VIDYA JYOTHI
INSTITUTE OF TECHNOLOGY
AN AUTONOMOUS INSTITUTION

(Approved by AICTE, Accredited by NAAC, NBA & permanently Affiliated to JNTUH, Hyderabad)
Aziz Nagar Gate, C.B. Post, Hyderabad-500075

Department of Computer Science and Engineering

DATA WAREHOUSING AND DATA MINING

LAB PROJECT

TITLE:

DESIGN AND IMPLEMENTATION OF AN IPL DATA WAREHOUSE FOR PERFORMANCE ANALYSIS

Submitted to G Surekha

Assistant Professor

BANTU BHASKAR **(22911A05L6)**

ATHELLI TEJA REDDY **(22911A05L4)**

BALLA SHIREESHA **(22911A05L5)**

1. INTRODUCTION

1.1 Background

The Indian Premier League (IPL) is one of the most prominent Twenty20 cricket leagues globally, producing a vast amount of data each season. This data, when processed and stored efficiently, provides valuable insights into player performances, match outcomes, and team strategies. The data from 2008 to 2020 presents an excellent opportunity to apply data warehousing techniques for analytical purposes.

1.2 Objective

The objective of this project is to design and implement a data warehouse that organizes IPL match delivery-level data for the years 2008–2020. The goal is to enable meaningful analysis and visualization through structured storage and transformation of raw match data.

1.3 Scope

- Ingest raw IPL CSV data into a MySQL-based data warehouse.
- Normalize and structure data using dimensional modelling (star schema).
- Load data into appropriate fact and dimension tables.
- Provide support for analysis using SQL queries and Power BI.
- Enable performance metrics evaluation for batsmen, bowlers, teams, and match events.

1.4 Significance

This project demonstrates how data warehousing principles can be applied to a real-world sports dataset. It provides a foundational learning experience in:

- ETL (Extract, Transform, Load) processes
- Dimensional modelling
- SQL querying for analytical purposes
- Integration with BI tools for data visualization.

2. METHODOLOGY

2.1 Data Collection

The dataset used in this project was sourced from publicly available IPL match data in CSV format, covering delivery-level information for all matches played from 2008 to 2020. This data includes details about batsmen, bowlers, runs, dismissals, teams, and more.

2.2 Data Modelling

To enable efficient querying and analysis, a **star schema** was designed for the data warehouse. It includes:

- **Fact Table:**
  - Fact_deliveries: Stores granular delivery-level metrics such as runs, dismissals, and player IDs.
- **Dimension Tables:**
  - dim_players: Stores player information.
  - dim_teams: Contains team names.
  - dim_extras_type: Classifies extra runs (e.g., wide, no-ball).
  - dim_dismissal_kinds: Describes types of dismissals.

This model supports fast aggregation and filtering required for business intelligence use cases.

2.3 ETL Process

An ETL (Extract, Transform, Load) script was developed in **Python** using the mysql.connector module.

- **Extract:** CSV files were read using Python.
- **Transform:** Data was cleaned, foreign keys were mapped from dimension tables.
- **Load:** Transformed data was inserted into MySQL tables with referential integrity.

2.4 Data Loading

The ETL process was run to load over 190,000 rows of delivery data into the MySQL data warehouse. Indexes and constraints were applied to ensure consistency and query performance.

2.5 Data Validation

SQL queries were executed to validate:

- Referential integrity between tables.
- Presence of null/missing values.
- Correctness of player and team mappings.

2.6 Visualization

Power BI was connected to the MySQL data warehouse for creating dashboards and interactive visualizations. Key performance metrics such as top scorers, best bowlers, team statistics, and dismissal analysis were developed.

3. IMPLEMENTATION STEPS

3.1 Environment Setup

- Installed **MySQL Server** for data storage and relational database management.
- Configured **MySQL Workbench** for database administration.
- Set up **Python (v3.x)** with required packages such as pandas, mysql-connector-python.
- Installed **Power BI Desktop** for data visualization and report building.

3.2 Database Schema Design

- Designed a **Star Schema** model with one fact table (fact_deliveries) and multiple dimension tables (dim_players, dim_teams, etc.).
- Created MySQL tables with appropriate data types, primary keys, and foreign key constraints to maintain referential integrity.

3.3 Data Preprocessing

- Loaded raw CSV files into Python.
- Cleaned data by handling:
    - Null and missing values.
    - Inconsistent formatting.
    - Duplicate entries.

3.4 ETL Process

- **Extract:** Read IPL match delivery data from CSV using pandas.
- **Transform:** Mapped raw data fields to dimension table IDs using lookups and merged values.
- **Load:** Inserted data into the MySQL database in a batch-wise manner using mysql-connector.

3.5 Data Analysis & Reporting

- Connected Power BI to MySQL data source.
- Imported relevant tables and built data relationships using the star schema.

4. TOOLS AND TECHNOLOGIES USED

4.1 Programming Language

- **Python 3.x:** Used for data cleaning, preprocessing, transformation, and executing SQL commands for loading data into the database.

4.2 Database Management

- **MySQL Server:** Used as the backend RDBMS to store the fact and dimension tables using a star schema model.
- **MySQL Workbench:** Provided a graphical interface for database creation, schema management, and query execution.

4.3 Business Intelligence Tool

- **Power BI Desktop:** Utilized for importing data from the MySQL database and building interactive dashboards and visual reports for exploratory data analysis.

4.4 Data Handling Libraries (Python)

- **Pandas:** Used for reading CSV data, cleaning, transforming, and preparing the dataset for loading into the MySQL database.
- **mysql-connector-python:** Enabled Python to connect and interact with the MySQL database for the ETL process.

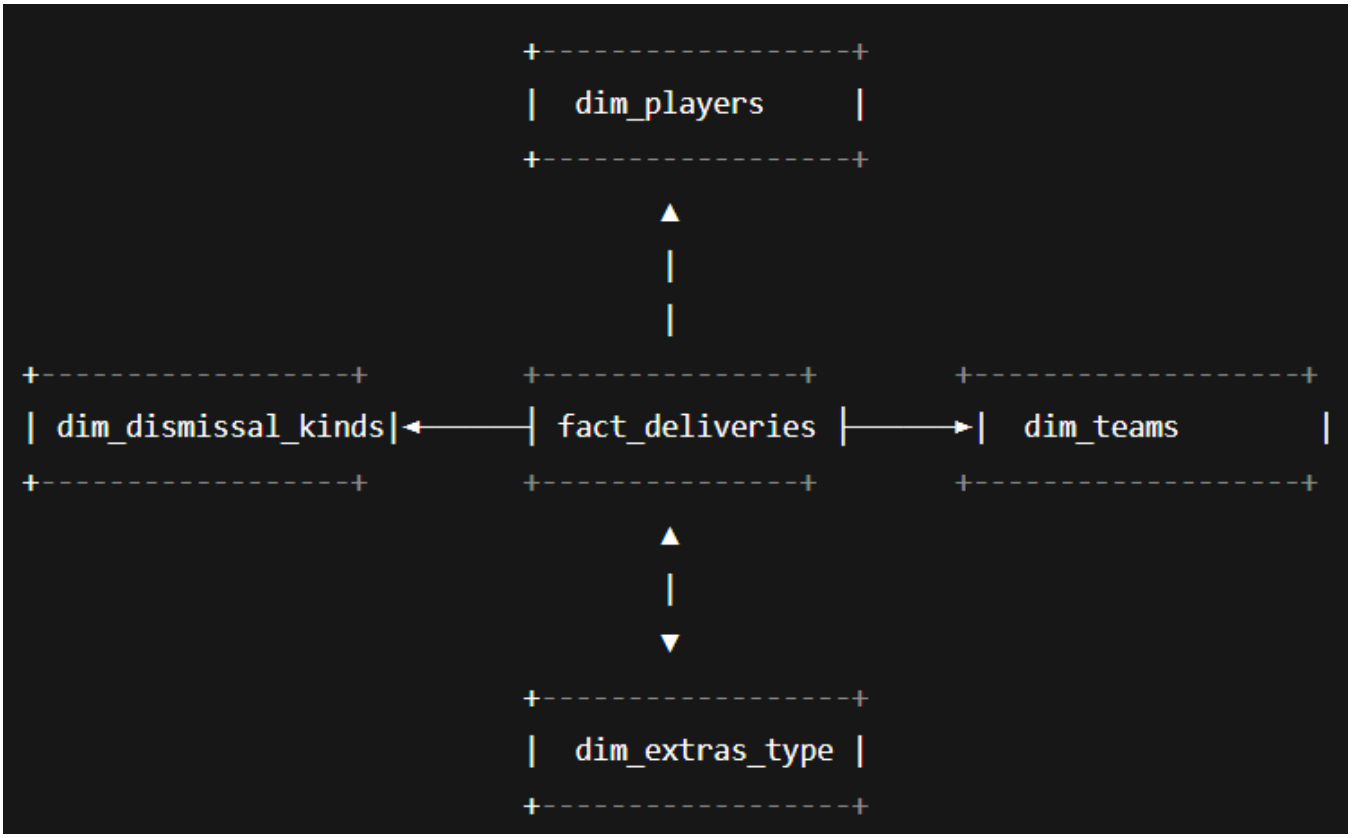
4.5 Development Environment

- **Visual Studio Code / Jupyter Notebook:** Used for writing and executing Python scripts and performing intermediate analysis during the ETL process.

5. STAR SCHEMA DIAGRAM AND TABLE DESCRIPTIONS

5.1 Star Schema Diagram

The data is organized using a star schema, which consists of one central fact table connected to multiple dimension tables. Below is the visual representation:



5.2 Table Descriptions

Table Name	Description
fact_deliveries	Core fact table that stores delivery-level IPL match data including runs, wickets, and player actions.
dim_players	Dimension table storing unique player IDs and names.
dim_teams	Dimension table containing team names and unique IDs.
dim_dismissal_kinds	Dimension table listing types of dismissals (e.g., bowled, caught).
dim_extras_type	Dimension table listing extra run types (e.g., wide, no-ball).

Example: fact_deliveries Table Structure

Column	Description
delivery_id	Unique ID for each delivery
match_id	Match identifier
inning	Inning number
over_number	Over in which delivery was bowled
ball_number	Ball number within the over
batsman_id	Foreign key to dim_players
bowler_id	Foreign key to dim_players
batsman_runs	Runs scored by the batsman
extra_runs	Extra runs awarded
total_runs	Total runs in the delivery
dismissal_kind_id	Foreign key to dim_dismissal_kinds
player_dismissed_id	Player who was dismissed
fielder_id	Fielder involved in dismissal
extras_type_id	Foreign key to dim_extras_type
batting_team_id	Foreign key to dim_teams
bowling_team_id	Foreign key to dim_teams

6. SAMPLE CODE

6.5 SQL Implementation

```
CREATE DATABASE ipl_data_warehouse;
USE ipl_data_warehouse;
CREATE TABLE dim_players (
    player_id INT PRIMARY KEY AUTO_INCREMENT,
    player_name VARCHAR(100) UNIQUE
);
CREATE TABLE fact_deliveries (
    delivery_id INT PRIMARY KEY AUTO_INCREMENT,
    match_id INT,
    inning INT,
    over_number INT,
    ball_number INT,
    batsman_id INT,
    bowler_id INT,
    batsman_runs INT,
    extra_runs INT,

    dismissal_kind_id INT,
    player_dismissed_id INT,
    fielder_id INT,
    extras_type_id INT,
    batting_team_id INT,
    bowling_team_id INT,
    FOREIGN KEY (batsman_id) REFERENCES dim_players(player_id),
    FOREIGN KEY (bowler_id) REFERENCES dim_players(player_id),
    FOREIGN KEY (dismissal_kind_id) REFERENCES dim_dismissal_kinds(id),
    FOREIGN KEY (extras_type_id) REFERENCES dim_extras_type(id),
    FOREIGN KEY (batting_team_id) REFERENCES dim_teams(team_id),
    FOREIGN KEY (bowling_team_id) REFERENCES dim_teams(team_id));
```

6.6 Python Code Sample

```
import pandas as pd
import mysql.connector
df = pd.read_csv('cleaned_data.csv')
conn = mysql.connector.connect(
    host="localhost",
    user="root",
    password="root",  # change this to your actual MySQL password
    database="ipl_data_warehouse")
cursor = conn.cursor()
id_column_map = {
    'dim_players': 'player_id',
    'dim_teams': 'team_id',
    'dim_dismissal_kinds': 'dismissal_kind_id',
    'dim_extras_type': 'extras_type_id'
}
def get_or_create(table, column, value):
    if pd.isna(value): return None
    id_column = id_column_map[table]
    cursor.execute(f"SELECT {id_column} FROM {table} WHERE {column} = %s", (value,))
    result = cursor.fetchone()
    if result: return result[0]
    cursor.execute(f"INSERT INTO {table} ({column}) VALUES (%s)", (value,))
    conn.commit()
    return cursor.lastrowid
for i, row in df.iterrows():
    batsman_id = get_or_create('dim_players', 'player_name', row['batsman'])
    non_striker_id = get_or_create('dim_players', 'player_name', row['non_striker'])
    bowler_id = get_or_create('dim_players', 'player_name', row['bowler'])
```

```
        player_dismissed_id = get_or_create('dim_players', 'player_name', row['player_dismissed']) if pd.notna(row['player_dismissed']) else None
    fielder_id = get_or_create('dim_players', 'player_name', row['fielder']) if pd.notna(row['fielder']) else None
     dismissal_kind_id = get_or_create('dim_dismissal_kinds', 'dismissal_kind', row['dismissal_kind']) if pd.notna(row['dismissal_kind']) else None
          extras_type_id = get_or_create('dim_extras_type', 'extras_type', row['extras_type']) if pd.notna(row['extras_type']) else None
    batting_team_id = get_or_create('dim_teams', 'team_name', row['batting_team'])
    bowling_team_id = get_or_create('dim_teams', 'team_name', row['bowling_team'])
    cursor.execute("""
        INSERT INTO fact_deliveries (
            match_id, inning, over_number, ball_number,
            batsman_id, non_striker_id, bowler_id,
            batsman_runs, extra_runs, total_runs, is_wicket,
            dismissal_kind_id, player_dismissed_id, fielder_id,
            extras_type_id, batting_team_id, bowling_team_id
        ) VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
    """,(
        row['id'], row['inning'], row['over'], row['ball'],
        batsman_id, non_striker_id, bowler_id,
        row['batsman_runs'], row['extra_runs'], row['total_runs'], bool(row['is_wicket']),
        dismissal_kind_id, player_dismissed_id, fielder_id,
        extras_type_id, batting_team_id, bowling_team_id ))
    if i % 5000 == 0:
        print(f"Processed {i} rows...")
conn.commit()
cursor.close()
conn.close()
print(" Data loaded into MySQL data warehouse successfully.")
```

7. RESULTS AND OUTPUT SCREENS

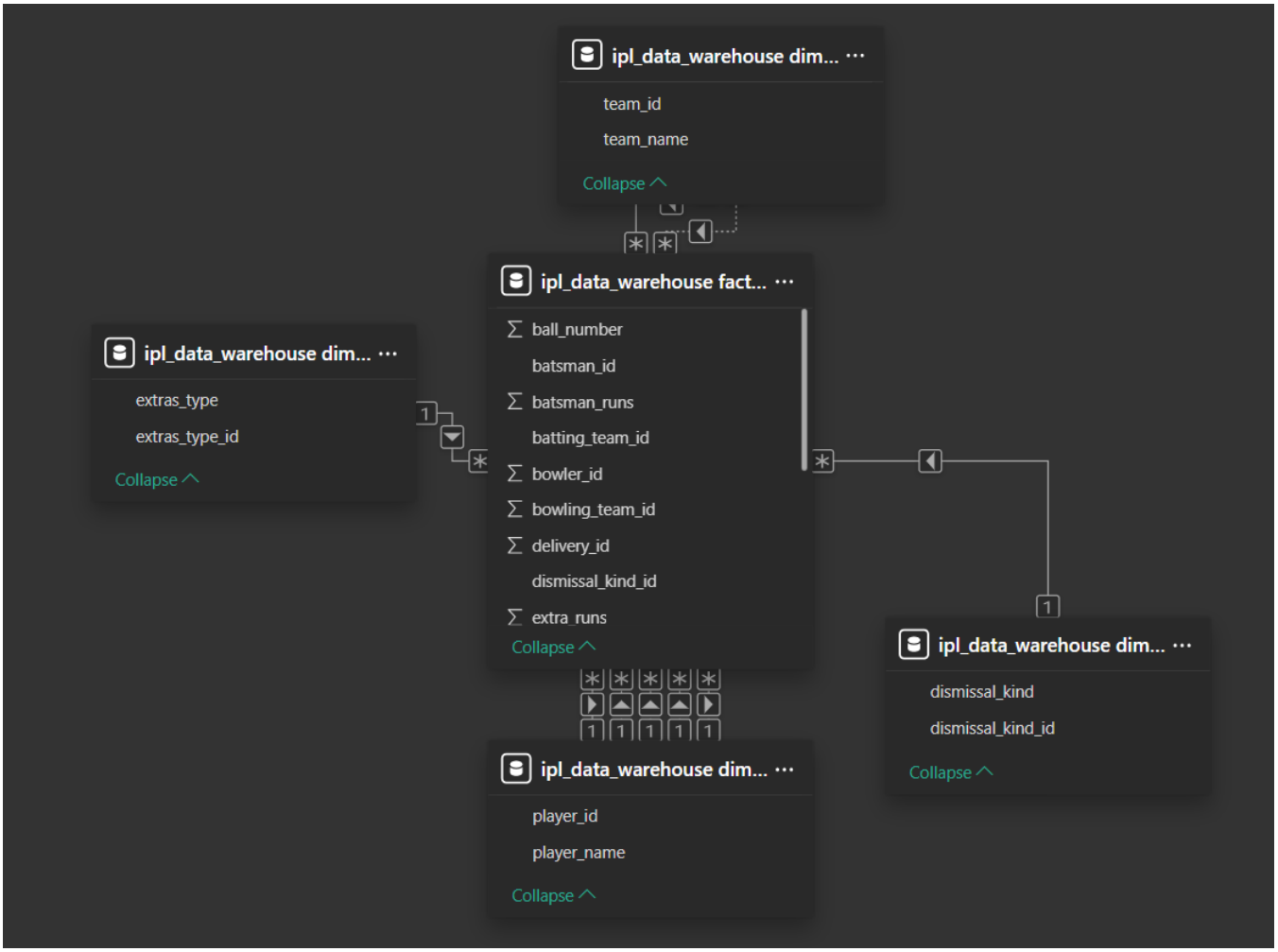
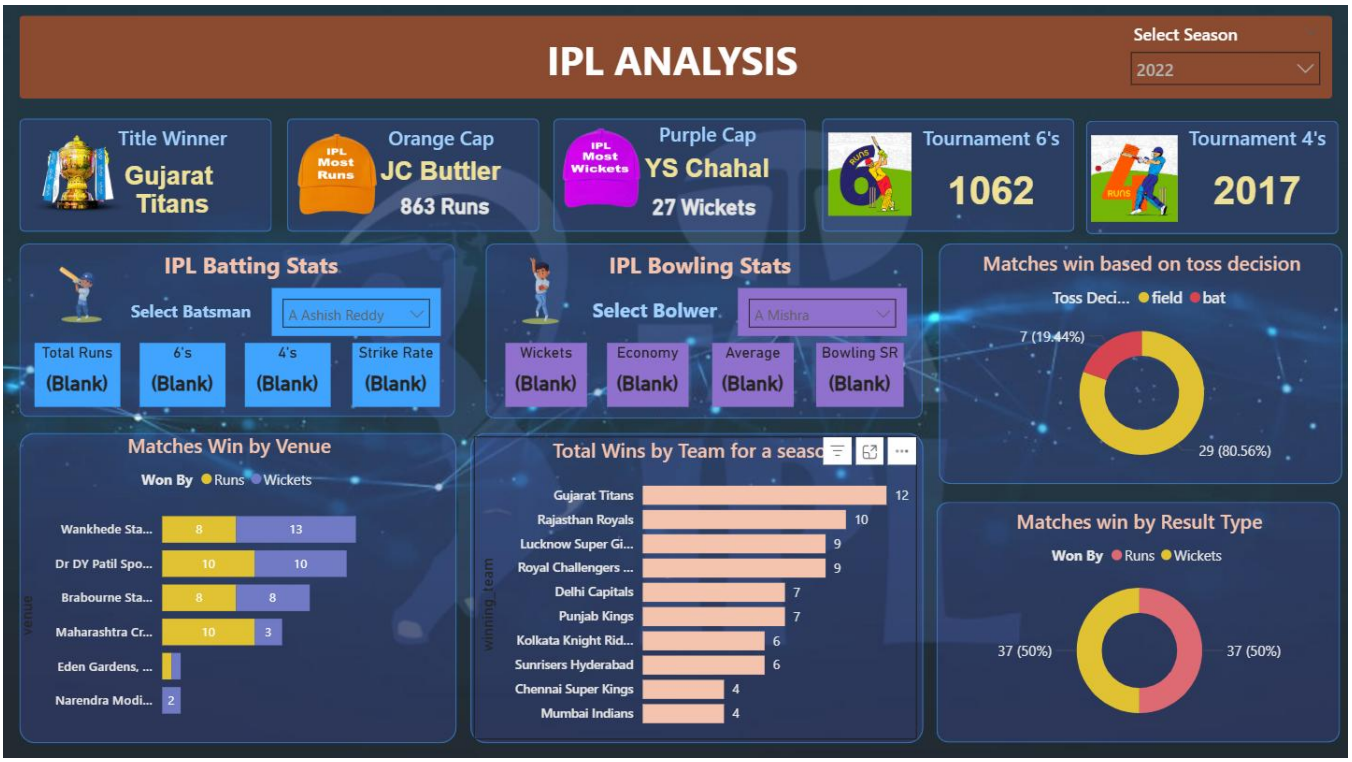


Table structure


IPL_analysis_proj • Last saved: Today at 5:48 PM (Power BI Project)
Search
Sign in
File
Home
Insert
Modeling
View
Optimize
Help
Share
Paste
Cut
Copy
Format painter
Clipboard
Get data
Excel workbook
OneLake catalog
SQL Server
Enter data
Dataverse
Recent sources
Data
Transform data
Refresh
Queries
New visual
Text box
More visuals
Insert
New visual calculation
New measure
Quick measure
Calculations
Sensitivity
Sensitivity
Publish
Share
Copilot
Copilot
IPL ANALYSIS
Select Season
All
Title Winner
Gujarat Titans
Orange Cap
V Kohli
6634 Runs
Purple Cap
DJ Bravo
183 Wickets
Tournament 6's
10.66K
Tournament 4's
25.49K
IPL Batting Stats
Select Batsman
A Ashish Reddy
Total Runs
280
6's
15
4's
16
Strike Rate
142.86
IPL Bowling
Select Bolwer
A Mishra
Wickets
166
Economy
7.19
Average
22.72
Bowling SR
18.95
Matches win based on toss decision
Toss Deci... field bat
159 (32.52%)
330 (67.48%)
Matches Win by Venue
Won By NoResults Runs SuperOver Wickets
Eden Gardens 32 45
Wankhede Sta... 35 37
M Chinnaswam... 26 36
Feroz Shah Kotla 28 31
Rajiv Gandhi In... 20 28
MA Chidambar... 30 17
Sawai Mansing... 15 32
Dubai Internati... 21 22
Punjab Cricket ... 15 20
venue
Total Wins by Team for a season
Delhi Daredevils 67
Delhi Capitals 36
Deccan Chargers 29
Rising Pune Super... 15
Gujarat Lions 13
Punjab Kings 13
Gujarat Titans 12
Pune Warriors 12
Lucknow Super Gi... 9
Kochi Tuskers Kerala 6
NA 4
winning_team
Matches win by Result Type
Won By Wickets Runs SuperOver NoResults
14 (1.47%)
423 (44.53%)
509 (53.58%)
Filters
Visualizations
Data
Page 1


IPL ANALYSIS
Select Season
2022
Title Winner
Gujarat Titans
Orange Cap
JC Buttler
863 Runs
Purple Cap
YS Chahal
27 Wickets
Tournament 6's
1062
Tournament 4's
2017
IPL Batting Stats
Select Batsman
A Ashish Reddy
Total Runs
(Blank)
6's
(Blank)
4's
(Blank)
Strike Rate
(Blank)
IPL Bowling Stats
Select Bolwer
A Mishra
Wickets
(Blank)
Economy
(Blank)
Average
(Blank)
Bowling SR
(Blank)
Matches win based on toss decision
Toss Deci... field bat
7 (19.44%)
29 (80.56%)
Matches Win by Venue
Won By Runs Wickets
Wankhede Sta... 8 13
Dr DY Patil Spo... 10 10
Brabourne Sta... 8 8
Maharashtra Cr... 10 3
Eden Gardens, ... 1
Narendra Modi... 2
venue
Total Wins by Team for a seaso
Gujarat Titans 12
Rajasthan Royals 10
Lucknow Super Gi... 9
Royal Challengers ... 9
Delhi Capitals 7
Punjab Kings 7
Kolkata Knight Rid... 6
Sunrisers Hyderabad 6
Chennai Super Kings 4
Mumbai Indians 4
winning_team
Matches win by Result Type
Won By Runs Wickets
37 (50%)
37 (50%)

8. CONCLUSION

The IPL Data Warehousing project successfully demonstrates how data can be extracted, cleaned, transformed, and loaded into a data warehouse for analytical purposes. By organizing the raw match data from the Indian Premier League (IPL) into fact and dimension tables, we were able to perform detailed analysis and gain insights into player performance, team statistics, and other relevant metrics.

Key Achievements:

- **Data Integration**: The raw IPL dataset was integrated from a CSV format into a structured database schema, following the principles of dimensional modeling (star schema).
- **Data Cleaning**: Missing and erroneous data points were cleaned, ensuring the quality and consistency of the data. Null and incorrect values were addressed through automated scripts.
- **Data Warehousing**: A comprehensive data warehouse was created, consisting of fact tables and multiple dimension tables, allowing efficient querying and reporting.
- **Analytical Queries**: Sample queries, including player statistics, team performance, and match-based analysis, provided valuable insights into IPL match data.
- **Visualization**: Power BI was used for interactive visualizations, offering a dynamic interface to explore and analyse IPL performance trends.