UNIVERSIDADE FEDERAL DE SÃO CARLOS

JOÃO VITOR AVERALDO ANTUNES - 813979

PEDRO ENRICO BARCHI NOGUEIRA - 813099

RAFAEL MORI PINHEIRO - 813851

Fase Intermediária I

Planejamento e Definição de Proposta de Projeto

SOROCABA

2025

SUMÁRIO

1. INTRODUÇÃO

O futebol moderno é uma indústria multibilionária, onde dados são um ativo estratégico para clubes, investidores e mídia. Dessa forma, decisões sobre contratações, táticas de jogo e desenvolvimento de atletas são cada vez mais orientadas por análises detalhadas. No entanto, a grande variedade e o volume massivo de dados – estatísticas de desempenho, valores de mercado, histórico de transferências, eventos de jogo – representam um desafio computacional significativo.

Este projeto prático propõe a construção de um pipeline de processamento de dados para analisar o ecossistema do futebol profissional. Assim sendo, utilizando um grande conjunto de dados do portal Transfermarkt, investigaremos padrões complexos relacionados à valorização de jogadores, ao impacto tático de escalações e à trajetória de carreira dos atletas. Por fim, para lidar com a escala e a complexidade dos dados, empregaremos uma arquitetura combinando o poder do processamento distribuído do Apache Spark com a flexibilidade de bancos de dados NoSQL, utilizando MongoDB, além de uma análise estrutural acerca das relações entre os jogadores utilizando Neo4j.

2. OBJETIVOS DO PROJETO

A princípio, o objetivo geral é desenvolver e implementar uma solução que envolva todas as etapas de ingestão, processamento, armazenamento e análise de dados de futebol em larga escala. Consequentemente, os resultados obtidos permitirão extrair insights valiosos que hoje são de difícil obtenção através de métodos tradicionais.

Desse modo, os objetivos centrais do nosso trabalho são divididos em três eixos de análise, que guiarão toda a implementação técnica:

2.1. Eixo 1: Evolução do Valor de Mercado e Impacto das Transferências

- Analisar a correlação entre a idade, o desempenho em campo (gols, assistências, minutos jogados) e a curva de valorização de um jogador.

- Identificar quais ligas ou clubes têm maior impacto na inflação do valor de mercado dos atletas.
- Calcular o Retorno sobre o Investimento (ROI) para clubes em transferências específicas, comparando o custo de aquisição com a evolução do valor e o desempenho subsequente do jogador.

2.2. Eixo 2: Impacto das Escalações e Substituições no Desempenho do Time

- Avaliar a eficácia de diferentes formações táticas (inferidas a partir das posições dos jogadores em campo) em termos de probabilidade de vitória, gols marcados e sofridos.
- Analisar o impacto das substituições: em que momento do jogo e em quais posições as substituições geram um impacto mais positivo
- Comparar a contribuição ofensiva (gols, assistências) de jogadores titulares versus jogadores que entram como substitutos.

2.3. Eixo 3: Análise da Carreira e Padrões de Trajetória de Jogadores

- Mapear e visualizar a trajetória de carreira completa de um jogador, identificando os clubes pelos quais passou e os períodos correspondentes.
- Identificar padrões de transferência comuns.
- Descobrir "clusters" de relacionamento entre clubes, ou seja, quais clubes mais negociam jogadores entre si, formando uma rede de transferências.

3. FONTE DE DADOS

Para atingir os objetivos propostos, utilizamos um único, porém bem populado, conjunto de dados disponível publicamente no Kaggle: **Player Scores from Transfermarkt.**

Este dataset contém múltiplos arquivos CSV inter-relacionados que cobrem mais de 30.000 jogadores, 400 clubes, 60.000 jogos e, crucialmente, 400.000 registros históricos de valor de mercado e 1.200.000 registros de participações em jogos.

4. TECNOLOGIAS ESCOLHIDAS E JUSTIFICATIVAS

Para atender aos requisitos do projeto e aos nossos objetivos analíticos, selecionamos uma arquitetura que combina o processamento dos dados com dois modelos de banco de dados NoSQL, o MongoDB e o Neo4j, utilizando o Apache Spark para o pré-processamento dos dados. Desse modo, a seguir estão justificadas as tecnologias citadas, assim como seus papéis na arquitetura do projeto:

4.1. Apache Spark

- **ETL:** O dataset é composto por diversos arquivos CSV. Utilizaremos o Spark (com PySpark) para ler, limpar, padronizar e unificar esses arquivos de forma eficiente. Portanto, operações como a conversão de tipos de dados, tratamento de valores nulos e a criação de novos atributos serão realizadas em paralelo sobre todo o conjunto de dados.
- **Agregações e Cálculos Complexos:** O Spark é capaz de realizar os cálculos pesados antes de carregar os dados nos bancos de dados de consulta. Por exemplo, para cada jogador, podemos usar o Spark para pré-calcular estatísticas de carreira (total de gols, média de minutos por jogo, desvio padrão do valor de mercado) que seriam computacionalmente caras para calcular em tempo real.
- **Joins em Larga Escala:** A necessidade de cruzar informações entre jogadores, seus valores de mercado, suas aparições em jogos e os

detalhes desses jogos (players + player_valuations + appearances + games) exige *joins* sobre um grande volume de dados. Logo, realizar essas operações em memória com a performance do Spark é mais eficiente do que sobrecarregar os bancos NoSQL com essa tarefa.

4.2. MongoDB

- **Modelagem do Perfil do Jogador:** Em um único documento no MongoDB, podemos armazenar todas as informações relevantes de um atleta: dados pessoais, estatísticas de carreira e, sua série temporal de valores de mercado e um histórico de suas principais atuações. Criando um perfil completo e autocontido, otimizado para consultas que pedem "todas as informações sobre o jogador X", eliminando a necessidade de *joins* no momento da leitura.
- **Estrutura de Jogos:** De forma similar, cada jogo pode ser um documento contendo seus dados principais (data, resultado, competição) e listas de eventos ou jogadores participantes.
- **Flexibilidade de Esquema:** Baseando-se nos pontos anteriormente apresentados, a flexibilidade de esquema torna-se um ponto crucial para a estruturação da informação, afinal precisamos da união da informação de maneira simplificada e direta, facilitando o processo de recuperação da mesma.

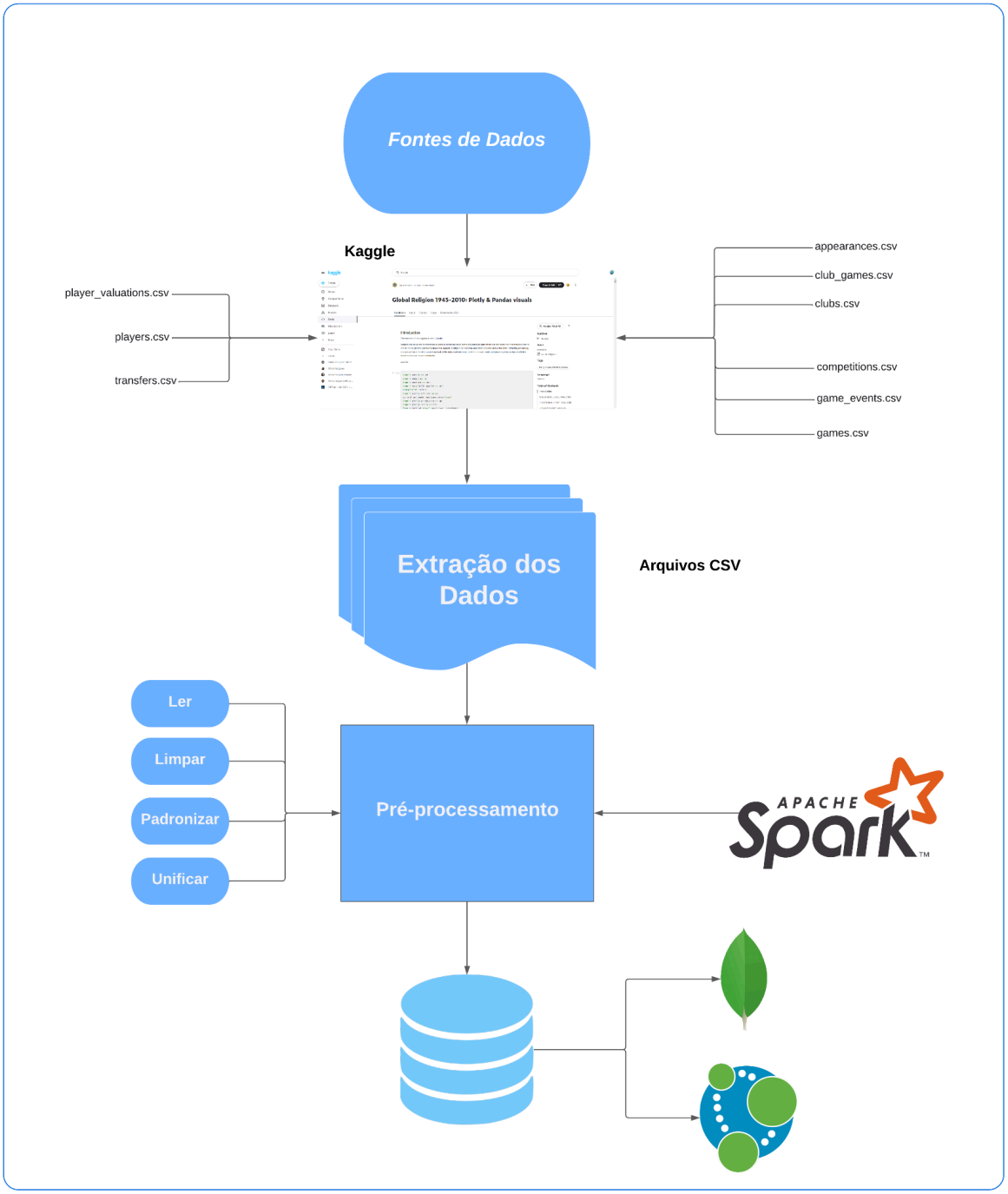
4.3. Neo4j

- **Análise de Trajetória e Transferências:** Podemos analisar a carreira de um jogador como um grafo de forma natural. Assim sendo, as entidades podem representar os nós (Jogador, Clube, País) e as conexões destes, podem ser vistas como as arestas (*jogou_por*, *foi_transferido_de_para*).
- **Análise de Padrões e Redes:** O modelo de grafos é mais adequado para encontrar padrões de relacionamentos entre entidades.

5. ARQUITETURA E FLUXOGRAMA DE DADOS

O fluxo de dados em nossa aplicação será linear e bem definido, garantindo que cada tecnologia seja usada em seu ponto de melhor atuação.

5.1. Fluxograma de Dados



5.2. Descrição do Fluxo

5.2.1. Ingestão e Processamento (Apache Spark)

- Os arquivos CSV do Kaggle são lidos pelo Apache Spark.
- O Spark executa o processo de ETL: limpeza de dados, padronização, junção das diferentes tabelas.
- São calculadas agregações complexas, como estatísticas de carreira por jogador.

5.2.2. Carga nos Bancos de Dados (Spark Connectors)

- Para o MongoDB: Os dados processados são modelados como documentos JSON. Por exemplo, para cada jogador, um documento único pode ser criado contendo seus dados e um array com seu histórico de valores. Esses documentos são carregados em coleções (players, games) usando o Spark Connector for MongoDB.
- Para o Neo4j: A partir dos mesmos dados processados, são criados os nós e as arestas que serão carregados no Neo4j usando o Spark Connector for Neo4j.

5.2.3. Análise e Consulta (MongoDB e Neo4j)

- As análises dos eixos 1 e 2 serão realizadas principalmente com consultas ao MongoDB, que é otimizado para recuperar documentos completos (perfis de jogadores, dados de jogos).
- As análises do eixo 3, focadas em relacionamentos e trajetórias, serão feitas com consultas Cypher no Neo4j.

6. FONTES E REFERÊNCIAS BIBLIOGRÁFICAS

- https://www.kaggle.com/datasets/davidcariboo/player-scores
- https://www.mongodb.com/docs/manual/
- https://www.mongodb.com/docs/spark-connector/current/
- https://neo4j.com/docs/
- https://neo4j.com/developer/spark/
- https://spark.apache.org/docs/latest/