

Name – JEEVA D

Email: [jiieevad@gmail.com](mailto:jiieevad@gmail.com)

Mobile: 91 9206403317

## SQL Projects

### Database - Pelican Store

**Q1.What is the most common method of payment among the customers?**

```
SELECT method_of_payment ,COUNT(*) AS Total_Count FROM PelicanStore
```

```
GROUP BY method_of_payment
```

```
ORDER BY Total_count DESC
```

```
LIMIT 1;
```

The screenshot shows a SQL IDE interface with a sidebar on the left and a main workspace on the right. The sidebar contains a menu with options: "Add DataBase", "SQLite 0.1.3 beta", "Table", "PelicanStore", "MariaDB", "PostgreSQL", and "MS SQL". The main workspace displays a SQL query in a text editor and its results in a table below it.

**SQL Query:**

```
1 ---1. What is the most common method of payment among the customers?
2 SELECT method_of_payment ,COUNT(*) AS Total_Count FROM PelicanStore
3 GROUP BY method_of_payment
4 ORDER BY Total_count DESC
5 LIMIT 1;
```

**Query Results:**

Method_of_Payment	Total_Count
Proprietary Card	70

Q2.What is the age distribution of the customers and how does it affect their spending habits?

SELECT age ,Avg(items) AS Average\_items ,Avg(sales) AS Average\_sales FROM PelicanStore

GROUP by age

ORDER BY age;

Pricing

→ "Add DataBase"

Create a database linked to your account.

*This is only available with a paid subscription.*

SQLite

0.1.3 beta

Table

PelicanStore

Column

Customer INTEGER

Method\_of\_Payment...

Items INTEGER

Discount\_ REAL

Sales REAL

Gender TEXT

Marital\_Status TEXT

Age INTEGER

MariaDB

PostgreSQL

Day 1

Day 2

Day 3

Day 4

Project

13

14

15

16

17

18 ---What is the age distribution of the customers and how does it affect their spending habits?

19 SELECT age ,Avg(items) AS Average\_items ,Avg(sales) AS Average\_sales FROM PelicanStore

20 GROUP BY age

21 ORDER BY age;

22

23

24

Age	Average_items	Average_sales
20	1.5	52.8
22	2	81.5
24	1	49.5
28	3.4	90.62800000000001
30	5.875	117.52125
32	2	42.565
34	1.666666666666667	51.166666666666664
36	2.1666666666666665	68.88333333333334
38	3.6	99.928

### Q3. What is the retention rate of customers based on their method of payment?

SELECT method\_of\_payment ,COUNT(DISTINCT customer) as Unique\_Customers FROM PelicanStore

GROUP BY method\_of\_payment

HAVING COUNT(DISTINCT customer)>4;

The screenshot shows a database management interface with a sidebar on the left and a main workspace on the right. The sidebar contains a section for "Add DataBase" and a list of databases: SQLite (0.1.3 beta), MariaDB, and PostgreSQL. The SQLite database is selected, and the "PelicanStore" table is visible. The main workspace displays a SQL query in a text editor, which is the same query provided in the text blocks. Below the editor, the results of the query are shown in a table format.

Method_of_Payment	Unique_Customers
Mastercard	14
Proprietary Card	70
Visa	10

#### Q4. How does age and marital status affect the number of transactions made by customers?

SELECT age,marital\_status ,COUNT(\*)as number\_of\_transactions FROM PelicanStore

GROUP by age,marital\_status

order BY age ,marital\_status;

The screenshot shows a database management interface. On the left, there's a sidebar with a menu. The main area displays a SQL query and its results.

**SQL Query:**

```
--How does age and marital status affect the number of transactions made by customers?
SELECT age,marital_status ,COUNT(*)AS number_of_transactions FROM PelicanStore
GROUP BY age,marital_status
ORDER BY age ,marital_status;
```

**Results Table:**

Age	Marital_Status	number_of_transactions
20	Single	2
22	Single	2
24	Single	1
28	Married	4
28	Single	1
30	Married	6
30	Single	2
32	Married	8
34	Married	3

### Q5. How does the average discount vary by gender?

SELECT gender ,Avg(discount\_) AS Average\_Discount from PelicanStore

GROUP BY gender;

The screenshot shows a database query editor interface. On the left, there is a sidebar with a menu for "Add DataBase" and a list of databases: SQLite (selected), MariaDB, and PostgreSQL. Under SQLite, there is a table named "PelicanStore" with columns: Customer (INTEGER), Method\_of\_Payment..., Items (INTEGER), Discount\_ (REAL), Sales (REAL), Gender (TEXT), Marital\_Status (TEXT), and Age (INTEGER). The main editor area shows the SQL query: `--How does the average discount vary by gender?`  
`SELECT gender ,Avg(discount_) AS Average_Discount FROM PelicanStore`  
`GROUP BY gender;` The results are displayed in a table with two columns: "Gender" and "Average\_Discount". The results are: Female with an average discount of 23.16752688172043, and Male with an average discount of 12.942857142857145.

Gender	Average_Discount
Female	23.16752688172043
Male	12.942857142857145

## Database – World\_cup\_data

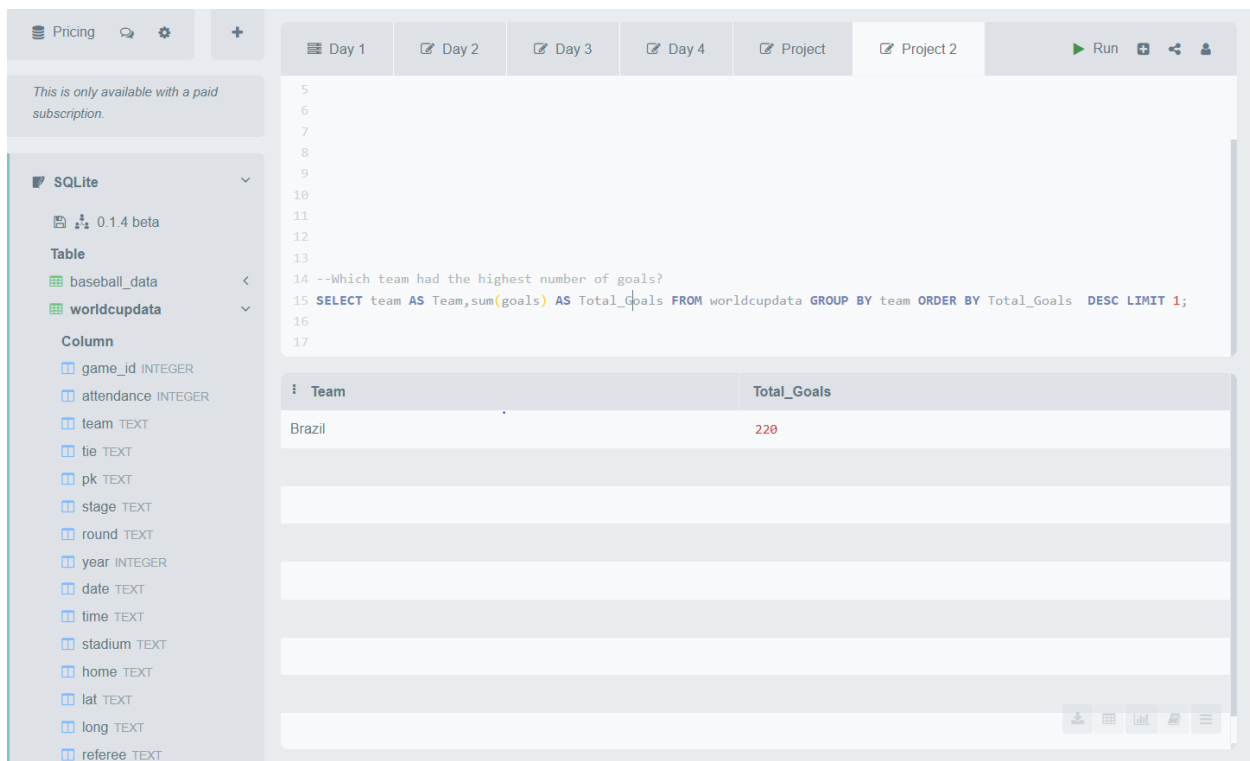
### Q6. Which team had the highest number of goals

SELECT team AS Team,sum(goals) AS Total\_Goals FROM worldcupdata

GROUP BY team

ORDER BY Total\_Goals DESC

LIMIT 1;



The screenshot shows a SQL query editor interface. On the left, there is a sidebar with a tree view showing the database structure: SQLite, 0.1.4 beta, Table, baseball\_data, worldcupdata, and a list of columns for worldcupdata including game\_id, attendance, team, tie, pk, stage, round, year, date, time, stadium, home, lat, long, and referee. The main area displays the SQL query: `--Which team had the highest number of goals?`  
`SELECT team AS Team,sum(goals) AS Total_Goals FROM worldcupdata GROUP BY team ORDER BY Total_Goals DESC LIMIT 1;` Below the query, the results are shown in a table with two columns: Team and Total\_Goals. The first row shows Brazil with 229 goals. The interface also includes a top bar with tabs for Day 1 through Day 4, Project, and Project 2, and a Run button.

Team	Total_Goals
Brazil	229

## Q7.Team Performance over the Years

SELECT team ,sum(goals) AS goals\_scored FROM worldcupdata

GROUP BY year,team

ORDER BY goals\_scored DESC;

Pricing

+

This is only available with a paid subscription.

SQLite

0.1.4 beta

Table

baseball\_data

worldcupdata

Column

game\_id INTEGER

attendance INTEGER

team TEXT

tie TEXT

pk TEXT

stage TEXT

round TEXT

year INTEGER

date TEXT

time TEXT

stadium TEXT

home TEXT

lat TEXT

long TEXT

referee TEXT

Day 1

Day 2

Day 3

Day 4

Project

Project 2

Run

16

17

18

19

20

21

22

23

24 --Team Performance Over the Years

25

26 SELECT team ,sum(goals) AS goals\_scored FROM worldcupdata GROUP BY year,team ORDER BY goals\_scored DESC;

27

28

team	goals_scored
Hungary	27
FRG	25
France	23
Brazil	22
Brazil	19
Argentina	18
Germany	18
Austria	17
Portugal	17

## Q8. Stadium Statistics

### Games Hosted by Each Stadium

SELECT stadium ,COUNT (DISTINCT game\_id ) As games\_hosted FROM worldcupdata

GROUP by stadium

ORDER BY games\_hosted DESC;

Pricing

This is only available with a paid subscription.

SQLite

0.1.4 beta

Table

baseball\_data

worldcupdata

Column

game\_id INTEGER

attendance INTEGER

team TEXT

tie TEXT

pk TEXT

stage TEXT

round TEXT

year INTEGER

date TEXT

time TEXT

stadium TEXT

home TEXT

lat TEXT

long TEXT

referee TEXT

Day 1

Day 2

Day 3

Day 4

Project

Project 2

Run

```
-- SELECT team ,SUM(game_id) AS games_hosted FROM worldcupdata GROUP BY team ORDER BY games_hosted DESC
20
21
22
23
24
25
26
27
28 --Stadium Statistics
29 -- Games Hosted by Each Stadium
30 SELECT stadium ,COUNT (DISTINCT game_id ) AS games_hosted FROM worldcupdata GROUP BY stadium ORDER BY games_hosted DESC
31
32
```

stadium	games_hosted
Estadio Azteca (MÃ©xico D.F.)	19
Estadio Jalisco (Guadalajara)	15
Waldstadion (Frankfurt am Main)	10
Estadio del Centenario (Montevideo)	10
Estadio Nacional (Santiago de Chile)	10
Wembley (London)	9
VÃ©lodrome (Marseille)	9
Parc des Princes (Paris)	9
Parc Lescure (Bordeaux)	9



## Q9.Geographic and Temporal Insights

### Performance by Match Time

SELECT time,team ,Avg(goals) AS Average\_goals FROM worldcupdata

GROUP by time,team

ORDER by Average\_goals DESC;

Pricing

This is only available with a paid subscription.

SQLite

0.1.4 beta

Table

baseball\_data

worldcupdata

Column

game\_id INTEGER

attendance INTEGER

team TEXT

tie TEXT

pk TEXT

stage TEXT

round TEXT

year INTEGER

date TEXT

time TEXT

stadium TEXT

home TEXT

lat TEXT

long TEXT

referee TEXT

Day 1

Day 2

Day 3

Day 4

Project

Project 2

Run

```
32
33
34
35
36
37
38
39 --Geographic and Temporal Insights
40 --Performance by Match Time
41 SELECT time,team ,Avg(goals) AS Average_goals FROM worldcupdata
42 GROUP BY time,team
43 ORDER BY Average_goals DESC;
44
```

time	team	Average_goals
16:50	Hungary	8
13:30	Portugal	7
16:50	Uruguay	7
17:00	Turkey	7
14:45	Uruguay	6
18:00	Hungary	6
18:00	FRG	5.666666666666667
17:30	Argentina	5
17:30	Poland	5

## Q10 .Home Advantage Analysis

SELECT home , COUNT(game\_id) AS home\_games ,AVG(goals) AS Avg\_home\_games FROM  
worldcupdata

WHERE home =team

GROUP BY home

ORDER BY Avg\_home\_games DESC;

Pricing

+

This is only available with a paid subscription.

SQLite

0.1.4 beta

Table

baseball\_data

worldcupdata

Column

game\_id INTEGER

attendance INTEGER

team TEXT

tie TEXT

pk TEXT

stage TEXT

round TEXT

year INTEGER

date TEXT

time TEXT

stadium TEXT

home TEXT

lat TEXT

long TEXT

referee TEXT

Day 1

Day 2

Day 3

Day 4

Project

Project 2

Run

--

48

49

50

51

52

53

54

--Home Advantage Analysis

56 SELECT home , COUNT(game\_id) AS home\_games ,AVG(goals) AS Avg\_home\_games FROM worldcupdata

57 WHERE home =team

58 GROUP BY home

59 ORDER BY Avg\_home\_games DESC;

home	home_games	Avg_home_games
Uruguay	4	3.75
Switzerland	4	2.75
Brazil	13	2.5384615384615383
Argentina	7	2.142857142857143
France	9	2.111111111111111
Sweden	6	2
Germany	7	2
Italy	12	1.8333333333333333
England	6	1.8333333333333333

### Q11. Predictive Analysis: Predicting if a match will end in a tie based on other factors

```
SELECT time ,team,attendance,home ,goals ,CASE WHEN goals = (SELECT avg(goals) FROM
worldcupdata) THEN 'Likely_Tie' ELSE'Not_Likely_Tie' END AS tie_predication

FROM worldcupdata;
```

Pricing

This is only available with a paid subscription.

SQLite

0.1.4 beta

Table

baseball\_data

worldcupdata

Column

game\_id INTEGER

attendance INTEGER

team TEXT

tie TEXT

pk TEXT

stage TEXT

round TEXT

year INTEGER

date TEXT

time TEXT

stadium TEXT

home TEXT

lat TEXT

long TEXT

referee TEXT

Day 1

Day 2

Day 3

Day 4

Project

Project 2

Run

```
--
52
53 |
54 --Predictive AnalysisExample: Predicting if a match will end in a tie based on other factors
55 SELECT time ,team,attendance,home ,goals ,CASE WHEN goals = (SELECT avg(goals) FROM worldcupdata) THEN 'Likely_Tie' ELSE
56 FROM worldcupdata;
57
58
59
60
61
62
63
```

time	team	attendance	home	goals	tie_predication
16:00	Italy	25000	Italy	7	Not_Likely_Tie
16:00	USA	25000	Italy	1	Not_Likely_Tie
16:30	Austria	16000	Italy	3	Not_Likely_Tie
16:30	France	16000	Italy	2	Not_Likely_Tie
16:30	Germany	8000	Italy	5	Not_Likely_Tie
16:30	Belgium	8000	Italy	2	Not_Likely_Tie
16:30	Czechoslovakia	9000	Italy	2	Not_Likely_Tie
16:30	Romania	9000	Italy	1	Not_Likely_Tie
16:30	Switzerland	33000	Italy	3	Not_Likely_Tie

## Q12.Seasonal Performance Trends

SELECT year ,team ,avg(goals) AS Average\_golas FROM worldcupdata

GROUP by year,team

ORDER by Average\_golas DESC;

Pricing

This is only available with a paid subscription.

SQLite

0.1.4 beta

Table

baseball\_data

worldcupdata

Column

game\_id INTEGER

attendance INTEGER

team TEXT

tie TEXT

pk TEXT

stage TEXT

round TEXT

year INTEGER

date TEXT

time TEXT

stadium TEXT

home TEXT

lat TEXT

long TEXT

referee TEXT

Day 1

Day 2

Day 3

Day 4

Project

Project 2

Run

51

52

53

54 --Predictive AnalysisExample: Predicting if a match will end in a tie based on other factors

55 --SELECT time ,team,attendance,home ,goals ,CASE WHEN goals = (SELECT avg(goals) FROM worldcupdata) THEN 'Likely\_Tie' t

56 --FROM worldcupdata;

57 --Seasonal Performance Trends

58 SELECT year ,team ,avg(goals) AS Average\_golas FROM worldcupdata

59 GROUP BY year,team

60 ORDER BY Average\_golas DESC;

61

62

63

i	year	team	Average_golas
	1954	Hungary	5.4
	1938	Poland	5
	1954	FRG	4.166666666666667
	1958	France	3.8333333333333335
	1930	Uruguay	3.75
	1938	Hungary	3.75
	1950	Uruguay	3.75
	1938	Sweden	3.6666666666666665
	1950	Brazil	3.6666666666666665

## Q13.Attendance Analysis

### Min, Max, and Average Attendance

```
SELECT min(attendance) AS Minimum_attendance ,Max(attendance) AS Maximum_attendance  
,Avg(attendance) AS Average_attendance FROM worldcupdata;
```

The screenshot shows a database query interface. On the left, there is a sidebar with a 'SQLite' section containing a list of tables: 'baseball\_data' and 'worldcupdata'. The 'worldcupdata' table is selected, and its columns are listed: game\_id (INTEGER), attendance (INTEGER), team (TEXT), tie (TEXT), pk (TEXT), stage (TEXT), round (TEXT), year (INTEGER), date (TEXT), time (TEXT), stadium (TEXT), home (TEXT), lat (TEXT), long (TEXT), and referee (TEXT). The main area displays the SQL query: `SELECT min(attendance) AS Minimum_attendance ,Max(attendance) AS Maximum_attendance ,Avg(attendance) AS Average_attendance FROM worldcupdata;`. Below the query, the results are shown in a table with three columns: 'Minimum\_attendance', 'Maximum\_attendance', and 'Average\_attendance'. The values are 2000, 179854, and 44651.4019138756 respectively. The interface also includes a 'Run' button and a 'Pricing' section at the top.

Minimum_attendance	Maximum_attendance	Average_attendance
2000	179854	44651.4019138756

## Database – baseball data

### Q14. List the top 5 players by most home runs

SELECT name,HR FROM baseball\_data ORDER by HR DESC LIMIT 5;

The screenshot shows a database management interface with a sidebar on the left listing tables in the 'chinook.db' database. The 'baseball\_data' table is selected. The main area displays a SQL query editor with the following query:

```
--- List the top 5 players by most home runs
SELECT name ,HR FROM baseball_data ORDER BY hr DESC LIMIT 5;
```

Below the query editor, the results are displayed in a table with two columns: 'name' and 'HR'.

name	HR
Reggie Jackson	563
Mike Schmidt	548
Willie Stargell	475
Carl Yastrzemski	452
Dave Kingman	442

## Q15. Correlation Between Height, Weight, and Performance

SELECT height, weight, AVG(avg) AS average\_batting\_average, AVG(HR) AS average\_home\_runs

FROM baseball\_data

GROUP BY height, weight

ORDER BY average\_batting\_average DESC, average\_home\_runs DESC;

The screenshot shows a database query editor interface. On the left, a sidebar displays the database structure for 'chinook.db', including tables like 'albums', 'artists', 'baseball\_data', 'customers', 'employees', 'genres', 'IMDBMovieData', 'invoices', 'invoice\_items', 'media\_types', 'playlists', 'playlist\_track', and 'sqlite\_sequence'. The 'baseball\_data' table is selected. The main area shows the SQL query:   
---Correlation Between Height, Weight, and Performance  
SELECT height, weight, AVG(avg) AS average\_batting\_average, AVG(HR) AS average\_home\_runs  
FROM baseball\_data  
GROUP BY height, weight  
ORDER BY average\_batting\_average DESC, average\_home\_runs DESC;  
The results are displayed in a table with 4 columns: height, weight, average\_batting\_average, and average\_home\_runs. The data is sorted by average\_batting\_average in descending order, then by average\_home\_runs in descending order.

height	weight	average_batting_average	average_home_runs
71	192	0.303	160
74	212	0.297	252
72	201	0.295	184
70	192	0.294	272
71	193	0.285	248
67	150	0.279	36
71	172	0.279	22
71	187	0.27749999999999997	158
76	170	0.277	60

## Q.16 Handedness Effect on Performance

SELECT handedness ,AVG(avg) AS Batting\_Average ,AVG(hr) AS Average\_home\_runs FROM  
baseball\_data

GROUP BY handedness

ORDER BY Batting\_Average DESC , Average\_home\_runs DESC;

The screenshot shows a SQL query editor interface. On the left, a sidebar displays the database schema for 'chinook.db', including tables like 'albums', 'artists', 'baseball\_data', and columns like 'name', 'handedness', 'height', 'weight', 'avg', 'hr'. The main editor area contains the following SQL query:

```
--Handedness Effect on Performance
SELECT handedness ,AVG(avg) AS Batting_Average ,AVG(hr) AS Average_home_runs FROM baseball_data
GROUP BY handedness
ORDER BY Batting_Average DESC , Average_home_runs DESC;
```

Below the query editor, the results are displayed in a table with three columns: 'handedness', 'Batting\_Average', and 'Average\_home\_runs'. The results are ordered by 'Batting\_Average' in descending order.

handedness	Batting_Average	Average_home_runs
B	0.20504807692307692	32.14423076923077
L	0.20451265822784812	56.14873417721519
R	0.1766200814111262	42.59837177747625



## Q17 .Top 10 Players by Batting Average or Home Runs

SELECT name,avg,hr FROM baseball\_data

ORDER by avg DESC ,hr DESC

LIMIT 10;

The screenshot shows a database management interface with a sidebar on the left listing tables and columns for a database named 'chinook.db'. The main area displays a SQL query and its results. The query is as follows:

```
--Top 10 Players by Batting Average or Home Runs
SELECT name,avg,hr FROM baseball_data
ORDER BY avg DESC ,hr DESC
LIMIT 10;
```

The results are displayed in a table with the following data:

name	avg	HR
Rod Carew	0.328	92
Lyman Bostock	0.311	23
Matty Alou	0.307	31
Ralph Garr	0.306	75
Bill Madlock	0.305	163
Tony Oliva	0.304	220
Manny Mota	0.304	31
Al Oliver	0.303	219
Pete Rose	0.303	160

## Q.18 Players with the Lowest Batting Averages and No Home Runs

SELECT name,avg,hr from baseball\_data

where HR =0

ORDER by avg

limit 10 ;

The screenshot shows a database management interface with a sidebar on the left listing tables in the 'chinook.db' database. The main area displays a SQL query and its results. The query is:   
SELECT name,avg,hr FROM baseball\_data  
WHERE HR =0  
ORDER BY avg  
LIMIT 10 ;  
The results table shows 10 players with a batting average of 0 and 0 home runs, ordered by their batting average. The players are: Tom Brown, Ozzie Osborn, John LaRose, Jeff Byrd, Jerry Kutzler, Jeff Barkley, Eric Wilkins, Britt Burns, and Nardi Contreras.

name	avg	HR
Tom Brown	0	0
Ozzie Osborn	0	0
John LaRose	0	0
Jeff Byrd	0	0
Jerry Kutzler	0	0
Jeff Barkley	0	0
Eric Wilkins	0	0
Britt Burns	0	0
Nardi Contreras	0	0

## Q19.Distribution of Handedness

```
SELECT handedness ,COUNT(*) As Player_count
```

```
FROM baseball_data
```

```
GROUP BY handedness
```

```
ORDER by Player_count DESC;
```

The screenshot shows a SQL query editor interface. On the left, a sidebar displays the database structure for 'chinook.db', including tables like 'albums', 'artists', 'baseball\_data', 'customers', 'employees', 'genres', and 'IMDBMovieData'. The 'baseball\_data' table is selected, showing its columns: name (TEXT), handedness (TEXT), height (INTEGER), weight (INTEGER), avg (REAL), and HR (INTEGER). The main editor area contains the following SQL query:

```
57  
58  
59  
60  
61  
62  
63 ---Distribution of Handednes  
64 SELECT handedness ,COUNT(*) AS Player_count  
65 FROM baseball_data  
66 GROUP BY handedness  
67 ORDER BY Player_count DESC  
68
```

Below the query editor, the results are displayed in a table with two columns: 'handedness' and 'Player\_count'. The results are ordered by Player\_count in descending order.

handedness	Player_count
R	737
L	316
B	104

## Q20 .What is the average batting average for players

SELECT Avg(avg) As Average\_Batting\_Average from baseball\_data ;

The screenshot shows a database query editor interface. On the left, a sidebar lists the database 'chinook.db' and its tables. The 'baseball\_data' table is selected, and its columns are listed: name (TEXT), handedness (TEXT), height (INTEGER), weight (INTEGER), avg (REAL), and HR (INTEGER). The main editor area shows a SQL query: `SELECT Avg(avg) AS Average_Batting_Average FROM baseball_data ;`. The query is executed, and the results are displayed in a table with one row: `Average_Batting_Average` with the value `0.1867934312878133`.

Average_Batting_Average
0.1867934312878133

## Q21. What is the distribution of albums among different artists (Database – Chinook.db)

SELECT name,COUNT(albumid) AS Album\_Count from albums INNER JOIN artists on albums.ArtistId = artists.ArtistId GROUP by name order BY Album\_Count DESC LIMIT 10;

The screenshot shows a SQL query editor interface. On the left, the 'chinook.db' database schema is visible, listing tables like 'albums', 'artists', 'customers', etc. The main area displays a SQL query: `SELECT name,COUNT(albumid) AS Album_Count FROM albums INNER JOIN artists ON albums.ArtistId = artists.ArtistId GROUP BY name ORDER BY Album_Count DESC LIMIT 10;`. The results are shown in a table with two columns: 'Name' and 'Album\_Count'.

Name	Album_Count
Iron Maiden	21
Led Zeppelin	14
Deep Purple	11
U2	10
Metallica	10
Ozzy Osbourne	6
Pearl Jam	5
Various Artists	4
Van Halen	4