



A Project Report On “CRICKET DATABASE MANAGEMENT”

Submitted By:

M.Hanumantha rao

CBSE Roll No :

School No: 53

Class: XII B

Under the Guidance of

Mr. Anoop V S

PGT (Computer Science)

Department of Computer Science

SAINIK SCHOOL KALIKIRI

Department of Computer Science

SAINIK SCHOOL KALIKIRI



This is to certify that **Cdt. M.Hanumantha rao**, Roll No. 53 of Class XII has prepared the report on the Project entitled **“CRICKET DATABASE MANAGEMENT”**. The report is the result of his efforts & endeavors. The report is found worthy of acceptance as final project report for the subject Computer Science of Class XII.

Signature
(Internal Examiner)

Signature
(External Examiner)



DECLARATION

I hereby declare that the project work entitled “**CRICKET DATABASE MANAGEMENT**”, submitted to Department of **Computer Science**, **SAINIK SCHOOL KALIKIRI** is **prepared by me**. All the **coding** is the result of my **personal efforts**.

Cdt. M.Hanumantha rao
Roll No: 53
Class: XII B
SAINIK SCHOOL KALIKIRI



ACKNOWLEDGEMENT

I would like to express a deep sense of thanks & gratitude to my **project guide Mr. Anoop V S** Sir for guiding me immensely through the course of the project. He always evinced keen interest in my work. His constructive advice & **constant motivation** have been responsible for the **successful** completion of this project.

My sincere thanks go to **Lt Col Susheel Kumar Mahapatro SM** , our **Offg Principal** sir, for his co-ordination in extending every **possible support** for the completion of this project.

I also thanks to my **parents** for their **motivation & support**. I must thanks to my **classmates** for their timely help & support for **compilation** of this **project**.

Last but not the least, I would like to thank all those who had helped directly or indirectly towards the completion of this project.

Cdt. M.Hanumantha rao
Roll No:53
Class: XII B
SAINIK SCHOOL KALIKIRI

CONTENTS

1. Working Description
2. Code of the Project
3. Output Screens
4. Bibliography

1. WORKING DESCRIPTION

1.Format table

- This shows a table that contains international statistics of all players that are present in the database.

2.Individual player stats

- This shows the statistics of a individual player.
- User is provided with a set of players that are present in the database from which he could select his choice.

3.Updating player stats

- This feature is used to update the statistics of the set of players that are present in the database.

4.Adding a new player

- This feature is used to add a new player to the database.
- But the statistics of the player are saved as 0 by default.
- So user has to update the newly added player stats using update feature.

5.Deleting a retired player

- This feature is used to delete a retired player from the database.
- Once deleted,all the statistics of the player are removed automatically.

6.Exiting the program

- After user completes all his queries he can exit the program by this option.

2.1 SOURCE FILE

```

import mysql.connector as sq
from prettytable import PrettyTable

formats_list = ['TEST', 'ODI', 'T20']
player_type_list = ['BATSMEN', 'BOWLERS']
type_dict = {'1': 'batsmen',
             '2': 'bowlers'}
format_dict = {
    '1': 'test',
    '2': 'odi',
    '3': 't20'
}
batsmen_orderby = ['Matches Played', 'Runs Scored', 'Strike Rate',
                   'Average', 'Half Centuries', 'Centuries', 'Highest
Score']
bowlers_orderby = ['Matches Played', 'Wickets Taken',
                   'Economy', '5 Wickets In A Match']
batsmen_orderby_dict = {
    '1': 'matchesPlayed',
    '2': 'runsScored',
    '3': 'strikeRate',
    '4': 'average',
    '5': 'halfCenturies',
    '6': 'centuries',
    '7': 'highestScore'}
bowlers_orderby_dict = {
    '1': 'matchesPlayed',
    '2': 'wicketsTaken',
    '3': 'economy',
    '4': '5wicketsInAMatch'}
order_type_dict = {
    '1': '',
    '2': 'desc'}
batsmen_fields = ['Name', 'DOB', 'Matches Played', 'Runs Scored',
                  'Strike Rate',
                  'Average', 'Half Centuries', 'Centuries', 'Highest
Score']
bowlers_fields = ['Name', 'DOB', 'Matches Played', 'Wickets Taken',
                  'Economy', '5 Wickets In A Match']

```

CRICKET DATABASE MANAGEMENT

```
def exit_program():
    """Exits the program when called"""
    print("Thank you for using our program")
    exit()

def establish_connection():
    """Establishes connection the the mysql database"""
    conn = sq.connect(host='localhost', user='root',
                      password='student', database='cricket')
    cursor = conn.cursor()

    return conn, cursor

def print_format_type():
    """Prints the format types to choose option"""
    print(f"--- FORMAT TYPE ---")
    for i in range(len(formats_list)):
        print(f'Press ({i + 1}) for', formats_list[i])
    print()
    return None

def print_player_type():
    """Prints the player types to choose an option"""
    print(f"--- PLAYER TYPE ---")
    for i in range(len(player_type_list)):
        print(f'Press ({i + 1}) for', player_type_list[i])
    print()
    return None

def get_order_based_on(player_type):
    """Prints the order types to choose an option"""
    if player_type == 'batsmen':
        """The order type is bowler"""
        while True:
            print(f"--- Order Value ---")
            for i in range(len(batsmen_orderby)):
                print(f"Press ({i + 1}) to order by
{batsmen_orderby[i]}")
            print()
            order_value_input = input("Enter the desired order number :
")
            if order_value_input in batsmen_orderby_dict:
```


CRICKET DATABASE MANAGEMENT

```

order_value = batsmen_orderby_dict[order_value_input]
while True:
    print("-- Order Type")
    print("Press (1) for ascending order")
    print("Press (2) for descending order")
    order_type_input = input("Enter the desired number :

```

")

```

    if order_type_input in order_type_dict:
        order_type = order_type_dict[order_type_input]
        return order_value, order_type
    else:
        print(
            f"<{order_type_input}> is an invalid input.

```

Please Try Again")

```

        continue

```

```

    else:

```

```

        print(
            f"<{order_value_input}> in an invalid order input.

```

Please Try Again")

```

        continue

```

```

elif player_type == 'bowlers':

```

```

    """The order type is bowlers"""

```

```

    while True:

```

```

        print(f"--- Order Value ---")

```

```

        for i in range(len(bowlers_orderby)):

```

```

            print(f"Press ({i + 1}) to order by

```

{bowlers_orderby[i]}")

```

            print()

```

```

            order_value_input = input("Enter the desired order number :

```

")

```

        if order_value_input in bowlers_orderby_dict:

```

```

            order_value = bowlers_orderby_dict[order_value_input]

```

```

            while True:

```

```

                print("-- Order Type")

```

```

                print("Press (1) for ascending order")

```

```

                print("Press (2) for descending order")

```

```

                order_type_input = input("Enter the desired number :

```

")

```

            if order_type_input in order_type_dict:

```

```

                order_type = order_type_dict[order_type_input]

```

```

                return order_value, order_type

```

```

            else:

```

```

                print(

```

```

                    f"<{order_type_input}> is an invalid input.

```

Please Try Again")

CRICKET DATABASE MANAGEMENT

```
        continue
    else:
        print(
            f"<{order_value_input}> in an invalid order input.
Please Try Again")
        continue
    else:
        return None, None

def get_format_and_type():
    """Prints the format and type to get options"""
    while True:
        print_format_type()
        format_input = input("Enter the format type : ")

        if format_input in format_dict:
            return_format = format_dict[format_input]
            print()
            while True:
                print_player_type()
                type_input = input("Enter the player type : ")
                if type_input in type_dict:
                    return_type = type_dict[type_input]
                    return return_format, return_type
                else:
                    print(
                        f"<{type_input}> is an invalid input. Please Try
Again\n")
                    continue
            else:
                print(f"<{format_input}> is an invalid input. Pleae Try
Again\n")
                continue

def get_players_name_from_type():
    print_player_type()
    usertype_input = input("Enter an option from above : ")
    if usertype_input in type_dict:
        usertype_input = type_dict[usertype_input]
    else:
        print(f"{usertype_input} is an invalid input. Please Try again")
    conn = sq.connect(host='localhost', user='root',
                      password='student', database='cricket')
    cursor = conn.cursor()
```

CRICKET DATABASE MANAGEMENT

```
data = cursor.execute(
    f"SELECT name from {'test'+usertype_input}")
data = cursor.fetchall()
nametable = PrettyTable()
playersname = []
nametable.field_names = ['Name']
for row in data:
    nametable.add_row([row[0]])
    playersname.append(row[0])

print(nametable)
gotname = False
while not gotname:
    nameinput = input("Please Enter a name to see the data :
").lower()
    if nameinput in playersname:
        gotname = True
        print(nameinput, usertype_input)
        return nameinput, usertype_input

    else:
        print(f"<{nameinput}> is not there in the table. Please try
again")

def see_format_table():
    format, type = get_format_and_type()
    order_value, order_type = get_order_based_on(type)
    if order_type == '':
        print(
            f"Table for format:{format}, type:{type},
order_value:{order_value}, order_type:ascending")
    else:
        print(
            f"Table for format:{format}, type:{type},
order_value:{order_value}, order_type:descending")

    conn = sq.connect(host='localhost', user='root',
                      password='student', database='cricket')
    cursor = conn.cursor()
    tablename = format.lower() + type.lower()
    cursor.execute(
        f"SELECT * FROM {tablename} ORDER BY {order_value}
{order_type}")
    data = cursor.fetchall()
    table = PrettyTable()
```

CRICKET DATABASE MANAGEMENT

```
if type == 'batsmen':
    field_names_list = batsmen_fields
else:
    field_names_list = bowlers_fields
for row in data:
    row = list(row)
    table.field_names = field_names_list
    table.add_row(row)
print(table)
conn.close()
```

```
def print_player_table(playername, playertype):
    """user can see all the data of the player"""
    conn = sq.connect(host='localhost', user='root',
                      password='student', database='cricket')
    cursor = conn.cursor()

    if playertype == 'batsmen':
        querystring = "SELECT
matchesPlayed,runsScored,strikeRate,average,halfCenturies,centuries,highestScore FROM {} WHERE name = '{}'"
        datatable = PrettyTable()
        datatable.add_column('fields\\formats', batsmen_orderby)
    else:
        querystring = "SELECT
matchesPlayed,wicketsTaken,economy,5wicketsInAMatch FROM {} WHERE name = '{}'"
        datatable = PrettyTable()
        datatable.add_column('fields\\formats', bowlers_orderby)

    for type in formats_list:
        cursor.execute(querystring.format(type.lower()+playertype,
playername))
        data = cursor.fetchall()
        datatable.add_column(type, list(data[0]))
    conn.close()
    print(datatable)
```

```
def see_player_stats():
    "user can see the stats of a player"
    playername, playertype = get_players_name_from_type()
    conn = sq.connect(host='localhost', user='root',
                      password='student', database='cricket')
    cursor = conn.cursor()
```

CRICKET DATABASE MANAGEMENT

```
cursor.execute(
    f"SELECT dob FROM {'odi'+playertype} WHERE name = '{playername}'")
dob = cursor.fetchone()[0]
conn.close()
print(f"\nPlayer Name : {playername}")
print(f"Type : {playertype}")
print(f"DOB : {dob}\n")
print_player_table(playername, playertype)

return playername, playertype

def get_date():
    """user enters the date of the player"""
    print("- DATE INPUT -")
    day = input("Enter the day : ")
    month = input("Enter the month ('jan' for january) : ")
    year = input("Enter the year : ")
    date = f"{day} {month} {year}"
    return date

def delete_player():
    """user can delete a retired player from the database"""
    conn = sq.connect(host='localhost', user='root',
                      password='student', database='cricket')
    cursor = conn.cursor()

    plr_format, plr_type = get_format_and_type()
    get_table_name = plr_format+plr_type
    cursor.execute(f"select name from {get_table_name}")
    player_name = cursor.fetchall()
    print("Available players in the database")
    player_table = PrettyTable()
    player_table.field_names = ['Player Name']
    for row in player_name:
        player_table.add_row(row)

    print(player_table)
    print('\n')

    player_input = input("Enter the player name : ")
    check_val = False
    for row in player_name:
```

CRICKET DATABASE MANAGEMENT

```
if row[0].lower() == player_input.lower():
    check_val = True
    break

if check_val:
    print("Player match found.\n")
    confirm_input = input("Do you want to delete the player (Y / N)
: ")
    if confirm_input.lower() == 'y':
        try:
            cursor.execute(f"delete from {get_table_name} where name
= '{player_input}'")
            conn.commit()
            print(f"You have successfully deleted <{player_input}>
from {plr_format} table")
        except conn.Error as err:
            print("The following error as occurred")
            print(err)
            conn.close()
            return None

    else:
        print("You chose not to delete the player.")
        conn.close()
        return None

else:
    print("The enetered player does not exists. Please Try again")
    return None

conn.close()

def add_new_player():
    """user can add a player to the database"""
    print("You chose to add a new player to the database\n")
    conn = sq.connect(host='localhost', user='root',
                      password='student', database='cricket')
    cursor = conn.cursor()
    plr_format, plr_type = get_format_and_type()
    get_table_name = 'odi'+plr_type
    cursor.execute(f"select name from {get_table_name}")
    player_name = cursor.fetchall()
    print("Available players in the database")
```

CRICKET DATABASE MANAGEMENT

```
player_table = PrettyTable()
player_table.field_names = ['Player Name']
for row in player_name:
    player_table.add_row(row)

print(player_table)
print('\n')
player_input = input("Enter the player name : ")
for name in player_name:
    if name[0].lower() == player_input.lower():
        print(f"<{player_input}> already exists in the database.
Please update the data")
        conn.close()
        return None
player_dob = get_date()

if plr_type == 'batsmen':
    cursor.execute(f"insert into {'test'+plr_type} values
('{player_input}','{player_dob}',{0},{0},{0.0},{0.0},{0},{0},{0})")
    cursor.execute(f"insert into {'odi'+plr_type} values
('{player_input}','{player_dob}',{0},{0},{0.0},{0.0},{0},{0},{0})")
    cursor.execute(f"insert into {'t20'+plr_type} values
('{player_input}','{player_dob}',{0},{0},{0.0},{0.0},{0},{0},{0})")

elif plr_type == 'bowlers':
    cursor.execute(f"insert into {'test'+plr_type} values
('{player_input}','{player_dob}',{0},{0},{0.0},{0})")
    cursor.execute(f"insert into {'odi'+plr_type} values
('{player_input}','{player_dob}',{0},{0},{0.0},{0})")
    cursor.execute(f"insert into {'t20'+plr_type} values
('{player_input}','{player_dob}',{0},{0},{0.0},{0})")

conn.commit()
print("The data has been added sucessfully. To update the stats use
<update option>")

conn.close()

def get_specific_value(playername, playertype, playerformat,
ordervalue):
    conn = sq.connect(host='localhost', user='root',
                      password='student', database='cricket')
    cursor = conn.cursor()
    cursor.execute(
```

CRICKET DATABASE MANAGEMENT

```

        f"SELECT {ordervalue} FROM {playerformat+playertype} WHERE name
= '{playername}'")
        value = cursor.fetchone()
        conn.close()
        return value

```

```

def update_player_stats():
    """user can update stats of a player"""
    playername, playertype = see_player_stats()
    while True:
        print_format_type()
        format_input = input("Select a format to update : ")
        if format_input in format_dict:
            playerformat = format_dict[format_input]
        else:
            print(f"<{format_input}> is invalid format. Please Try
Again")
            continue
        print(playerformat)
        if playertype == 'batsmen':
            playerlist = batsmen_orderby
            playerdict = batsmen_orderby_dict
        else:
            playerlist = bowlers_orderby
            playerdict = bowlers_orderby_dict
        while True:
            print(f"--- Player Data ---")
            for i in range(len(playerlist)):
                print(f"Press ({i + 1}) to update {playerlist[i]}")
            print()
            order_value_input = input("Enter the desired order number :
")
            if order_value_input in playerdict:
                ordervalue = playerdict[order_value_input]
            else:
                print(f"<{order_value_input}> is invalid. Please Try
Again")
                continue
            present_value = get_specific_value(
                playername, playertype, playerformat, ordervalue)

            print(f"The current value of {ordervalue} is
{present_value[0]}")

```


CRICKET DATABASE MANAGEMENT

```

confirm_input = input("Are you sure you want to update (Y /
N) : ")
if confirm_input.lower() == 'y':
    value_input = input(f"Enter the new value for
{ordervalue} : ")
    if isinstance(present_value, int):
        try:
            value_input = int(value_input)
        except ValueError:
            print(
                f"{value_input} is invalid data for
updating. Please Try Again")
            continue
    else:
        try:
            value_input = float(value_input)
        except ValueError:
            print(
                f"{value_input} is invalid data for
updating. Please Try Again")
            continue
    conn = sq.connect(host='localhost', user='root',
                      password='student',
database='cricket')
    cursor = conn.cursor()
    try:
        cursor.execute(
            f"UPDATE {playerformat+playertype} SET
{ordervalue} = {value_input} WHERE name = '{playername}'")
        conn.commit()
    except sq.Error:
        print("An Error Occurred while updating the data.
Please Try Again")
        return None
    print("#"*20)
    print("Data Updated Sucessfully")
    print("#"*20, '\n')
    print_player_table(playername, playertype)
    break

else:
    print("You chose not to update the data.")
    return None

update_more = input("Do you want to update more (Y / N) : ")
if update_more.lower() == 'y':

```

```

        continue
    else:
        return None

```

2.2 MAIN FILE

```

import os
import src

```

```

heading = """____SAINIK SCHOOL KALIKIRI____
<<<<<-- ICC CRICKET DATABASE MANAGEMENT MANAGEMENTMANAGEMENT-->>>>>\n"""

```

```

menu = """\n>>>>> MAIN MENU <<<<<
Press (1) to see the format table
Press (2) to see the player stats
Press (3) to update the player stats
Press (4) to add a new player
Press (5) to delete a retired player
Press (6) to exit the program\n"""

```

```

menu_dict = {
    '1': src.see_format_table,
    '2': src.see_player_stats,
    '3': src.update_player_stats,
    '4': src.add_new_player,
    '5': src.delete_player,
    '6': src.exit_program
}

```

```

def main():
    os.system('cls')
    print(heading)

    while True:
        print(menu)

        user_input = input("Enter your choice from the above options : ")

        if user_input in menu_dict:
            menu_dict[user_input]()

        else:
            print("You have entered an invalid input. Please try again")
            continue

```

```
if __name__ == '__main__':
    main()
```

3.OUTPUT SCREENS

O.HOME MENU

```
____SAINIK SCHOOL KALIKIRI____
<<<<<-- ICC CRICKET DATABASE -->>>>>
```

```
>>>>> MAIN MENU <<<<<
Press (1) to see the format table
Press (2) to see the player stats
Press (3) to update the player stats
Press (4) to add a new player
Press (5) to delete a retired player
Press (6) to exit the program
```

Enter your choice from the above options :

1. FORMAT TABLE FEATURE

```
Enter your choice from the above options : 1
--- FORMAT TYPE ---
Press (1) for TEST
Press (2) for ODI
Press (3) for T20

Enter the format type :
```

1.1 OPTION 1(TEST)

Enter the format type : 1

```
--- PLAYER TYPE ---
Press (1) for BATSMEN
Press (2) for BOWLERS
```

Enter the player type :

1.1.1 OPTION 1(BATSMEN)

Table for format:test, type:batsmen, order_value:runsScored, order_type:ascending

Name	DOB	Matches Played	Runs Scored	Strike Rate	Average	Half Centuries	Centuries	Highest Score
aaron finch	17 nov 1986	5	278	443.98	27.80	2	0	62
k l rahul	18 apr 1992	36	2006	56.46	34.59	11	5	199
rohit sharma	30 apr 1987	32	2141	59.26	46.54	10	6	212
shikhar dhawan	05 dec 1985	34	2315	66.95	40.61	5	7	190
ab devilliers	12 dec 1987	180	3000	80.54	38.98	70	18	202
m s dhoni	07 jul 1981	90	4876	59.12	38.09	33	6	224
kane williamson	08 aug 1990	77	6370	51.57	52.21	31	21	242
david warner	27 oct 1986	82	7009	73.21	48.34	30	23	335
steven smith	02 jun 1989	71	7072	55.98	63.14	27	26	239
virat kohli	05 nov 1988	88	7202	57.81	54.98	22	27	254
chris gayle	21 sep 1979	103	7215	60.28	42.19	37	15	333
sachin tendulkar	25 apr 1973	200	15921	54.08	53.79	68	51	248

- IN THIS FEATURE THE ORDER OF THE ARRANGEMENT CAN BE CHOSEN.

2. VIEWING PLAYER STATS

```

Enter your choice from the above options : 2
--- PLAYER TYPE ---
Press (1) for BATSMEN
Press (2) for BOWLERS

Enter an option from above :
```

2.1 PLAYER STATS(BOWLER)

CRICKET DATABASE MANAGEMENT

Enter an option from above : 2

Name
bhuvaneshwar kumar
dale steyn
imran tahir
jasprit bumrah
lasith malinga
mittchell starc
mohammed shami
rangana herath
ravichandran ashwin
trent bourt
yuzvendra chahal

Please Enter a name to see the data :

2.1.1 INDIVIDUAL STATS

Please Enter a name to see the data : jasprit bumrah
jasprit bumrah bowlers

Player Name : jasprit bumrah
Type : bowlers
DOB : 06 dec 1993

fields\formats	TEST	ODI	T20
Matches Played	12	61	42
Wickets Taken	62	104	51
Economy	2.54	4.49	6.72
5 Wickets In A Match	5	1	0

3. UPDATING A PLAYER STATS (BATSMEN-K L RAHUL,ODI,CENTURIES)

CRICKET DATABASE MANAGEMENT

Name
aaron finch
ab devilliers
chris gayle
david warner
k l rahul
kane williamson
m s dhoni
rohit sharma
sachin tendulkar
shikhar dhawan
steven smith
virat kohli

Please Enter a name to see the data : k l rahul
k l rahul batsmen

Player Name : k l rahul
Type : batsmen
DOB : 18 apr 1992

fields\formats	TEST	ODI	T20
Matches Played	36	28	34
Runs Scored	2006	997	1192
Strike Rate	56.46	80.48	146.46
Average	34.59	40.60	43.77
Half Centuries	11	5	9
Centuries	5	4	2
Highest Score	199	111	110

Select a format to update : 2
odi

--- Player Data ---

Press (1) to update Matches Played
Press (2) to update Runs Scored
Press (3) to update Strike Rate
Press (4) to update Average
Press (5) to update Half Centuries
Press (6) to update Centuries
Press (7) to update Highest Score

Enter the desired order number : 6
The current value of centuries is 4
Are you sure you want to update (Y / N) : Y
Enter the new value for centuries : 6

3.1 UPDATED STATS

CRICKET DATABASE MANAGEMENT

```
#####  
Data Updated Sucessfully  
#####
```

fields\formats	TEST	ODI	T20
Matches Played	36	28	34
Runs Scored	2006	997	1192
Strike Rate	56.46	80.48	146.46
Average	34.59	40.60	43.77
Half Centuries	11	5	9
Centuries	5	6	2
Highest Score	199	111	110

Do you want to update more (Y / N) :

5. ADDING A NEW PLAYER TO THE DATABASE

Available players in the databas

Player Name
aaron finch
ab devilliers
chris gayle
david warner
k l rahul
kane williamson
m s dhoni
rohit sharma
sachin tendulkar
shikhar dhawan
steven smith
virat kohli

4.1 ADDING VALUES (joe root,2 feb 1984)

```
Enter the player name : joe root  
- DATE INPUT -  
Enter the day : 2  
Enter the month ('jan' for january) : feb  
Enter the year : 1984  
The data has been added sucessfully. To update the stats use <update option>
```

- All the values of matches played,runs scored etc, are saved as 0 by default and those values are to be updated using update feature

6. DELETING A RETIRED PLAYER

CRICKET DATABASE MANAGEMENT

Enter the player type : 1

Available players in the database

Player Name
aaron finch
ab devilliers
chris gayle
david warner
joe root
k l rahul
kane williamson
m s dhoni
rohit sharma
sachin tendulkar
shikhar dhawan
steven smith
virat kohli

Enter the player name : ab devilliers

Player match found.

5.1 DELETING PLAYER (ab devilliers from all formats)

Do you want to delete the player (Y / N) : y

You have successfully deleted <ab devilliers> from test table

6.EXITING PROGRAM

Enter your choice from the above options : 6

Thank you for using our program

<<<<END OF THE REPORT>>>>

4. BIBLIOGRAPHY

1. Computer Science with Python [Textbook XII] by Sumita Arora
2. <https://ptable.readthedocs.io/en/latest/>
3. <https://docs.python.org/3/library/getpass.html>