PROJECT TEAM 4

CRICKET LEAGUE DATA MANAGEMENT SYSTEM

Team Members

Sai Meghana Surapaneni (002929424)
Vamsi Krishna Poluru (002924538)
Jayanth Vakkalagadda (002950342)
Roja Pinnamraju (002925814)
Naga Priya Kumbham (002921442)

Problem Statement

- The purpose of this database is to maintain a major cricket league (Indian Premier League) data for managing the players, teams, staff, and other statistics like schedule, match results, player leaderboard, and points table
- This database captures minute details like ball-to-ball score, bowling speed which could enable sports analysts and team coaches to provide strategic tips to the players
- The league partners can also leverage this data for updating the scoreboard and other statistics on the league's website and mobile application

TOP 10 TRENDING POPULAR SPORTS IN THE WORLD 2022

Here is the list of Top 10 Trending Popular Sports in the World 2022.

Rank	Sport	Worldwide Estimated Fans	Mainly Popular in
1	Soccer (Football)	3.5 billion	Europe, Africa, Asia, America
2	Cricket	2.5 billion	Asia, Australia, UK
3	Basketball	2.2 billion	US, Canada, China, Philippines

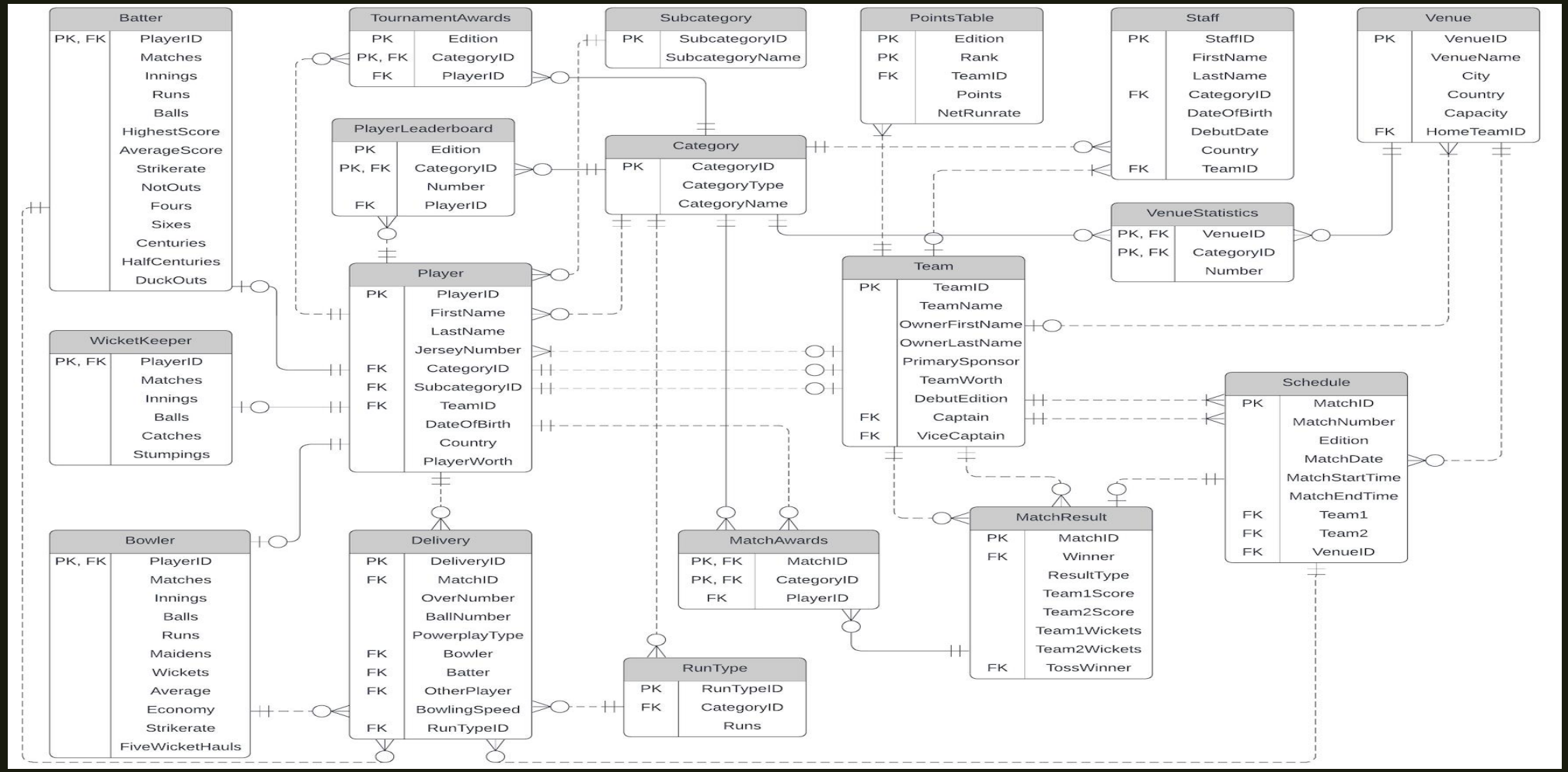
Key Business Rules

- Each match will involve two teams
- Each match will have only 22 players and 11 from each team
- Each match has 2 innings
- Each innings should have up to 20 overs
- Each innings may have up to 10 wickets
- Each team in a match gets a result of win/ lose/ draw
- Each venue may have a home team
- Each over may have only 6 legal deliveries
- Each delivery has a bowler and a batter
- Each run type may have 0-6 runs usually
- Every team may have multiple staff
- Each staff member can be mapped to only one team

Database Tables

	Table Name
1	Category
2	Subcategory
3	Player
4	Team
5	Batter
6	Bowler
7	WicketKeeper
8	Venue
9	Schedule
10	RunType
11	Delivery
12	PlayerLeaderboard
13	TournamentAwards
14	MatchResult
15	MatchAwards
16	PointsTable
17	Staff
18	VenueStatistics

ER Diagram


Batter
PK, FK PlayerID
Matches
Innings
Runs
Balls
HighestScore
AverageScore
Strikerate
NotOuts
Fours
Sixes
Centuries
HalfCenturies
DuckOuts
TournamentAwards
PK Edition
PK, FK CategoryID
FK PlayerID
Subcategory
PK SubcategoryID
SubcategoryName
PointsTable
PK Edition
PK Rank
FK TeamID
Points
NetRunrate
Staff
PK StaffID
FirstName
LastName
FK CategoryID
DateOfBirth
DebutDate
Country
FK TeamID
Venue
PK VenueID
VenueName
City
Country
Capacity
FK HomeTeamID
PlayerLeaderboard
PK Edition
PK, FK CategoryID
Number
FK PlayerID
Category
PK CategoryID
CategoryType
CategoryName
VenueStatistics
PK, FK VenueID
PK, FK CategoryID
Number
Player
PK PlayerID
FirstName
LastName
JerseyNumber
FK CategoryID
FK SubcategoryID
FK TeamID
DateOfBirth
Country
PlayerWorth
Team
PK TeamID
TeamName
OwnerFirstName
OwnerLastName
PrimarySponsor
TeamWorth
DebutEdition
FK Captain
FK ViceCaptain
WicketKeeper
PK, FK PlayerID
Matches
Innings
Balls
Catches
Stumpings
Schedule
PK MatchID
MatchNumber
Edition
MatchDate
MatchStartTime
MatchEndTime
FK Team1
FK Team2
FK VenueID
Bowler
PK, FK PlayerID
Matches
Innings
Balls
Runs
Maidens
Wickets
Average
Economy
Strikerate
FiveWicketHauls
Delivery
PK DeliveryID
FK MatchID
OverNumber
BallNumber
PowerplayType
FK Bowler
FK Batter
FK OtherPlayer
BowlingSpeed
FK RunTypeID
MatchAwards
PK, FK MatchID
PK, FK CategoryID
FK PlayerID
MatchResult
PK MatchID
FK Winner
ResultType
Team1Score
Team2Score
Team1Wickets
Team2Wickets
FK TossWinner
RunType
PK RunTypeID
FK CategoryID
Runs

Computed Columns

```
--Batter Table
CREATE TABLE [Batter] (
  [PlayerID] INT NOT NULL FOREIGN KEY REFERENCES Player(PlayerID),
  [Matches] INT,
  [Innings] INT,
  [Runs] INT,
  [Balls] INT,
  [HighestScore] INT,
  [AverageScore] AS [Runs]/NULLIF([Innings],0),
  [Strikerate] AS 100*[Runs]/NULLIF([Balls],0),
  [NotOuts] INT,
  [Fours] INT,
  [Sixes] INT,
  [Centuries] INT,
  [HalfCenturies] INT,
  [DuckOuts] INT,
  CONSTRAINT PKBatter PRIMARY KEY (PlayerID)
);
```

	PlayerID	Matches	Innings	Runs	Balls	HighestScore	AverageScore	Strikerate	NotOuts	Fours	Sixes	Centuries	HalfCenturies	DuckOuts
1	1	12	10	420	302	102	42	139	10	20	11	1	2	1
2	2	13	13	631	420	133	48	150	12	33	17	2	5	0
3	3	14	9	331	292	56	36	113	4	21	7	0	2	2
4	4	13	11	228	210	47	20	108	3	26	4	0	0	1
5	5	11	7	194	147	61	27	131	3	19	8	0	1	0
6	6	15	13	368	322	74	28	114	5	34	12	0	2	3
7	7	12	10	424	332	92	42	127	0	32	3	0	4	4
8	8	7	7	286	271	26	40	105	3	9	4	0	0	1
9	9	13	13	541	412	121	41	131	8	41	20	1	3	2
10	10	15	13	368	322	74	28	114	5	34	12	0	2	3
11	11	12	10	424	332	92	42	127	0	32	3	0	4	4
12	12	7	7	286	271	26	40	105	3	9	4	0	0	1
13	13	13	13	541	412	121	41	131	8	41	20	1	3	2
14	14	15	13	379	322	74	29	117	5	34	12	1	2	3
15	15	12	10	404	332	89	40	121	0	32	3	0	3	4

Batter Table -
Average Score, Strikerate

Computed Columns Using Functions

	VenueID	CategoryID	Number
1	1	1	120.00
2	1	2	110.00
3	2	1	130.00
4	2	2	100.00
5	3	1	110.00
6	3	2	100.00
7	4	1	150.00
8	4	2	140.00
9	5	1	170.00
10	5	2	120.00

VenueStatistics Table - Number

```
--Computed Columns Using Functions
USE Team4;
GO
--VenueStatistics
--Get the Highest Team Score category value for a venue
--Get Least Team Score category value for a venue

CREATE FUNCTION CalcNumber(@VId INT, @CId INT)
RETURNS DECIMAL(10,2)
AS
BEGIN
    DECLARE @Team1Max DECIMAL(10,2)=0;
    DECLARE @Team2Max DECIMAL(10,2)=0;
    DECLARE @Team1Min DECIMAL(10,2)=0;
    DECLARE @Team2Min DECIMAL(10,2)=0;
    DECLARE @FinalValue DECIMAL(10,2)=0;

    SELECT @Team1Max =MAX(Team1Score),
           @Team2Max =MAX(Team2Score),
           @Team1Min =MIN(Team1Score),
           @Team2Min =MIN(Team2Score)
        FROM Schedule A
        INNER JOIN MatchResult B ON A.MatchID = B.MatchID
        WHERE VenueID = @VId;

    IF @CId = 1
        IF @Team1Max > @Team2Max
            SET @FinalValue = @Team1Max
        ELSE
            SET @FinalValue = @Team2Max
    ELSE
            IF @Team1Min < @Team2Min
            SET @FinalValue = @Team1Min
        ELSE
            SET @FinalValue = @Team2Min

    RETURN @FinalValue;

END
GO

ALTER TABLE VenueStatistics ADD [Number] AS (dbo.CalcNumber(VenueID, CategoryID));
```

Computed Columns Using Functions

	Edition	Rank	TeamID	Points	NetRunrate
1	2021	1	2	18	0.00
2	2021	2	5	14	0.00
3	2021	3	4	14	0.00
4	2021	4	10	12	0.00
5	2021	5	7	11	0.00
6	2021	6	8	9	0.00
7	2021	7	6	8	0.00
8	2021	8	3	6	0.00
9	2021	9	6	4	0.00
10	2021	10	1	4	0.00
11	2022	1	5	18	0.00
12	2022	2	2	14	0.50
13	2022	3	1	14	0.50
14	2022	4	10	12	0.00
15	2022	5	6	11	1.00

PointsTable - NetRunRate

```sql
--PointsTable
--Calculate NetRunRate by calculating the win percentage for each team

CREATE FUNCTION CalcNRR(@Edi INT, @TId INT)
RETURNS DECIMAL(10,2)
AS
BEGIN
    DECLARE @MatchesPlayed DECIMAL(10,2) = 0;
    DECLARE @MatchesWon DECIMAL(10,2) = 0;
    DECLARE @NRR DECIMAL(10,2) = 0;

    SELECT @MatchesPlayed = COUNT(*)
    FROM Schedule
    WHERE Edition = @Edi AND (Team1 = @TId OR Team2 = @TId);

    SELECT @MatchesWon = COUNT(*)
    FROM Schedule A
    INNER JOIN MatchResult B ON A.MatchID = B.MatchID
    WHERE A.Edition = @Edi AND (B.Winner = @TId);

    IF @MatchesPlayed=0
        SET @NRR = 0
    ELSE
        SET @NRR = @MatchesWon/@MatchesPlayed;

    RETURN @NRR;
END
GO

ALTER TABLE PointsTable ADD [NetRunrate] AS (dbo.CalcNRR(Edition, TeamID));
```

Table-Level Check Constraints

PointsTable ⇨

```
--Table Level Constraints Using Functions
USE Team4;

--PointsTable
--Allow Teams in Points Table from their debut edition

CREATE FUNCTION CheckEdition(@TId INT, @Edition INT)
RETURNS INT
AS
BEGIN
    DECLARE @DebutFlag int=0;
        SELECT @DebutFlag = @Edition-DebutEdition
        FROM Team
        WHERE TeamID = @TId;
    RETURN @DebutFlag;
END;

ALTER TABLE PointsTable ADD CONSTRAINT RejectWrongEdition CHECK (dbo.CheckEdition(TeamID, Edition)>=0);
```

```
--Bowler
--Allow Players who are not WicketKeepers

CREATE FUNCTION CheckWicketKeeper(@PId INT)
RETURNS INT
AS
BEGIN
    DECLARE @WicketKeeperFlag int=0;
        SELECT @WicketKeeperFlag = COUNT(*)
        FROM Wicketkeeper
        WHERE PlayerID = @PId;
    RETURN @WicketKeeperFlag;
END;

ALTER TABLE Bowler ADD CONSTRAINT RejectWicketKeeper CHECK (dbo.CheckWicketKeeper(PlayerID)=0);
```

⇦ **Bowler Table**

Table-Level Check Constraints

```
--Wicketkeeper
--Allow Players who are not Bowlers

CREATE FUNCTION CheckBowler(@PId INT)
RETURNS INT
AS
BEGIN
    DECLARE @BowlerFlag int=0;
        SELECT @BowlerFlag = COUNT(*)
        FROM Bowler
        WHERE PlayerID = @PId;
    RETURN @BowlerFlag;
END;

ALTER TABLE WicketKeeper ADD CONSTRAINT RejectBowler CHECK (dbo.CheckBowler(PlayerID)=0);
```

⇧

Wicketkeeper Table

TournamentAwards Table

⇩

```
--TournamentAwards
--Allow 'Purple Cap' Award for only Bowlers, Allow 'Orange Cap' for only Batters

CREATE FUNCTION CheckPlayerTypeAwards(@PId INT, @AwardCategory INT)
RETURNS INT
AS
BEGIN
    DECLARE @AwardFlag int=0;

    IF @AwardCategory = 10
        BEGIN
        SELECT @AwardFlag = COUNT(*)
            FROM TournamentAwards A
            INNER JOIN Player B ON A.PlayerID=B.PlayerID
            INNER JOIN Category C ON B.CategoryID=C.CategoryID
            WHERE C.CategoryName = 'Batter' AND A.PlayerID=@PId;
        END

    ELSE
        BEGIN
        IF @AwardCategory = 11
            BEGIN
            SELECT @AwardFlag = COUNT(*)
                FROM TournamentAwards A
                INNER JOIN Player B ON A.PlayerID=B.PlayerID
                INNER JOIN Category C ON B.CategoryID=C.CategoryID
                WHERE C.CategoryName = 'Bowler' AND A.PlayerID=@PId;
            END

        ELSE
            SET @AwardFlag = 1
        END

    RETURN @AwardFlag;
END;

ALTER TABLE TournamentAwards ADD CONSTRAINT RejectPlayerAward
CHECK (dbo.CheckPlayerTypeAwards(PlayerID, CategoryID)>0);
```

Views

Report2

```
--Captains and their Player of the Match Awards Count
CREATE VIEW Report2 AS
SELECT B.FirstName, B.LastName, COUNT(*) AS PlayerOfTheMatchAwards
FROM Team A
INNER JOIN Player B ON A.Captain = B.PlayerID
INNER JOIN MatchAwards C ON A.Captain = C.PlayerID
INNER JOIN Category D ON C.CategoryID = D.CategoryID
WHERE D.CategoryName = 'Player of the Match'
GROUP BY B.FirstName, B.LastName;
GO


SELECT * FROM Report2;
```

	FirstName	LastName	PlayerOfTheMatchAwards
1	Shikar	Dhawan	1
2	Virat	Kohli	1

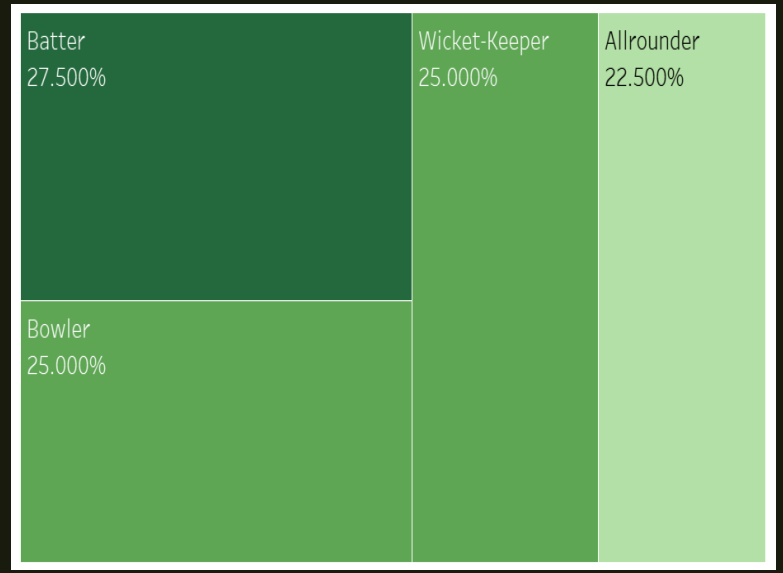
```
--Creating Views
USE Team4;
GO

--Teams and their Wins at Home Venue
CREATE VIEW Report1 AS
SELECT D.TeamName, COUNT(*) AS HomeVenueWins
FROM Schedule A
INNER JOIN MatchResult B ON A.MatchID=B.MatchID
INNER JOIN Venue C ON A.VenueID=C.VenueID
INNER JOIN Team D ON B.Winner=D.TeamID
WHERE B.Winner = C.HomeTeamID AND B.Winner IS NOT NULL
GROUP BY D.TeamName;
GO

SELECT * FROM Report1;
```

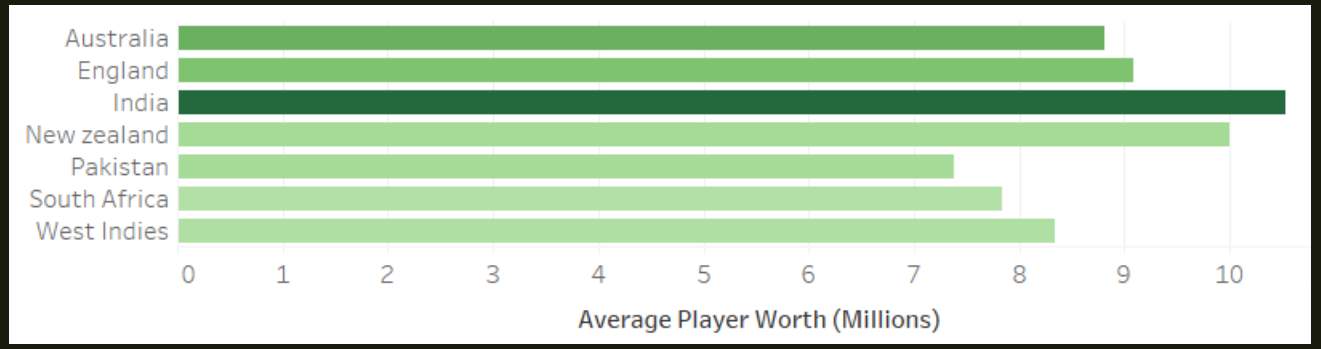
	TeamName	HomeVenueWins
1	Chennai Super Kings	1
2	Delhi Capitals	1
3	Kolkata Knight Riders	1
4	Mumbai Indians	1
5	Rising Pune Supergiants	1
6	Royal Challengers Bangalore	1

Report1

Visualizations

Percentage of different categories of players in the league

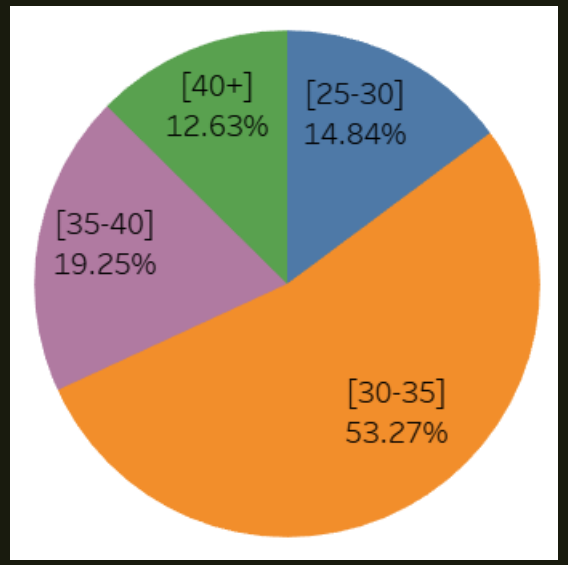


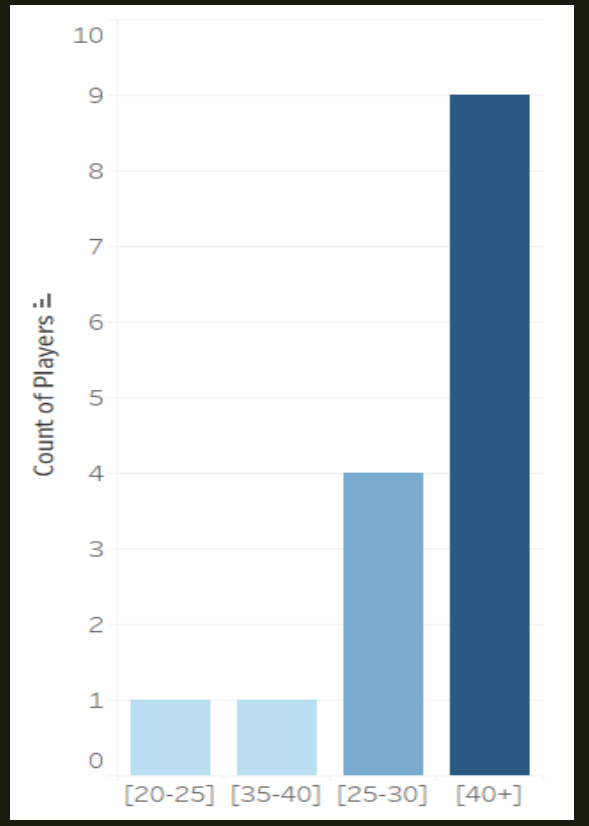


Average Player Worth for various Countries

Visualizations

Percentage of players in various Age groups





Batter Average groups

Addressed Problems

- Allow maintenance and tracking of data on Players, Teams, Staff, Venue
- Provide data on ball-to-ball scores and other details like bowling speed, runs, extra-runs, and wickets
- Maintain and track several statistics of players like centuries, half-centuries, five wicket hauls, maiden overs
- Help provide history of past and current edition match results, toss results, team scores, points table
- Understand statistics like team performances at home venue
- Venue Insights like level highest and lowest totals
- Provide details on various player level performances in batting and bowling powerplays
- Maintain data on various awards distributed to players on match level and edition level


Thank You