ĐẠI HỌC BÁCH KHOA HÀ NỘI
HANOI UNIVERSITY OF SCIENCE AND TECHNOLOGY

# ĐỒ ÁN II

# XÂY DỰNG KHO DỮ LIỆU CHO PHÂN TÍCH BÓNG ĐÁ

SINH VIÊN THỰC HIỆN: NGUYỄN PHÚ VINH - 20227169
GIẢNG VIÊN HƯỚNG DẪN: PGS. TS. NGUYỄN ĐÌNH HÂN

Ngày 26 tháng 1 năm 2026

# Nội dung chính

# Nội dung chính

# Đặt vấn đề



**Bối cảnh:** Bóng đá hiện đại phụ thuộc vào dữ liệu sự kiện để tối ưu chiến thuật và tuyển trạch.

**Vấn đề:** Dữ liệu thô thường phức tạp, phi cấu trúc (JSON), khó truy vấn trực tiếp.

**Mục tiêu đồ án:**

Xây dựng quy trình tự động thu thập và xử lý dữ liệu.
Thiết kế Data Warehouse theo mô hình đa chiều.
Tính toán các chỉ số nâng cao (xG, PPDA,...).

**Phạm vi dữ liệu:** Case study CLB Barcelona (La Liga) từ nguồn dữ liệu mở StatsBomb.

**Kiến trúc lựa chọn:** ETL (Extract - Transform - Load) tận dụng sức mạnh tính toán của Apache Spark để xử lý dữ liệu thô.

*xG* (Expected Goals): Xác suất một cú sút thành bàn.

$G - xG$ (Goals minus Expected Goals): Hiệu số giữa tổng số bàn thắng thực tế và tổng xG của cầu thủ hoặc đội bóng.

*xA* (Expected Assists): Xác suất một đường chuyền trở thành kiến tạo, được tính bằng cách lấy xG của cú sút ngay sau đường chuyền đó.

*PPDA* (Passes Per Defensive Action): Số đường chuyền trung bình của đội B trong khu vực 2/3 sân cuối cùng (có tọa độ $x \geq 40$ trên sân có kích cỡ $120 \times 80$) trước khi đội A thực hiện một hành động phòng ngự.

$$\textbf{PPDA}_A = \frac{\text{Số đường chuyền của B trong khu vực } x \geq 40}{\text{Số sự kiện phòng ngự của A trong khu vực } x \geq 40} \quad (1)$$

# Các chỉ số cơ bản trong phân tích bóng đá



*TiB*/90 (Touches in Box/90): Số lần chạm bóng trong vòng cấm của đối phương, được chuẩn hóa theo 90 phút thi đấu.

$$\textbf{TiB/90} = \frac{\text{Tổng số lần chạm bóng trong vòng cấm}}{\text{Tổng số phút đã chơi}} \times 90 \quad (2)$$

*PAdjI*/90 (Possession-Adjusted Interceptions/90): Số lần cắt bóng đã điều chỉnh theo quyền kiểm soát bóng.

$$\textbf{PAdjI/90} = \frac{\text{Tổng số lần cắt bóng}}{\text{Tổng số phút đã chơi}} \times 90 \times \frac{\text{Tỷ lệ \% kiểm soát bóng đội bạn}}{\text{Tỷ lệ \% kiểm soát bóng đội nhà}} \quad (3)$$

*TSR* (Tackles Success Rate): Tỷ lệ tắc bóng thành công.

$$\textbf{TSR} = \frac{\text{Tổng số lần tắc bóng thành công}}{\text{Tổng số lần tắc bóng}} \times 100\% \quad (4)$$

# Nội dung chính

# Nhu cầu của các bên liên quan



**Ban huấn luyện:** Phân tích hiệu suất đội nhà, phân tích đối thủ (chiến thuật, điểm yếu), tối ưu hóa kế hoạch tập luyện.

**Bộ phận tuyển trạch:** Sàng lọc cầu thủ từ tập dữ liệu lớn, so sánh ứng viên tiềm năng.

**Ban lãnh đạo:** Cái nhìn tổng quan, mang tính chiến lược trong điều hành đội bóng, đánh giá hiệu quả đầu tư, ra quyết định dài hạn.

**Cầu thủ:** Tự đánh giá và phát triển, so sánh và đặt mục tiêu, đàm phán hợp đồng.

**Bộ phận truyền thông và Marketing:** Sản xuất nội dung, cá nhân hóa trải nghiệm người hâm mộ.

**Yêu cầu báo cáo:**

Nhóm báo cáo phân tích diễn biến trận đấu & chiến thuật.
Nhóm báo cáo đánh giá hiệu suất cầu thủ.
Nhóm báo cáo phân tích đội nhà & đối thủ.
Nhóm báo cáo tuyển trạch.

# Đặc điểm và quy mô dữ liệu



**Đặc điểm:**

**Định dạng:** JSON bán cấu trúc.

**Cấu trúc:** Phức tạp, lồng nhau nhiều cấp. Một bản ghi sự kiện chứa nhiều object con như `tactics.lineup`, `shot.freeze_frame` (vị trí 22 cầu thủ), `location[x,y]`.

**Quy mô:**

**Số lượng bản ghi:** Khoảng **2.000.000 – 3.000.000** sự kiện. Trung bình một trận đấu chứa khoảng 3.500 sự kiện.

**Dung lượng lưu trữ:** Khoảng **1.5 GB – 2.0 GB** dữ liệu thô (JSON).

**Thách thức kỹ thuật:**

Tuy dung lượng lưu trữ không quá lớn nhưng độ phức tạp của cấu trúc JSON yêu cầu tài nguyên tính toán lớn để thực hiện quá trình làm phẳng. Đây là lý do chính cho việc sử dụng **Apache Spark**.

**Ghi chú:** Dữ liệu sử dụng trong đồ án được lấy từ **StatsBomb Free dataset** cho câu lạc bộ **FC Barcelona** thuộc giải đấu La Liga. Do giới hạn dữ liệu mở, đề tài chọn Barcelona làm *case study* để minh hoạ quy trình xây dựng kho dữ liệu và báo cáo phân tích.

# Nội dung chính

# Tổng quan về cấu trúc dữ liệu



1. Bảng Matches (1 Mùa giải): 36 bản ghi (trận đấu)
2. Bảng Events (1 Trận): 3831 bản ghi (sự kiện)
3. Bảng Lineups (1 Trận): 2 bản ghi (2 đội bóng)
4. Định dạng file: JSON (Nested Structure)

Hình: Tổng quan về cấu trúc các file dữ liệu dạng JSON

Quá trình khảo sát cho thấy dữ liệu từ StatsBomb được tổ chức thành 3 nhóm đối tượng chính: Matches (Thông tin trận đấu), Lineups (Danh sách đăng ký thi đấu) và Events (Chi tiết sự kiện). Dữ liệu này được lưu trữ dưới dạng JSON lồng nhau thay vì dạng bảng phẳng truyền thống, phản ánh độ phức tạp cao của các tình huống trong bóng đá.

# Cấu trúc schema



| Tên trường | Kiểu dữ liệu | Mô tả |
|---|---|---|
| match_id | Long | Khóa chính của trận đấu. |
| match_date | String | Ngày diễn ra trận đấu (YYYY-MM-DD). |
| kick_off | String | Thời gian bắt đầu trận đấu. |
| home_team | Struct | Đội nhà (home_team_id, home_team_name,...). |
| away_team | Struct | Đội khách (away_team_id, away_team_name,...). |
| home_score | Long | Số bàn thắng của đội nhà. |
| away_score | Long | Số bàn thắng của đội khách. |
| competition | Struct | Giải đấu (id, name, country_name). |
| season | Struct | Mùa giải (season_id, season_name). |

Bảng: Tóm tắt cấu trúc dữ liệu bảng Matches

# Cấu trúc schema



| Tên trường | Kiểu dữ liệu | Mô tả |
|---|---|---|
| id | String | Khóa chính của sự kiện. |
| index | Long | Số thứ tự của sự kiện trong trận đấu. |
| timestamp | String | Thời điểm xảy ra sự kiện (phút:giây.miligiây). |
| type | Struct | Loại sự kiện (Pass, Shot,...). |
| possession_team | Struct | Đội đang kiểm soát bóng tại thời điểm đó. |
| play_pattern | Struct | Tình huống bóng (From Corner,...). |
| player | Struct | Thông tin cầu thủ thực hiện hành động (id, name). |
| location | Array<Double> | Tọa độ trên sân dạng mảng $[x, y]$. |
| shot | Struct | Chi tiết cú sút: statsbomb_xg, outcome, body_part,... |
| pass | Struct | Chi tiết đường chuyền: length, angle, height,... |
| tactics | Struct | Thông tin đội hình chiến thuật và vị trí. |

Bảng: Tóm tắt cấu trúc dữ liệu bảng Events

# Cấu trúc schema



| **Tên trường** | **Kiểu dữ liệu** | **Mô tả** |
|---|---|---|
| team_id | Long | ID của đội bóng. |
| team_name | String | Tên đội bóng. |
| lineup | Array<Struct> | Danh sách cầu thủ đăng ký thi đấu. Dữ liệu là một mảng chứa thông tin cầu thủ. |
| – *element* | *Struct* | *Thông tin chi tiết của cầu thủ trong mảng lineup:* |
| .player_id | Long | ID cầu thủ. |
| .player_name | String | Tên đầy đủ cầu thủ. |
| .jersey_number | Long | Số áo thi đấu. |
| .country | Struct | Quốc tịch cầu thủ. |
| .cards | Array | Danh sách thẻ phạt (nếu có). |

Bảng: Tóm tắt cấu trúc dữ liệu bảng Lineups

# Chất lượng dữ liệu



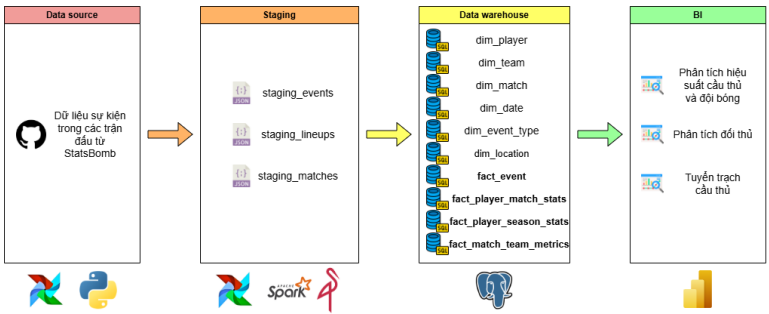
```
1. Số lượng giá trị Null:
+-------------+--------------+--------------+
|Null Location|Null Player ID|Null Timestamp|
+-------------+--------------+--------------+
|           30|            13|             0|
+-------------+--------------+--------------+

2. Số lượng ID sự kiện bị trùng lặp: 0
3. Sự kiện là 'Shot' nhưng thiếu dữ liệu 'shot': 0
```

Hình: Chất lượng dữ liệu sự kiện

Phân tích trên tập dữ liệu đại diện (một trận El Clásico ở mùa giải 2017/2018) cho thấy chất lượng dữ liệu tương đối tốt nhưng vẫn tồn tại Null. Cụ thể, trường location xuất hiện các giá trị Null ở các sự kiện mang tính thủ tục (như tiếng còi bắt đầu hiệp đấu). Không phát hiện trùng lặp khóa chính (ID) trong mẫu thử.

# Kiến trúc Data Warehouse






Hình: Kiến trúc Data Warehouse

**Nguồn dữ liệu (Data source):** Dữ liệu từ GitHub của StatsBomb.

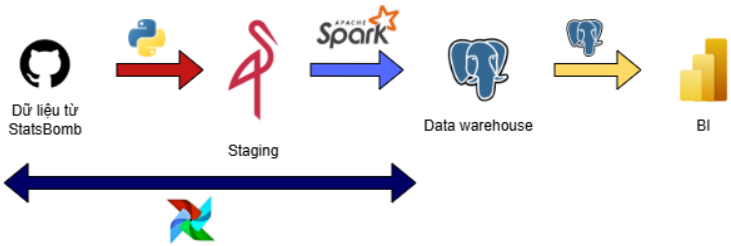
**Vùng đệm (Staging/Data Lake):** Sử dụng **MinIO** để lưu trữ dữ liệu thô.

**Kho dữ liệu (Data Warehouse):** Sử dụng **Apache Spark** để đọc dữ liệu từ MinIO, làm sạch, chuẩn hóa. Tải dữ liệu sạch vào **PostgreSQL**.

**Phân tích và báo cáo (BI):** Sử dụng **Microsoft PowerBI** kết nối trực tiếp với PostgreSQL.

# Đường ống dữ liệu






Hình: Đường ống dữ liệu

Hệ thống sử dụng đường ống dữ liệu tự động hóa được điều phối bởi **Apache Airflow** gồm các giai đoạn:

**Giai đoạn 1: Trích xuất và tập kết (Extract & Ingest)**

**Giai đoạn 2: Chuyển đổi và làm sạch (Transform)**

**Giai đoạn 3: Nạp dữ liệu (Load)**

**Giai đoạn 4: Khai thác và phân phối (Serving)**

# Hệ thống chiều khái niệm



**Nhóm chiều thời gian:** Cung cấp trục thời gian cho phân tích.

| date_value | year | month | day | is_weekend |
|---|---|---|---|---|
| 2017-08-20 | 2017 | 1 | 1 | False |
| 2017-08-26 | 2018 | 10 | 10 | True |
| 2017-09-09 | | 11 | 11 | |
| 2017-09-16 | | 12 | 14 | |
| 2017-09-19 | | 2 | 16 | |
| 2017-09-23 | | 3 | 17 | |
| 2017-10-01 | | 4 | 18 | |
| 2017-10-14 | | 5 | 19 | |
| 2017-10-21 | | 8 | 2 | |
| 2017-10-28 | | 9 | 20 | |
| 2017-11-04 | | | 21 | |
| 2017-11-18 | | | 23 | |
| 2017-11-26 | | | 24 | |
| 2017-12-02 | | | 26 | |
| 2017-12-10 | | | 28 | |
| 2017-12-17 | | | 29 | |
| 2017-12-23 | | | 31 | |
| 2018-01-07 | | | 4 | |
| 2018-01-14 | | | 6 | |
| 2018-01-21 | | | 7 | |
| 2018-01-28 | | | 9 | |

Hình: Nhóm chiều thời gian

# Hệ thống chiều khái niệm



**Nhóm chiều thông tin trận đấu:** Cung cấp thông tin ngữ cảnh cho các trận đấu.

| season | competition | kickoff_tim | stadium | referee | roun | season_stage |
|---|---|---|---|---|---|---|
| 2017/2018 | La Liga | 13:00:00.000 | Estádio Cívitas Metropolitano | Alberto Undiano Mallenco | 1 | Regular Season |
| | | 16:15:00.000 | Abanca-Balaídos | Alejandro José Hernández Hernández | 10 | |
| | | 18:15:00.000 | Coliseum Alfonso Pérez | Antonio Miguel Mateu Lahoz | 11 | |
| | | 20:00:00.000 | Estadi Municipal de Montilivi | Carlos del Cerro Grande | 12 | |
| | | 20:15:00.000 | Estadio Abanca-Riazor | Daniel Jesús Trujillo Suárez | 13 | |
| | | 20:45:00.000 | Estadio Benito Villamarín | David Fernández Borbalan | 14 | |
| | | 21:00:00.000 | Estadio Municipal de Butarque | Ignacio Iglesias Villanueva | 15 | |
| | | 22:00:00.000 | Estadio Municipal de Ipurúa | Jesús Gil Manzano | 16 | |
| | | | Estadio Ramón Sánchez Pizjuán | José Luis González González | 17 | |
| | | | Estadio Santiago Bernabéu | José Luis Munuera Montero | 18 | |
| | | | Estadio de Gran Canaria | José María Sánchez Martínez | 19 | |
| | | | Estadio de Mendizorroza | Juan Martínez Munuera | 2 | |
| | | | Estadio de Mestalla | N/A | 20 | |
| | | | Estadio de la Cerámica | Ricardo De Burgos Bengoetxea | 21 | |
| | | | RCDE Stadium | Santiago Jaime Latre | 22 | |
| | | | Reale Arena | | 23 | |
| | | | San Mamés Barria | | 24 | |
| | | | Spotify Camp Nou | | 25 | |

Hình: Nhóm chiều thông tin trận đấu

**Nhóm chiều thông tin đội bóng:** Cung cấp thông tin cơ bản của các đội bóng.

| team_name | manager_name | home_stadium |
|---|---|---|
| Athletic Club | Abelardo Fernández Antuña | Estádio Cívitas Metropolitano |
| Atlético Madrid | Asier Garitano Aguirrezábal | Abanca-Balaídos |
| Barcelona | Clarence Seedorf | Coliseum Alfonso Pérez |
| Celta Vigo | Cristóbal Parralo Aguilera | Estadi Municipal de Montilivi |
| Deportivo Alavés | Diego Pablo Simeone | Estadio Abanca-Riazor |
| Eibar | Eder Sarabia Armesto | Estadio Benito Villamarín |
| Espanyol | Enrique Setién Solar | Estadio Municipal de Butarque |
| Getafe | Enrique Sánchez Flores | Estadio Municipal de Ipurúa |
| Girona | Ernesto Valverde Tejedor | Estadio Ramón Sánchez Pizjuán |
| Las Palmas | Eusebio Sacristán Mena | Estadio Santiago Bernabéu |
| Leganés | Francisco Jémez Martín | Estadio de Gran Canaria |
| Levante UD | Francisco Martín Ayestarán Barandiarán | Estadio de Mendizorroza |
| Málaga | Imanol Alguacil Barrenetxea | Estadio de Mestalla |
| RC Deportivo La Coruña | Javier Calleja Revilla | Estadio de la Cerámica |
| Real Betis | José Bordalás Jiménez | N/A |
| Real Madrid | José Luis Mendilibar Etxebarria | RCDE Stadium |
| Real Sociedad | José Miguel González Martín del Campo | Reale Arena |
| Sevilla | José Ángel Ziganda Lacunza | San Mamés Barria |
| Valencia | Juan Carlos Unzué Labiano | Spotify Camp Nou |
| Villarreal | Juan Ramón López Muñiz | |

Hình: Nhóm chiều thông tin đội bóng

# Hệ thống chiều khái niệm



**Nhóm chiều thông tin cầu thủ:** Cung cấp thông tin cơ bản của các cầu thủ.

| player_name | nationality | position_group | specific_position |
|---|---|---|---|
| Andrés Iniesta Luján | Argentina | Defender | Center Defensive Midfield |
| Carlos Henrique Casimiro | Belgium | Forward | Center Forward |
| Cristiano Ronaldo dos Santos Aveiro | Brazil | Goalkeeper | Goalkeeper |
| Daniel Ceballos Fernández | Costa Rica | Midfielder | Left Back |
| Denis Suárez Fernández | Croatia | Other | Left Center Back |
| Francisco Alcácer García | France | | Left Center Forward |
| Francisco Casilla Cortés | Germany | | Left Center Midfield |
| Gareth Frank Bale | Netherlands | | Left Defensive Midfield |
| Gerard Piqué Bernabéu | Portugal | | Left Midfield |
| Ivan Rakitić | Spain | | Left Wing |
| Jasper Cillessen | Uruguay | | Right Back |
| Jesús Vallejo Lázaro | Wales | | Right Center Back |
| Jordi Alba Ramos | | | Right Center Forward |
| José Ignacio Fernández Iglesias | | | Right Center Midfield |
| José Paulo Bezzera Maciel Júnior | | | Right Defensive Midfield |
| Karim Benzema | | | Right Midfield |
| Keylor Navas Gamboa | | | Right Wing |
| Lionel Andrés Messi Cuccittini | | | Substitute/Unknown |

Hình: Nhóm chiều thông tin cầu thủ

# Hệ thống chiều khái niệm



**Nhóm chiều thông tin sự kiện:** Các loại hành động, tình huống bóng trong trận đấu (chuyền bóng, sút, tình huống cố định,...).

| event_type | event_category | outcome |
|---|---|---|
| Bad Behaviour | Attack | Blocked |
| Ball Receipt* | Contest | Complete |
| Ball Recovery | Defense | Goal |
| Block | Discipline | Incomplete |
| Carry | Distribution | Off T |
| Clearance | General Play | Out |
| Dispossessed | Goalkeeping | Pass Offside |
| Dribble | | Saved |
| Dribbled Past | | Success |
| Duel | | Unknown |
| Foul Committed | | Wayward |
| Foul Won | | |
| Goal Keeper | | |
| Half End | | |
| Half Start | | |
| Injury Stoppage | | |
| Interception | | |
| Miscontrol | | |
| Offside | | |

Hình: Nhóm chiều thông tin sự kiện

# Hệ thống chiều khái niệm



**Nhóm chiều tình huống bóng:** Các loại hành động, tình huống bóng trong trận đấu (chuyền bóng, sút, tình huống cố định,...).

| play_pattern_name |
|---|
| From Corner |
| From Counter |
| From Free Kick |
| From Goal Kick |
| From Keeper |
| From Kick Off |
| From Throw In |
| Regular Play |

Hình: Nhóm chiều tình huống bóng

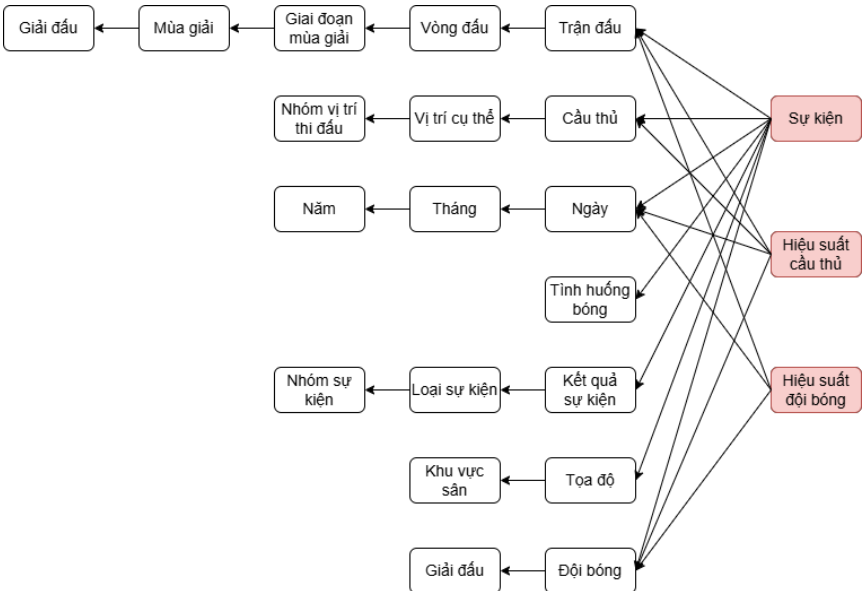
**Nhóm chiều khu vực sân:** Mô tả vị trí trên sân

| field_zone | is_box |
|---|---|
| Attacking Third | False |
| Defensive Third | True |
| Midfield | |

Hình: Nhóm chiều khu vực sân

# Mô hình dữ liệu Logic



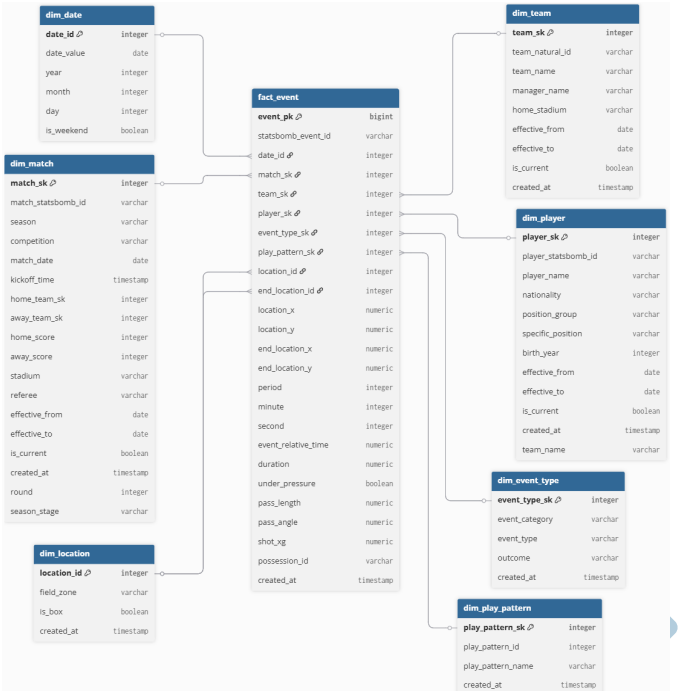



Hình: Mô hình dữ liệu logic

# Mô hình dữ liệu vật lý



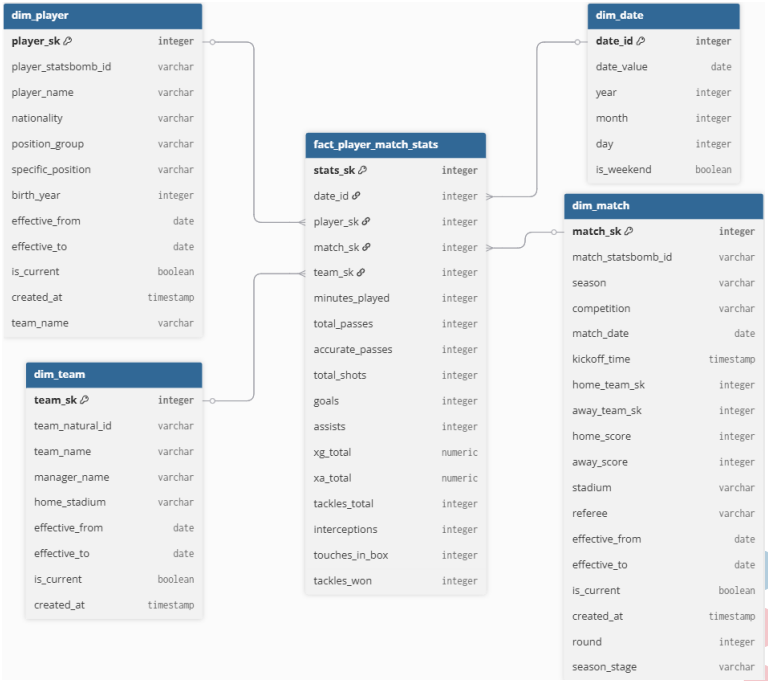
Mô hình dữ liệu vật lý của bảng fact_event:



# Mô hình dữ liệu vật lý



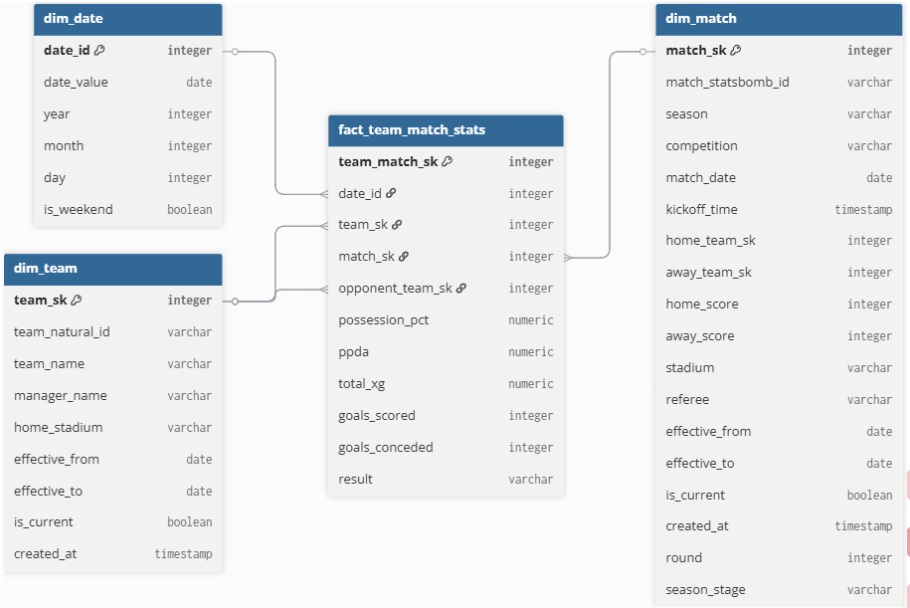
Mô hình dữ liệu vật lý của bảng fact_player_match_stats:



# Mô hình dữ liệu vật lý



Mô hình dữ liệu vật lý của bảng fact_team_match_stats:

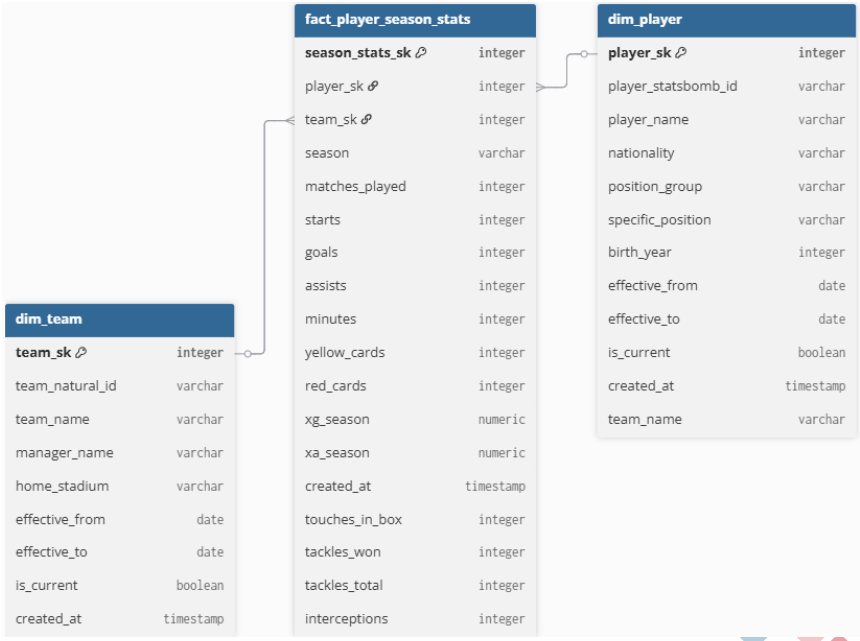



Hình: Mô hình dữ liệu vật lý của bảng fact_team_match_stats

# Mô hình dữ liệu vật lý



Mô hình dữ liệu vật lý của bảng fact_player_season_stats:




Hình: Mô hình dữ liệu vật lý của bảng fact_player_season_stats

# Nội dung chính

# Cấu hình môi trường và kết nối dữ liệu



Sử dụng thư viện `hadoop-aws` để Spark có thể giao tiếp trực tiếp với MinIO thông qua giao thức S3 (`s3a://`). Cấu hình `fs.s3a.path.style.access` được đặt là `true` để đảm bảo tương thích với kiến trúc MinIO chạy trên Docker nội bộ.
Việc ghi dữ liệu vào PostgreSQL được thực hiện thông qua JDBC Driver (`org.postgresql.Driver`). Các cấu hình kết nối được tham số hóa để đảm bảo bảo mật và dễ dàng thay đổi môi trường.

```
spark = SparkSession.builder \
    .appName("ETL_Dim_Date_Fixed") \
    .config("spark.hadoop.fs.s3a.endpoint", MINIO_CONF["endpoint"]) \
    .config("spark.hadoop.fs.s3a.access.key", MINIO_CONF["access_key"]) \
    .config("spark.hadoop.fs.s3a.secret.key", MINIO_CONF["secret_key"]) \
    .config("spark.hadoop.fs.s3a.path.style.access", "true") \
    .config("spark.hadoop.fs.s3a.impl", "org.apache.hadoop.fs.s3a.S3AFileSystem") \
    .config("spark.hadoop.fs.s3a.connection.ssl.enabled", "false") \
    .getOrCreate()
```

Hình: Cấu hình kết nối Apache Spark với MinIO

# Xử lý dữ liệu cho các bảng Dim



**Tính toán thời gian hiệu lực:**

`effective_from`: Là ngày diễn ra của trận đấu đầu tiên (`match_date`) xuất hiện sự thay đổi.

`effective_to`: Sử dụng hàm `lead()` để lấy ngày bắt đầu của bản ghi kế tiếp trừ đi 1 ngày. Nếu không có bản ghi kế tiếp (dữ liệu là bản ghi mới nhất), giá trị được gán mặc định là "9999-12-31".

**Đánh dấu hiện hành:** Cột `is_current` là `true` nếu `effective_to` là "9999-12-31".

```
window_next = Window.partitionBy("team_natural_id").orderBy("match_date")

df_final = df_changes.withColumn("effective_from", col("match_date")) \
    .withColumn("next_start_date", lead("match_date").over(window_next)) \
    .withColumn("effective_to",
        when(col("next_start_date").isNotNull(), date_sub(col("next_start_date"), 1))
        .otherwise(to_date(lit("9999-12-31")))
    ) \
    .withColumn("is_current",
        when(col("effective_to") == to_date(lit("9999-12-31")), True).otherwise(False)
    ) \
```

# Xử lý dữ liệu cho các bảng Dim



Thuật toán SCD Type 2 (Slowly Changing Dimension) có thể giúp giải quyết vấn đề về tính biến động theo thời gian của dữ liệu.

**Phân hoạch dữ liệu:** Dữ liệu nguồn được gom nhóm theo khóa (ví dụ: `player_id` hoặc `team_id`) và sắp xếp tăng dần theo thời gian.

```
window_spec = Window.partitionBy("team_natural_id").orderBy("match_date")
```

**Phát hiện thay đổi:** Sử dụng Window Function `lag()` để so sánh giá trị của bản ghi hiện tại với bản ghi liền trước.

```
df_scd = df_full.withColumn("prev_manager", lag("manager_name").over(window_spec)) \
                .withColumn("prev_name", lag("team_name").over(window_spec)) \
                df_changes = df_scd.filter(
                    (col("prev_manager").isNull()) |
                    (col("manager_name") != col("prev_manager")) |
                    (col("team_name") != col("prev_name"))
                )
```

**Kết quả:** Bảng dim_player và dim_team lưu trữ lịch sử chuyển nhượng và thay đổi nhân sự, cho phép truy vấn chính xác trạng thái của đối tượng tại bất kỳ thời điểm nào trong quá khứ.

# Xử lý dữ liệu cho các bảng Fact



Tự động quét schema của DataFrame để tìm tất cả các cấu trúc chứa trường outcome, giúp hợp nhất cấu trúc với các trường lồng nhau phức tạp trong file JSON gốc thành trường outcome_name duy nhất.

```
outcome_columns = []
for field in df_raw.schema.fields:
    if isinstance(field.dataType, StructType) and 'outcome' in field.dataType.names:
        outcome_columns.append(col(f"{field.name}.outcome.name"))
final_outcome_col = coalesce(*outcome_columns) if outcome_columns else lit(None)
```

Thời gian xảy ra sự kiện được chuyển đổi từ dạng "HH:mm:ss.SSS" sang dạng số thực (giây) để phục vụ các tính toán khoảng cách thời gian giữa các sự kiện.

```
df_ready = df_final_join \
    .withColumn("date_id", date_format(col("match_date"), "yyyyMMdd").cast(IntegerType())) \
    .withColumn("t_parts", split(col("event_relative_time_str"), ":")) \
    .withColumn("calc_seconds",
        col("t_parts")[0].cast("float") * 3600 +
        col("t_parts")[1].cast("float") * 60 +
        col("t_parts")[2].cast("float")
    ) \
    .withColumn("event_relative_time",
        when(col("calc_seconds") > 18000, lit(None)).otherwise(col("calc_seconds"))
    )
```

# Xử lý dữ liệu cho các bảng Fact



Chuẩn hóa tọa độ (x, y) thành các ID từ 1 đến 18 và khu vực đặc biệt (Penalty Box). Logic này sử dụng chuỗi điều kiện when-otherwise lồng nhau, giúp tối ưu tốc độ truy vấn phân tích không gian sau này.

```
def calculate_zone_id(x_col, y_col):
    return when((col(x_col) >= 102) & (col(y_col) >= 18) & (col(y_col) <= 62), 19) \
            .when((col(x_col) < 20), \
                when(col(y_col) < 26.6, 1).when(col(y_col) < 53.3, 2).otherwise(3)) \
            .when((col(x_col) < 40), \
                when(col(y_col) < 26.6, 4).when(col(y_col) < 53.3, 5).otherwise(6)) \
            .when((col(x_col) < 60), \
                when(col(y_col) < 26.6, 7).when(col(y_col) < 53.3, 8).otherwise(9)) \
            .when((col(x_col) < 80), \
                when(col(y_col) < 26.6, 10).when(col(y_col) < 53.3, 11).otherwise(12)) \
            .when((col(x_col) < 100), \
                when(col(y_col) < 26.6, 13).when(col(y_col) < 53.3, 14).otherwise(15)) \
            .otherwise( \
                when(col(y_col) < 26.6, 16).when(col(y_col) < 53.3, 17).otherwise(18))
```

# Xử lý dữ liệu cho các bảng Fact



**Sử dụng Broadcast Join để tối ưu hiệu năng**

Khi thực hiện Lookup dữ liệu từ các bảng Dimension có kích thước nhỏ (như dim_event_type, dim_play_pattern) vào bảng Fact khổng lồ (fact_event), hệ thống sử dụng kỹ thuật Broadcast Join.

**Cơ chế:** Spark sẽ gửi bản sao của bảng Dimension đến tất cả các node worker thay vì thực hiện Sort-Merge Join (yêu cầu shuffle cả bảng Fact lớn).

**Cài đặt:** Sử dụng hàm `broadcast()` bao quanh các DataFrame bảng Dimension trong câu lệnh join.

```
cond_type = (
    (df_j_player.event_type_name == dim_event_type.event_type) &
    (df_j_player.outcome_name.eqNullSafe(dim_event_type.outcome))
)
df_j_type = df_j_player.join(broadcast(dim_event_type), cond_type, "left") \
    .select(df_j_player["*"], dim_event_type["event_type_sk"])
```

**Hiệu quả:** Giảm lưu lượng mạng và loại bỏ hiện tượng phân bổ dữ liệu không đồng đều trên các phân vùng khi join.

# Xử lý dữ liệu cho các bảng Fact



**Chuẩn hóa dữ liệu thống kê tổng hợp**

Thuật toán tính số phút thi đấu thực tế:

**Xác định thời điểm vào sân:** 0 phút cho cầu thủ đá chính, hoặc phút thay người cho cầu thủ dự bị.

```
# A. Tìm thời lượng trận đấu
df_match_duration = df_events.groupBy("match_id") \
    .agg(max("minute").alias("match_end_min"))

# B. Xác định thời điểm VÀO SÂN
df_starters = df_events.filter(col("type.name") == "Starting XI") \
    .select("match_id", explode("tactics.lineup").alias("l")) \
    .select(
        col("match_id"),
        col("l.player.id").cast(StringType()).alias("player_statsbomb_id"),
        lit(0).alias("entry_min")
    )

df_subs_in = df_events.filter(col("type.name") == "Substitution") \
    .select(
        col("match_id"),
        col("substitution.replacement.id").cast(StringType()).alias("player_statsbomb_id"),
        col("minute").alias("entry_min")
    )

df_entries = df_starters.unionByName(df_subs_in)
```

# Xử lý dữ liệu cho các bảng Fact



**Xác định thời điểm rời sân:** Phút thay người (nếu bị thay ra) hoặc phút bị thẻ đỏ.

```
# C. Xác định thời điểm RỜI SÂN
df_subs_out = df_events.filter(col("type.name") == "Substitution") \
    .select(
        col("match_id"),
        col("player.id").cast(StringType()).alias("player_statsbomb_id"),
        col("minute").alias("exit_min")
    )

df_red_cards = df_events.filter(
    col("bad_behaviour.card.name").isin("Red Card", "Second Yellow") |
    col("foul_committed.card.name").isin("Red Card", "Second Yellow")
).select(
    col("match_id"),
    col("player.id").cast(StringType()).alias("player_statsbomb_id"),
    col("minute").alias("exit_min")
)

df_exits = df_subs_out.unionByName(df_red_cards)
```

**Công thức:** Số phút = Thời điểm rời sân/hết trận − Thời điểm vào sân.

# Xử lý dữ liệu cho các bảng Fact



Tính toán các chỉ số nâng cao:

**xG/xA:** Tổng hợp từ dữ liệu sự kiện chi tiết có sẵn trong nguồn dữ liệu gốc.

**Touches in Box:** Đếm số lần chạm bóng có tọa độ nằm trong vòng cấm địa đối phương.

**TSR:** Tính toán dựa trên kết quả của các sự kiện tranh chấp (Duel).

```
# TiB
sum(when(is_in_box, 1).otherwise(0)).alias("touches_in_box"),
# TSR (Total)
sum(when((col("type_name") == "Duel") & (col("duel_type") == "Tackle"), 1).otherwise(0)).alias("tackles_total"),
# TSR (Won)
sum(when(
    (col("type_name") == "Duel") &
    (col("duel_type") == "Tackle") &
    (col("duel_outcome").isin(tackle_won_outcomes)), 1
).otherwise(0)).alias("tackles_won")
```

# Xử lý dữ liệu cho các bảng Fact



**PPDA:** Sử dụng Window Functions để tính toán số đường chuyền của đối thủ trực tiếp trên dòng dữ liệu mà không cần Self-Join gây tốn kém tài nguyên.

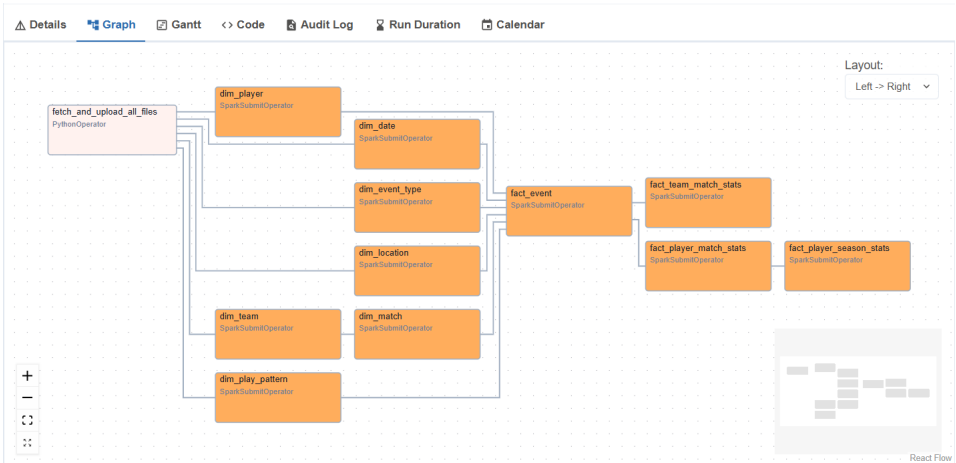
```
w_match = Window.partitionBy("match_id")

df_calc = df_agg_basic \
    .withColumn("match_total_duration", sum("my_duration").over(w_match)) \
    .withColumn("match_total_passes", sum("my_pass_count").over(w_match)) \
    .withColumn("opponent_pass_count", col("match_total_passes") - col("my_pass_count"))
```

```
df_agg_basic = df_metrics.groupBy("match_id", "team_natural_id") \
    .agg(
        # Tổng xG
        sum(coalesce(col("xg"), lit(0))).cast("numeric(6,3)").alias("total_xg"),

        # Thời gian cầm bóng (để tính %) - Clean overflow duration > 999
        sum(when(abs(col("duration")) > 999, 0).otherwise(coalesce(col("duration"), lit(0)))).alias("my_duration"),

        # Số đường chuyền của MÌNH (để tính PPDA cho đối thủ)
        sum(when(col("type_name") == "Pass", 1).otherwise(0)).alias("my_pass_count"),

        # Số hành động phòng ngự của MÌNH (để tính PPDA cho mình)
        sum(when(col("type_name").isin(def_actions), 1).otherwise(0)).alias("my_def_action_count")
    )
```

```
        # Possession % = My Duration / Total Duration * 100
        when(col("match_total_duration") > 0,
             (col("my_duration") / col("match_total_duration") * 100))
        .otherwise(50).cast("numeric(5,2)").alias("possession_pct"),

        # PPDA = Opponent Passes / My Def Actions
        when(col("my_def_action_count") > 0,
             col("opponent_pass_count") / col("my_def_action_count"))
        .otherwise(None).cast("numeric(6,2)").alias("ppda")
```

# Tự động hóa quy trình xử lý với Apache Airflow






Hình: Luồng thực hiện các task trên Apache Airflow

# Dashboard phân tích trận đấu




Hình: Dashboard phân tích trận đấu

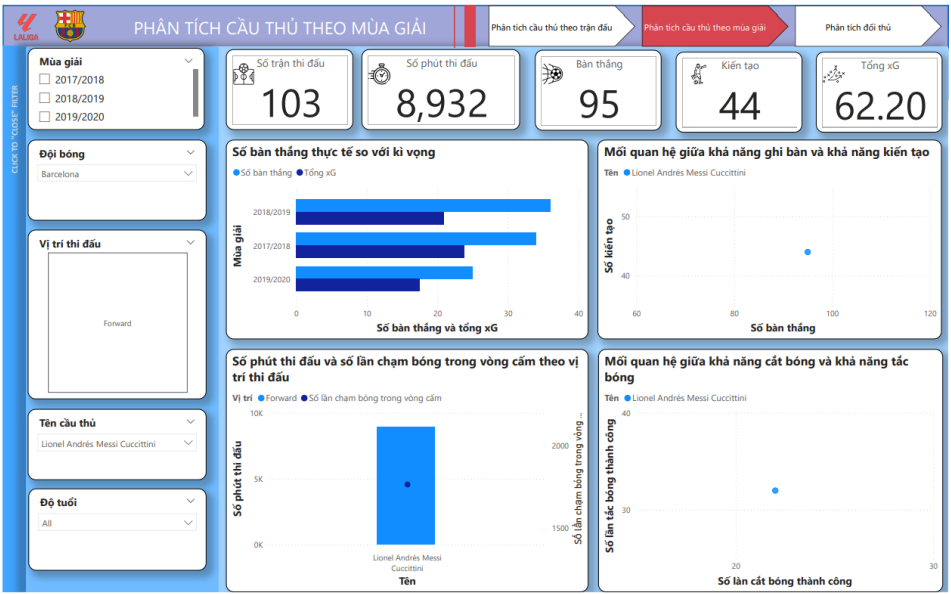
# Dashboard phân tích cầu thủ theo trận đấu




Hình: Dashboard phân tích cầu thủ theo trận đấu

# Dashboard phân tích cầu thủ theo mùa giải






Hình: Dashboard phân tích cầu thủ theo mùa giải

# Dashboard phân tích đối thủ




Hình: Dashboard phân tích đội bóng đối thủ

# Nội dung chính

# Kết luận & Hướng phát triển



## Kết quả đạt được

Xây dựng pipeline dữ liệu tự động (Airflow – MinIO – Spark) cho dữ liệu bóng đá bán cấu trúc.

Thiết kế kho dữ liệu lược đồ sao trên PostgreSQL, tối ưu cho các chỉ số phân tích (xG, xA, PPDA).

Phát triển hệ thống Dashboard Power BI hỗ trợ phân tích và ra quyết định.

## Hạn chế

Hệ thống xử lý theo lô, chưa hỗ trợ phân tích thời gian thực.

Phạm vi dữ liệu còn hạn chế do nguồn StatsBomb mở.

## Hướng phát triển

Tích hợp xử lý thời gian thực (Kafka, Spark Streaming).

Mở rộng hạ tầng đám mây để tăng khả năng mở rộng và hiệu năng.

# Nội dung chính

# Tài liệu tham khảo



[1] TS. Lê Hải Hà. *Bài giảng Phân tích và thiết kế hệ thống*. 2024.

[2] TS. Phạm Huyền Linh. *Bài giảng Phân tích và thiết kế hệ thống*. 2025.

[3] ThS. Nguyễn Danh Tú. *Giáo trình Kho dữ liệu và Kinh doanh thông minh*. 2025.

[4] Wikipedia. *Monopoly*. https://vi.wikipedia.org/wiki/Monopoly. 2025.

# Thank you!