos de aplicações e requisitos específicos.

A primeira tecnologia com a qual trabalhamos, o MySQL, é uma base de dados relacional amplamente utilizada devido principalmente à sua estabilidade e suporte abrangente. O MySQL utiliza a linguagem SQL de forma a gerir e consultar dados estruturados em tabelas, sendo muito utilizado em aplicações que necessitam de relações bem definidas entre os dados. No entanto, o MySQL pode apresentar limitações em termos de escalabilidade vertical e na gestão de grandes volumes de dados não estruturados.

O Cassandra, por outro lado, é uma base de dados NoSQL, construída com o intuito de lidar com grandes volumes de dados distribuídos por vários servidores. Esta utiliza a linguagem CQL, semelhante ao SQL. Além disso, o Cassandra permite a escalabilidade horizontal, facilitando a adição de novos nós. Sendo assim, esta tecnologia é ideal para aplicações que exigem alta disponibilidade, possibilitando um desempenho bastante eficiente. No entanto, a configuração e manutenção de um sistema distribuído do Cassandra pode ser complexa e exigir mais conhecimentos, além do facto do seu modelo de dados ser mais rígido em comparação com outras bases de dados NoSQL, apresentando uma menor flexibilidade.

Por último, usamos o Neo4j, que é uma base de dados que utiliza um modelo de grafos para representar e armazenar dados e utiliza a linguagem de consulta Cypher. Esta tecnologia é utilizada para consultas que envolvem várias relações entre dados, oferecendo um desempenho superior para tais situações, sendo ainda ideal para aplicações que requerem uma gestão eficiente dos dados interconectados. Contudo, embora o Neo4j tenha melhorado a sua escalabilidade, ainda não apresenta tanta eficiência quanto o Cassandra em cenários de grande escalabilidade horizontal.

Sendo assim, podemos concluir que cada uma destas tecnologias apresenta características específicas que as tornam mais ou menos adequadas para um determinado tipo de enquadramento. A escolha da base de dados deve ser guiada pelos requisitos específicos da aplicação, levando em conta fatores como a estrutura dos dados, a necessidade de escalabilidade, a complexidade e a facilidade de manutenção.

Conclusão

O modelo de gestão de bases de dados que nos apresentou maiores desafios durante o desenvolvimento do projeto foi o Neo4j, devido aos motivos previamente mencionados. A sua complexidade e as limitações inerentes exigiram um esforço significativo de pesquisa adicional para superarmos as dificuldades encontradas.

Em contrapartida, o MySQL foi o sistema com o qual melhor nos adaptamos e que mais apreciamos desenvolver. A sua interface intuitiva facilitou a nossa aprendizagem, permitindo-nos progredir rapidamente e sem grandes obstáculos. No caso deste, a experiência previamente adquirida na UC de Base de Dados I permitiu-nos implementar mais os nossos conhecimentos sobre armazenamento e manipulação de dados relacionais.

Em síntese, este projeto proporcionou-nos uma valiosa oportunidade de experimentar e compreender as vantagens e limitações de cada tecnologia de gestão de base de dados. Ao explorar as diferentes abordagens, fomos capazes de identificar as características que melhor se adequam a diferentes tipos de problemas, preparando-nos para fazer escolhas informadas e estratégicas em projetos futuros.