Problem Statement – Player Auction System
Design and implement an application program to automate the process of auctioning for cricket trophy. The application stores a collection of players and provides mechanism of displaying players details based on team name.
The main menu contains following options:

1. **Add a player**
2. **Display players**
3. **Exit**

It is redisplayed after each operation is completed, except for the **'Exit'** option. The details for each option are specified in the rest of the document.

Your application program should follow the control flow as, start from class **PlayerAuctionSystemClient -> PlayerAuctionSystemManager->DAO class -> Database**

Instructions:

I. Create following three tables **'Team', 'Player'** and **'Team_Player'** which will be used in the application. For Table **'Team'**, manually insert the values as per values given in this document. The values for **'Player'** and **'Team_Player'** must be inserted through your application.
II. All user defined exceptions must be created under com.mindtree.exceptions package
III. All exceptions must be handled in Manager Class. DAO class should not handle any exception; it must throw it to Manager Class.
IV. Loose coupling: Create a separate Dao interfaces and implementation classes. Database interaction code should be only in DAO classes. Dao classes should not have UI code.

1. **Database design:**

Create database "Player_DB" and create the following three tables used for the application: "Team", "Player" and a link table "Team_Player". Listed below is the table design with constraints.

Table Name: **Team**

Column Name	Data type	Constraint	Description
Team_Id	INT	Primary Key	
Team_Name	Varchar(6)	NOT NULL	

Below is the content of table **'Team',** which must be manually inserted into table**:**

Team_Id	Team_Name
1	CSK
2	RCB
3	MI
4	RR
5	SRH
6	KKR

Table Name: **Player**

Column Name	Data Type	Constraint	Description
Player_No	INT	Primary Key	It must be auto increment
Player_Name	Varchar(20)	NOT NULL	
Category	Varchar(20)	Can be only 'batsman' or 'bowler' or 'Allrounder'	
HighestScore	INT		
BestFigure	Varchar(10)	Can be like %/%	Example:3/25

Table Name: **Team_Player**

Column Name	Data type	Constraint	Description
Player_No	INT	Foreign Key	References Player_No column of Player
Team_Id	INT	Foreign Key	References Team_Id column of Team

2. **Add a Player**
   - When user selects option 1, the user will enter the **playerName, category, highestScore, bestFigure** and **teamName.**
   - Sample output shown below.

```
Add Player!!
--------------------
Enter Player Name: M S Dhoni
Enter Category: Batsman
Enter highest score: 75
Enter best figure: 0/20
Enter team name: CSK
```

Write code to display the output in below format by using the returned value if it is successfully saved in the table 'Player', otherwise display an appropriate exception/error message.

```
Player added successfully with player No:1
```

Your **PlayerAuctionSystemManager class** code should check for the business rules given below and throw appropriate user defined exception, when exception thrown corresponding message should be displayed to user, which is as given in the table:

Rule No	Business constraint	User defined exceptions to be thrown	Message to user to be displayed
1	**category** entered should be either 'Batsman','Bowler' or 'Allrounder'	If **category** is invalid, throw **InvalidCategoryException**	"Invalid category name please check your input"
2	**teamName** entered should be present in the database.	If **teamName** is invalid, throw **InvalidTeamNameException**	"Invalid team name, please check your input"
3	For a batsman**,** **highestScore** should be between 50 and 200 inclusive	If **category** is 'Batsman' and **highestScore** is invalid, throw **NotABatsmanException**	"Invalid Batsman, please check your input"
4	For a bowler, **bestFigure** should not be null and **highestScore** should not be negative	If category is 'Bowler' and **bestFigure** are invalid, throw **NotABowlerException**	"Invalid Bowler, please check your input"
5	Same player name for given category & team should not be repeated	If player name for given category & team is already exist in database, then it should throw **DuplicateEntryException**. For example: If input player name is M S Dhoni, category is 'Batsman' and team name is 'CSK', if another player with same name, category, and team is already exist in database, then it must throw DuplicateEntryException	"Player details already exist in the database"

3. **Display Players:**

- When user selects Option 2, user will input **teamName**.
- Sample output shown below.

```
Display Players!!
-------------------------
Enter team name: CSK
```

Write code to display the output in below sorted order (**sort by player's name**) by using the returned values if the business rules are satisfied; otherwise display the appropriate exception message.

```
Player Name              Category
-------------------------------------
M S Dhoni                  Batsman
Murali Vijay               Batsman
Ravindra Jadeja            Allrounder
Suresh Raina               Allrounder
```

Your **PlayerAuctionSystemManager class** code should check for the business rules given below and throw appropriate user defined exception, when exception thrown corresponding message is displayed to user, which is as given in the table:

Rule No	Business constraint	User defined exceptions to be thrown	Message to user to be displayed
1	**teamName** entered should be present in the database.	If **teamName** is invalid, throw **InvalidTeamNameException**	"Invalid team name, please check your input"

4. Exit

When user selects option 3, application should terminate