

SQL Project

Dataset: Fifa19 Players

Q-1 How many players are there in the datasets?

```
SELECT COUNT (*) as Total_Players from Players_Datasets;
```

Q-2 How many nationalities do these players belong to?

(16643)

```
SELECT count (*) AS Number_Of_Nationalities FROM Players_Datasets;  
(will show duplicates nationalities as well)
```

The screenshot shows the SQLiteOnline IDE interface. On the left, the sidebar lists databases: SQLite (0.1.3 beta), MariaDB, PostgreSQL, and MS SQL. The main area displays the following SQL code:

```
--SELECT * from Players_Datasets  
--Q1 How many players are there in the datasets?  
--SELECT COUNT (*) as Total_Players from Players_Datasets;  
--Q2 How many nationalities do these players belong to?  
SELECT COUNT (*) AS Number_of_Nationalities FROM Players_Datasets;
```

The results pane shows a single row with the value 16643 under the column "Number_of_Nationalities". The history pane on the right shows the execution of the same queries at different times: 13:35:23, 13:35:12, and 13:33:03.

```
SELECT COUNT(DISTINCT nationality) AS Number_of_Nationalities from Players_Datasets; (161)  
(will show unique count of nationalities)
```

The screenshot shows the SQLiteOnline IDE interface. The sidebar lists the same databases as before. The main area displays the following SQL code:

```
--SELECT * from Players_Datasets  
--Q1 How many players are there in the datasets?  
--SELECT COUNT (*) as Total_Players from Players_Datasets;  
--Q2 How many nationalities do these players belong to?  
--SELECT count (*) AS Number_of_Nationalities FROM Players_Datasets; (with Duplicate nationality)  
SELECT COUNT(DISTINCT nationality) AS Number_of_Nationalities FROM Players_Datasets; (WITH UNIQUE nationality)
```

The results pane shows a single row with the value 161 under the column "Number_of_Nationalities". The history pane on the right shows the execution of the queries at different times: 13:38:41, 13:35:23, and 13:35:12.

Q-3 Which nationality has the highest number of players, what are the top 3 nationalities by # of players?

SELECT COUNT (*) as frequency,nationality from Players_Datasets GROUP BY nationality order by frequency DESC limit 3;

The screenshot shows the SQL Online IDE interface. On the left, the SQLite database schema is visible, showing a single table 'Players_Datasets' with columns: ID (INTEGER), Name (TEXT), Age (INTEGER), Nationality (TEXT), Overall_rating (INTEGER), Potential_rating (INTEGER), Club (TEXT), Value (INTEGER), Wage (INTEGER), Preferred_Foot (TEXT), Jersey_No (INTEGER), Joined (TEXT), Height (TEXT), Weight (INTEGER), and Penalties (INTEGER). The main area displays the results of the query:

frequency	Nationality
1475	England
1151	Germany
974	Spain

The History panel on the right shows the execution of the query. It includes the SQL code, the output of the query, and a note about a syntax error: "Help: near \"order\": syntax error". The timestamp for the execution is 14:01:09.

Q-4 What is the total wage given to all players?

select sum(wage) as total_wage from Players_Datasets;

The screenshot shows the SQL Online IDE interface. On the left, the SQLite database schema is visible, showing the same table structure as before. The main area displays the results of the query:

total_wage
16007300

The History panel on the right shows the execution of the query. It includes the SQL code, the output of the query, and a note about a syntax error: "Help: near \"What\": syntax error". The timestamp for the execution is 14:14:59.

Q-5 What's the wage of average and standard deviation?

select Round(avg(wage),2) as average_wage, Round(STDEV(wage),2) as std_wage from

Players_Datasets;

The screenshot shows the SQLite tab in the SQL Online IDE. On the left, the table structure for 'Players_Datasets' is displayed with columns: ID (INTEGER), Name (TEXT), Age (INTEGER), Nationality (TEXT), Overall_rating (INTEGER), Potential_rating (INTEGER), Club (TEXT), Value (INTEGER), Wage (INTEGER), Preferred_Foot (TEXT), Jersey_No (INTEGER), Joined (TEXT), Height (TEXT), Weight (INTEGER), and Penalties (INTEGER). The right side shows the history of queries run against this table.

```
1 --SELECT * from Players_Datasets
2 --Q1 How many players are there in the datasets?
3 --SELECT COUNT (*) as Total_Players from Players_Datasets;
4 --Q2 How many nationalities do these players belong to?
5 --SELECT count (*) AS Number_Of_Nationalities FROM Players_Datasets; (with Duplicate nationality)
6 --SELECT COUNT(DISTINCT nationality) AS Number_of_Nationalities from Players_Datasets; (WITH UNIQUE nationality)
7 --Q3 Which nationality has the highest number of players, what are the top 3 nationalities by # of players?
8 --SELECT COUNT (*) as frequency,nationality from Players_Datasets GROUP BY nationality order by frequency DESC limit 3;
9 --Q4 What is the total wage given to all players?
10 --select sum(wage) as total_wage from Players_Datasets;
11 --Q5 What's the wage of average and standard deviation?
12 SELECT Round(avg(wage),2) AS average_wage, Round(STDEV(wage),2) AS std_wage FROM Players_Datasets;
```

Name	Club	Age	Wage
L. Messi	FC Barcelona	31	565000
L. Suárez	FC Barcelona	31	455000
L. Modrić	Real Madrid	32	420000
Cristiano Ronaldo	Juventus	33	405000
Sergio Ramos	Real Madrid	32	380000

Q-6 Give top 5 Players name with club and age to take maximum wage.

```
SELECT name,club,age,wage FROM Players_Datasets ORDER by wage DESC limit 5;
```

The screenshot shows the execution of the query 'SELECT name,club,age,wage FROM Players_Datasets ORDER by wage DESC limit 5;' in the SQLite tab. The results are displayed in a table below the query editor.

```
1 --Q6 Give top 5 Players name with club and age to take maximum wage.
2 SELECT name,club,age,wage FROM Players_Datasets ORDER by wage DESC LIMIT 5;
```

Name	Club	Age	Wage
L. Messi	FC Barcelona	31	565000
L. Suárez	FC Barcelona	31	455000
L. Modrić	Real Madrid	32	420000
Cristiano Ronaldo	Juventus	33	405000
Sergio Ramos	Real Madrid	32	380000

Q-7 Give bottom 10 Players name with club and age to take minimum wage.

```
SELECT name,club,age,wage FROM Players_Datasets ORDER by wage ASC limit 10;
```

```

SELECT name, club, age, wage FROM Players_Datasets WHERE overall_rating = (SELECT max(overall_rating) FROM Players_Datasets);

```

Name	Club	Age	Wage
L. Messi	Barcelona	36	94
Cristiano Ronaldo	Real Madrid	37	94

Q-8 which player having the best overall rating?

`SELECT name, club, overall_rating from Players_Datasets where overall_rating = (SELECT max(overall_rating) from Players_Datasets);`

```

SELECT name, club, overall_rating FROM Players_Datasets WHERE overall_rating = (SELECT max(overall_rating) FROM Players_Datasets);

```

club	Overall_rating
Barcelona	94
Real Madrid	94

Q-9 which player having the worst overall rating?

`SELECT name, nationality, club, overall_rating FROM Players_Datasets WHERE overall_rating=(SELECT MIN(overall_rating) FROM Players_Datasets);`

The screenshot shows the SQL Online IDE interface. On the left, the sidebar lists databases: SQLite (selected), MariaDB, and PostgreSQL. Under SQLite, the 'Players_Datasets' table is selected, showing columns: ID, Name, Nationality, Club, and Overall_rating. A query is being typed in the main editor:

```

SELECT name,nationality,club,overall_rating FROM Players_Datasets WHERE overall_rating=(SELECT MIN(overall_rating) FROM

```

The results table shows one row:

Name	Nationality	Club	Overall_rating
G. Nugent	England	Tranmere Rovers	46

On the right, the History panel shows previous queries:

- 15:02:56 --SELECT * from Players_Datasets --Q1 How many players are there in the --SELECT COUNT (*) ...
- 15:02:32 --SELECT * from Players_Datasets --Q1 How many players are there in the --SELECT COUNT (*) ...
- 14:59:06 --SELECT * from Players_Datasets --Q1 How many players are there in the --SELECT COUNT (*) ...

At the bottom, the system tray shows the date and time: 24-03-2024.

Q-10 which Top 3 club having Highest overall rating?

```
select sum(overall_rating) as total_rating, club from Players_Datasets group by club
order by total_rating desc limit 3;
```

The screenshot shows the SQL Online IDE interface. On the left, the sidebar lists databases: SQLite (selected), MariaDB, and PostgreSQL. Under SQLite, the 'Players_Datasets' table is selected, showing columns: ID, Name, Nationality, Club, and Overall_rating. A query is being typed in the main editor:

```

SELECT sum(overall_rating) AS total_rating, club FROM Players_Datasets GROUP BY club ORDER BY total_rating
DESC LIMIT 3;

```

The results table shows three rows:

total_rating	Club
2582	Real Madrid
2575	FC Barcelona
2549	Manchester United

On the right, the History panel shows previous queries:

- 15:08:08 --SELECT * from Players_Datasets --Q1 How many players are there in the --SELECT COUNT (*) ...
- 15:07:47 --SELECT * from Players_Datasets --Q1 How many players are there in the --SELECT COUNT (*) ...

At the bottom, the system tray shows the date and time: 24-03-2024.

Q-11 Find top 7 best players in penalties?

```
SELECT name,club,nationality,penalties from Players_Datasets ORDER by penalties desc
limit 7;
```

SQL Online IDE

sqliteonline.co

Suzuki Motorcycle L... (1) Ganesh Gayatri... This Mantra Helped... Power BI Dashboard... Power BI Dashboard... Content store (2) Road map to be... Power BI Masterclas... Learn from top.com... All Bookmarks

File Owner DB Run Export Import Client

SQLite

0.1.3 beta

Table

Players_Datasets

Column

- ID INTEGER
- Name TEXT
- Age INTEGER
- Nationality TEXT
- Overall_rating INTEGER
- Potential_rating INTEGER
- Club TEXT
- Value INTEGER
- Wage INTEGER
- Preferred_Foot TEXT
- Jersey_No INTEGER
- Joined TEXT
- Height TEXT
- Weight INTEGER
- Penalties INTEGER

MariaDB

PostgreSQL

INR/USD -0.47%

History

Syntax | History

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the dataset?
--SELECT COUNT(*)
```

15:22:20

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the dataset?
--SELECT COUNT(*)
```

15:19:54

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the dataset?
--SELECT COUNT(*)
```

Help near "O11" syntax error

15:22:20 24-03-2024

Name	Club	Nationality	Penalties
M. Balotelli	OGC Nice	Italy	92
Fabinho	Liverpool	Brazil	91
H. Kane	Tottenham Hotspur	England	90
M. Kruse	SV Werder Bremen	Germany	90
D. Perotti	Roma	Argentina	90
R. Boudebouz	Real Betis	Algeria	90
L. Baines	Everton	England	90

Q-12 What is the distribution of players whose preferred foot is left vs right?

SELECT COUNT(*) as Players_Number,preferred_foot from Players_Datasets group by preferred_foot ORDER BY Players_Number desc;

SQL Online IDE

sqliteonline.co

Suzuki Motorcycle L... (1) Ganesh Gayatri... This Mantra Helped... Power BI Dashboard... Power BI Dashboard... Content store (2) Road map to be... Power BI Masterclas... Learn from top.com... All Bookmarks

File Owner DB Run Export Import Client

SQLite

0.1.3 beta

Table

Players_Datasets

Column

- ID INTEGER
- Name TEXT
- Age INTEGER
- Nationality TEXT
- Overall_rating INTEGER
- Potential_rating INTEGER
- Club TEXT
- Value INTEGER
- Wage INTEGER
- Preferred_Foot TEXT
- Jersey_No INTEGER
- Joined TEXT
- Height TEXT
- Weight INTEGER
- Penalties INTEGER

MariaDB

PostgreSQL

33°C Sunny

History

Syntax | History

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the dataset?
--SELECT COUNT(*)
```

15:28:27

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the dataset?
--SELECT COUNT(*)
```

15:27:56

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the dataset?
--SELECT COUNT(*)
```

15:26:25

15:28 24-03-2024

Players_Number	Preferred_Foot
12823	Right
3820	Left

Q13 Which jersey number is the luckiest? (also with name and nationality)

SELECT sum(wage) as Total_Wage, name, nationality, jersey_no from Players_Datasets group by jersey_no ORDER by Total_Wage DESC limit 1;

SQL Online IDE

sqliteonline.co

Suzuki Motorcycle L... (1) Ganesh Gayatri... This Mantra Helped... Power BI Dashboard... Power BI Dashboard... Content store (2) Road map to be... Power BI Masterclas... Learn from top.com... All Bookmarks

File Owner DB Run Export Import Client

SQLite

0.1.3 beta

Table

Players_Datasets

Column

- ID INTEGER
- Name TEXT
- Age INTEGER
- Nationality TEXT
- Overall_rating INTEGER
- Potential_rating INTEGER
- Club TEXT
- Value INTEGER
- Wage INTEGER
- Preferred_Foot TEXT
- Jersey_No INTEGER
- Joined TEXT
- Height TEXT
- Weight INTEGER
- Penalties INTEGER

MariaDB

PostgreSQL

History

Syntax | History

```
--Q10 which Top 3 club having Highest overall rating ?
SELECT name,nationality,club,overall_rating FROM Players_Datasets WHERE overall_rating=(SELECT MIN(overall_rating) FF
21 --Q10 which Top 3 club having Highest overall rating ?
22 /* select sum(overall_rating) as total_rating, club from Players_Datasets group by club order by total_rating
23 desc limit 3; */
24 --Q11 Find top 7 best players in penalties?
25 --SELECT name,club,nationality,penalties from Players_Datasets ORDER by penalties desc limit 7;
26 --Q12 What is the distribution of players whose preferred foot is left vs right?
27 /* SELECT COUNT(*) as Players_Number,preferred_foot from Players_Datasets group by preferred_foot
28 ORDER BY Players_Number desc; */
29 --Q13 Which jersey number is the luckiest? (also with name nationality)
30 SELECT sum(wage) AS Total_Wage,name,nationality,jersey_no FROM Players_Datasets GROUP BY jersey_no
31 ORDER BY Total_Wage DESC LIMIT 1;
```

Total_Wage	Name	Nationality	Jersey_No
9975000	L. Messi	Argentina	10

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the c
--SELECT COUNT (*)
```

15:45:20

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the c
--SELECT COUNT (*)
```

15:44:53

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the c
--SELECT COUNT (*)
```

15:44:29

33°C Sunny

Search

ENG IN 1547 24-03-2024

Q-14 How many players have joined their respective clubs date wise?

```
select count(*) as Players,club,joined from Players_Datasets group by joined order by Players desc;
```

SQL Online IDE

sqliteonline.co

Suzuki Motorcycle L... (1) Ganesh Gayatri... This Mantra Helped... Power BI Dashboard... Power BI Dashboard... Content store (2) Road map to be... Power BI Masterclas... Learn from top.com... All Bookmarks

File Owner DB Run Export Import Client

SQLite

0.1.3 beta

Table

Players_Datasets

Column

- ID INTEGER
- Name TEXT
- Age INTEGER
- Nationality TEXT
- Overall_rating INTEGER
- Potential_rating INTEGER
- Club TEXT
- Value INTEGER
- Wage INTEGER
- Preferred_Foot TEXT
- Jersey_No INTEGER
- Joined TEXT
- Height TEXT
- Weight INTEGER
- Penalties INTEGER

MariaDB

PostgreSQL

History

Syntax | History

```
--Q11 Find top 7 best players in penalties?
25 --SELECT name,club,nationality,penalties from Players_Datasets ORDER by penalties desc limit 7;
26 --Q12 What is the distribution of players whose preferred foot is left vs right?
27 /* SELECT COUNT(*) as Players_Number,preferred_foot from Players_Datasets group by preferred_foot
28 ORDER BY Players_Number desc; */
29 --Q13 Which jersey number is the luckiest? (also with name and nationality)
30 /* SELECT sum(wage) as Total_Wage,name,nationality,jersey_no from Players_Datasets group by jersey_no
31 ORDER by Total_Wage DESC limit 1; */
32 --Q14 How many players have joined their respective clubs date wise?
33 SELECT COUNT(*) AS Players,club,joined FROM Players_Datasets GROUP BY joined ORDER BY Players DESC;
```

Players	Club	Joined
1538	Paris Saint-Germain	01-07-2018
1133	Liverpool	01-07-2017
635	Liverpool	01-01-2018
614	FC Bayern München	01-07-2016
368	Juventus	01-07-2015
231	Shanghai SIPG FC	01-01-2017
226	FC Bayern München	01-07-2014

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the c
--SELECT COUNT (*)
```

16:02:58

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the c
--SELECT COUNT (*)
```

16:02:09

SQLite

```
--SELECT * from Players_Datasets
--Q1 How many players are there in the c
--SELECT COUNT (*)
```

16:01:00

INR/USD -0.47%

Search

ENG IN 16:03 24-03-2024

Dataset: Digital media store's artists, albums, media tracks, invoices, and Customers

Q-1 Show full name, customer ID, and country for all customers who are not in the USA.

```
SELECT first_name,last_name,customer_id,country FROM customer WHERE country <> 'USA'  
ORDER BY country;
```

The screenshot shows the SQLite Online IDE interface. On the left, there is a sidebar with a tree view of databases (chinook.db), tables (album, artist, customer, employee, genre, invoice, invoice_line, media_type, playlist, playlist_track, track), and other database types (SQLite, MariaDB, PostgreSQL, MS SQL). The main area has tabs for 'SQL Report' and 'History'. The 'SQL Report' tab contains the SQL query from Q-1. The 'History' tab shows two previous queries: one for all customers and another for a specific customer. The results pane displays a table with columns: first_name, last_name, customer_id, and country. The data shows customers from Argentina, Australia, Austria, Belgium, Brazil, and Brazil.

first_name	last_name	customer_id	country
Diego	Gutiérrez	56	Argentina
Mark	Taylor	55	Australia
Astrid	Gruber	7	Austria
Daan	Peeters	8	Belgium
Luis	Gonçalves	1	Brazil
Eduardo	Martins	10	Brazil
Alexandre	Rocha	11	Brazil

Q-2 Show Only customer From India.

```
SELECT first_name,last_name,customer_id,country from customer WHERE country = "India";
```

The screenshot shows the SQLite Online IDE interface. The sidebar and tabs are identical to the previous screenshot. The 'SQL Report' tab now contains the SQL query from Q-2. The 'History' tab shows three previous queries: the first two are the same as in the previous screenshot, and the third is the new query for India. The results pane displays a table with columns: first_name, last_name, customer_id, and country. The data shows two customers from India.

first_name	last_name	customer_id	country
Manoj	Pareek	58	India
Puja	Srivastava	59	India

Q-3 Find the invoices of customers who are from India and display the customer's full name, invoice ID,date of invoice, and billing country.

```
SELECT c.first_name,c.last_name,i.invoice_id,i.invoice_date,i.billing_country FROM customer AS c LEFT join invoice AS i ON c.customer_id = i.customer_id WHERE i.billing_country = "India";
```

The screenshot shows the SQLite Online IDE interface. On the left, there's a sidebar with database connections to chinook.db, SQLite, MariaDB, PostgreSQL, and MS SQL. The main area has tabs for 'Table' and 'SQL Report'. The 'SQL Report' tab contains the following SQL code:

```
1 --Q-3 Find the invoices of customers who are from India and display the customer's full name, invoice ID, date of invoice, and billing country
2
3 SELECT c.first_name,c.last_name,i.invoice_id,i.invoice_date,i.billing_country
4
5 FROM customer AS c LEFT JOIN invoice AS i ON c.customer_id = i.customer_id
6 WHERE i.billing_country = "India";
```

The results table on the right shows the following data:

	first_name	last_name	invoice_id	invoice_date	billing_country
1	Manoj	Pareek	33	2017-02-21 00:00:00	India
2	Puja	Srivastava	61	2017-04-24 00:00:00	India

Q-4) Show the employees who are sales agents.

```
SELECT first_name,last_name,employee_id,title FROM employee WHERE title = 'Sales Support Agent' ORDER BY last_name;
```

The screenshot shows the SQLite Online IDE interface. On the left, there's a sidebar with database connections to chinook.db, SQLite, MariaDB, PostgreSQL, and MS SQL. The main area has tabs for 'Table' and 'SQL Report'. The 'SQL Report' tab contains the following SQL code:

```
1 --Q-4 Show the employees who are sales agents
2
3 SELECT first_name,last_name,employee_id,title FROM employee
4 WHERE title = 'Sales Support Agent' ORDER BY last_name;
```

The results table on the right shows the following data:

	first_name	last_name	employee_id	title
1	Steve	Johnson	5	Sales Support Agent
2	Margaret	Park	4	Sales Support Agent
3	Jane	Peacock	3	Sales Support Agent

Q-5 Find a unique list of billing countries from the invoices table

```
SELECT count(DISTINCT billing_country) FROM invoice ORDER BY billing_country; (24)
SELECT DISTINCT billing_country FROM invoice ORDER BY billing_country;
```

```

11 FROM customer AS c LEFT join invoice AS i ON c.customer_id = i.customer_id
12 WHERE i.billing_country = "India";
13 /*
14 --Q-4 Show the employees who are sales agents
15 --SELECT first_name,last_name,employee_id,title from employee;
16 */
17 SELECT first_name,last_name,employee_id,title FROM employee
18 WHERE title = 'Sales Support Agent' ORDER BY last_name;
19 /*
20 --Q-5 Find a unique list of billing countries from the invoices table
21 --SELECT count (DISTINCT billing_country) FROM invoice ORDER BY billing_country;
22 SELECT DISTINCT billing_country FROM invoice ORDER BY billing_country;

```

History

```

--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and ...
...
```

16:14:56

```

--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and ...
...
```

16:11:49

```

--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and ...
...
```

16:11:49

Help: unrecognized token: "ODISTINCT"

16:11:49

Q-6 Show the invoice IDs associated with each sales agent, including the sales agent's full name.

```

SELECT e.first_name,e.last_name,i.invoice_id from customer AS c join employee as e on
c.support_rep_id = e.employee_id join invoice as i on c.customer_id = i.customer_id
ORDER BY e.first_name;

```

```

--Q-4 Show the employees who are sales agents
15 --SELECT first_name,last_name,employee_id,title from employee;
16 /*
17 SELECT first_name,last_name,employee_id,title FROM employee
18 WHERE title = 'Sales Support Agent' ORDER BY last_name;
19 /*
20 --Q-5 Find a unique list of billing countries from the invoices table.
21 --SELECT count (DISTINCT billing_country) FROM invoice ORDER BY billing_country;
22 --SELECT DISTINCT billing_country FROM invoice ORDER BY billing_country;
23 --Q-6 Show the invoice IDs associated with each sales agent, including the sales agent's full name.
24 SELECT e.first_name,e.last_name,i.invoice_id FROM customer AS c JOIN employee AS e ON c.support_rep_id = e.employee_id
25 JOIN invoice AS i ON c.customer_id = i.customer_id ORDER BY e.first_name;

```

History

```

--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and ...
...
```

16:34:11

```

--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and ...
...
```

16:33:33

```

--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and ...
...
```

16:33:06

Q-7 How many invoices were there in 2019?

```

SELECT COUNT (invoice_id) AS '2019_Sales' FROM invoice where invoice_date like '%2019%';

```

```

19 /*
20 --Q-5 Find a unique list of billing countries from the invoices table.
21 --SELECT count (DISTINCT billing_country) FROM invoice ORDER BY billing_country;
22 --SELECT DISTINCT billing_country FROM invoice ORDER BY billing_country;
23 --Q-6 Show the invoice IDs associated with each sales agent, including the sales agent's full name.
24 /*
25 SELECT e.first_name,e.last_name,i.invoice_id FROM customer AS c join employee as e on c.support_rep_id = e.employee_id
26 join invoice as i on c.customer_id = i.customer_id ORDER BY e.first_name;
27 */
28 --Q-7 How many invoices were there in 2009?
29 SELECT COUNT (invoice_id) AS '2019_Sales' FROM invoice WHERE invoice_date LIKE '%2019%';
30

```

2019_Sales
159

History

Syntax | History

SQL Report

```
--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and...
```

16:54:33

SQL Report

```
--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and...
```

16:52:27

SQL Report

```
--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and...
```

16:51:30

SQL Report

```
--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and...
```

16:55 23-03-2024

Q-8 What are the total sales for 2019?

```
SELECT Round(sum(total),2) AS total_Sales FROM invoice where invoice_date like '%2019%';
```

```

25 SELECT e.first_name,e.last_name,i.invoice_id FROM customer AS c join employee as e on c.support_rep_id = e.employee_id
26 join invoice as i on c.customer_id = i.customer_id ORDER BY e.first_name;
27 /*
28 --Q-7 How many invoices were there in 2019?
29 --SELECT COUNT (invoice_id) AS '2019_Sales' FROM invoice where invoice_date like '%2019%';
30 --Q-8 What are the total sales for 2019?
31 SELECT Round(sum(total),2) AS total_Sales FROM invoice WHERE invoice_date LIKE '%2019%';
32
33
34
35
36

```

total_Sales
1221.66

History

Syntax | History

SQL Report

```
--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and...
```

17:06:08

SQL Report

```
--SELECT * FROM customer;
--Q-1 Show full name, customer ID, and...
```

17:06 23-03-2024

Help: no such column: employee_id
Table [customer] column: customer_id,
first_name, last_name, company, address, city,
state, country, postal_code, phone, fax, email,
support_rep_id
Table [employee] column: employee_id,
last_name, first_name, title, reports_to,
birthdate, hire_date, address, city, state,
country, postal_code, phone, tax, email
Table [invoice] column: invoice_id,
customer_id, invoice_date, billing_address,
billing_city, billing_state, billing_country,

Q-9 Show the total sales made by each sales agent.

```

SELECT e.first_name,e.last_name,round(sum(i.total),2) AS total_Sales from customer as c
JOIN employee as e on c.support_rep_id = e.employee_id JOIN invoice as i on c.customer_id =
i.customer_id group by c.support_rep_id ORDER BY total_Sales DESC;

```

The screenshot shows the SQL Online IDE interface. The left sidebar lists databases (chinook.db, 0.1.3 beta) and tables (album, artist, customer, employee, genre, invoice, invoice_line, media_type, playlist, playlist_track, track). The main area displays a SQL report with the following code:

```

31 --SELECT Round(sum(total),2) AS total_Sales FROM invoice where invoice_date like '%2019%';
32 --Q-9 Show the total sales made by each sales agent
33 /*
34 SELECT e.first_name,e.last_name,round(sum(i.total),2) AS total_Sales from customer as c JOIN employee as e
35 ON c.support_rep_id = e.employee_id JOIN invoice AS i ON c.customer_id = i.customer_id
36 GROUP BY c.support_rep_id ORDER BY total_Sales DESC;
37 */
38 /*
39 SELECT e.first_name,e.last_name,round(sum(i.total),2) AS total_Sales from customer as c JOIN employee as e
40 ON c.support_rep_id = e.employee_id JOIN invoice AS i ON c.customer_id = i.customer_id
41 WHERE i.invoice_date like '%2019%' GROUP BY c.support_rep_id ORDER BY total_Sales DESC;
42 */
43 /*

```

The results table shows the following data:

	first_name	last_name	total_Sales
1	Jane	Peacock	1731.51
2	Margaret	Park	1584
3	Steve	Johnson	1393.92

The right panel shows the History of previous queries, including:

- SQL Report (18:31:03): SELECT * FROM customer; --Q-1 Show full name, customer ID, and ...
- SQL Report (18:29:41): Help: no such table: employee_id List table's in database: album, artist, customer, employee, genre, invoice, invoice_line, media_type, playlist, playlist_track, track
- SQL Report (18:29:41): --SELECT * FROM customer; --Q-1 Show full name, customer ID, and ...

Q- 10 Which Two sales agent made the most dollars in sales in 2019?

```

SELECT e.first_name,e.last_name,round(sum(i.total),2) as Dollars_made from customer as c
JOIN employee as e on c.support_rep_id = e.employee_id JOIN invoice as i on c.customer_id =
i.customer_id WHERE i.invoice_date like '%2019%' GROUP BY C.support_rep_id
ORDER BY Dollars_made DESC LIMIT 2;

```

The screenshot shows the SQL Online IDE interface. The left sidebar lists databases (chinook.db, 0.1.3 beta) and tables (album, artist, customer, employee, genre, invoice, invoice_line, media_type, playlist, playlist_track, track). The main area displays a SQL report with the following code:

```

31 --SELECT Round(sum(total),2) AS total_Sales FROM invoice where invoice_date like '%2019%';
32 --Q-9 Show the total sales made by each sales agent
33 /*
34 SELECT e.first_name,e.last_name,round(sum(i.total),2) AS total_Sales from customer as c JOIN employee as e
35 ON c.support_rep_id = e.employee_id JOIN invoice AS i ON c.customer_id = i.customer_id
36 group by c.support_rep_id ORDER BY total_Sales DESC;
37 */
38 --Q-10 Which Two sales agent made the most dollars in sales in 2019?
39 SELECT e.first_name,e.last_name,round(sum(i.total),2) AS Dollars_made FROM customer AS c JOIN employee AS e
40 ON c.support_rep_id = e.employee_id JOIN invoice AS i ON c.customer_id = i.customer_id WHERE i.invoice_date
41 LIKE '%2019%' GROUP BY C.support_rep_id ORDER BY Dollars_made DESC LIMIT 2;
42 */
43 /*

```

The results table shows the following data:

	first_name	last_name	Dollars_made
1	Steve	Johnson	437.58
2	Margaret	Park	400.95

The right panel shows the History of previous queries, including:

- SQL Report (18:47:10): --SELECT * FROM customer; --Q-1 Show full name, customer ID, and ...
- SQL Report (18:46:50): --SELECT * FROM customer; --Q-1 Show full name, customer ID, and ...
- SQL Report (18:45:46): --SELECT * FROM customer; --Q-1 Show full name, customer ID, and ...

Dataset: Netflix-Data

Q-1 What were the top 5 movies according to IMDB score?

```
SELECT title,type,imdb_score FROM titles WHERE imdb_score >= 8.0 AND type = 'MOVIE'  
ORDER BY imdb_score DESC LIMIT 5;
```

The screenshot shows the SQL Online IDE interface. On the left, there's a sidebar with database connections (SQLite 0.1.3 beta, MariaDB, PostgreSQL) and a table named 'titles' with columns like id, title, type, etc. The main area shows the query:

```
1 --SELECT * from titles;  
2 --Q-1 What were the top 5 movies according to IMDB score?  
3 SELECT title,type,imdb_score FROM titles WHERE imdb_score >= 8.0 AND type = 'MOVIE'  
4 ORDER BY imdb_score DESC LIMIT 5;
```

Below the query, the results are displayed in a table:

title	type	imdb_score
Chhota Bheem & Krishna vs Zimbara	MOVIE	9.1
Major	MOVIE	9.1
C/o Kancharapalem	MOVIE	8.9
David Attenborough: A Life on Our Planet	MOVIE	8.9
Forrest Gump	MOVIE	8.8

The right side of the interface shows a 'History' panel with previous queries and their execution times.

Q-2) What were the top 5 shows according to IMDB score?

```
SELECT title,type,imdb_score FROM titles WHERE imdb_score >= 8.0 AND type = 'SHOW'  
ORDER BY imdb_score DESC LIMIT 5;
```

The screenshot shows the SQL Online IDE interface. On the left, there's a sidebar with database connections (SQLite 0.1.3 beta, MariaDB, PostgreSQL) and a table named 'titles' with columns like id, title, type, etc. The main area shows the query:

```
1 --SELECT * from titles;  
2 --Q-1 What were the top 5 movies according to IMDB score?  
3 /* SELECT title,type,imdb_score FROM titles WHERE imdb_score >= 8.0 AND type = 'MOVIE'  
4 ORDER BY imdb_score DESC LIMIT 5; */  
5 --Q-2) What were the top 5 shows according to IMDB score?  
6 SELECT title,type,imdb_score FROM titles WHERE imdb_score >= 8.0 AND type = 'SHOW'  
7 ORDER BY imdb_score DESC LIMIT 5;
```

Below the query, the results are displayed in a table:

title	type	imdb_score
#ABtalks	SHOW	9.6
Breaking Bad	SHOW	9.5
Khawatir	SHOW	9.5
Avatar: The Last Airbender	SHOW	9.3
Our Planet	SHOW	9.3

The right side of the interface shows a 'History' panel with previous queries and their execution times. There is a syntax error highlighted in the history panel.

Q-3 What were the bottom 5 movies according to IMDB score?

```
SELECT title,type,imdb_score FROM titles WHERE type = 'MOVIE' ORDER BY imdb_score ASC limit 5;
```

The screenshot shows the SQL OnLine IDE interface. The left sidebar lists databases: SQLite (0.1.3 beta), MariaDB, PostgreSQL, and MS SQL. The main area is titled "SQLite" and contains the following code:

```
1 --SELECT * from titles;
2 --Q-1 What were the top 5 movies according to IMDB score?
3 /* SELECT title,type,imdb_score FROM titles WHERE imdb_score >= 8.0 AND type = 'MOVIE'
4 ORDER BY imdb_score DESC LIMIT 5; */
5 --Q-2 What were the top 5 shows according to IMDB score?
6 /* SELECT title,type,imdb_score FROM titles WHERE imdb_score >= 8.0 AND type = 'SHOW'
7 ORDER BY imdb_score DESC LIMIT 5; */
8 --Q-3 What were the bottom 5 movies according to IMDB score?
9 SELECT title,type,imdb_score FROM titles WHERE type = 'MOVIE' ORDER BY imdb_score ASC LIMIT 5;
10
```

Below the code, the results table shows the following data:

title	type	imdb_score
Aerials	MOVIE	1.5
Me Against You: Mr. S's Vendetta	MOVIE	1.6
Himmatwala	MOVIE	1.7
Kyaa Kool Hain Hum 3	MOVIE	1.9
FRED 3: Camp Fred	MOVIE	2

The right panel, titled "History", shows the command history with timestamp 23:04:00.

Q-4 What were the bottom 10 shows according to IMDB score?

```
SELECT title,type,imdb_score from titles WHERE type = 'SHOW' order BY imdb_score ASC limit 5;
```

The screenshot shows the SQL OnLine IDE interface. The left sidebar lists databases: SQLite (0.1.3 beta), MariaDB, PostgreSQL, and MS SQL. The main area is titled "SQLite" and contains the following code:

```
1 --SELECT * from titles;
2 --Q-1 What were the top 5 movies according to IMDB score?
3 /* SELECT title,type,imdb_score FROM titles WHERE imdb_score >= 8.0 AND type = 'MOVIE'
4 ORDER BY imdb_score DESC LIMIT 5; */
5 --Q-2 What were the top 5 shows according to IMDB score?
6 /* SELECT title,type,imdb_score FROM titles WHERE imdb_score >= 8.0 AND type = 'SHOW'
7 ORDER BY imdb_score DESC LIMIT 5; */
8 --Q-3 What were the bottom 5 movies according to IMDB score?
9 --SELECT title,type,imdb_score FROM titles WHERE type = 'MOVIE' ORDER BY imdb_score ASC limit 5;
10 --Q-4 What were the bottom 10 shows according to IMDB score?
11 SELECT title,type,imdb_score FROM titles WHERE type = 'SHOW' ORDER BY imdb_score ASC LIMIT 5;
12
```

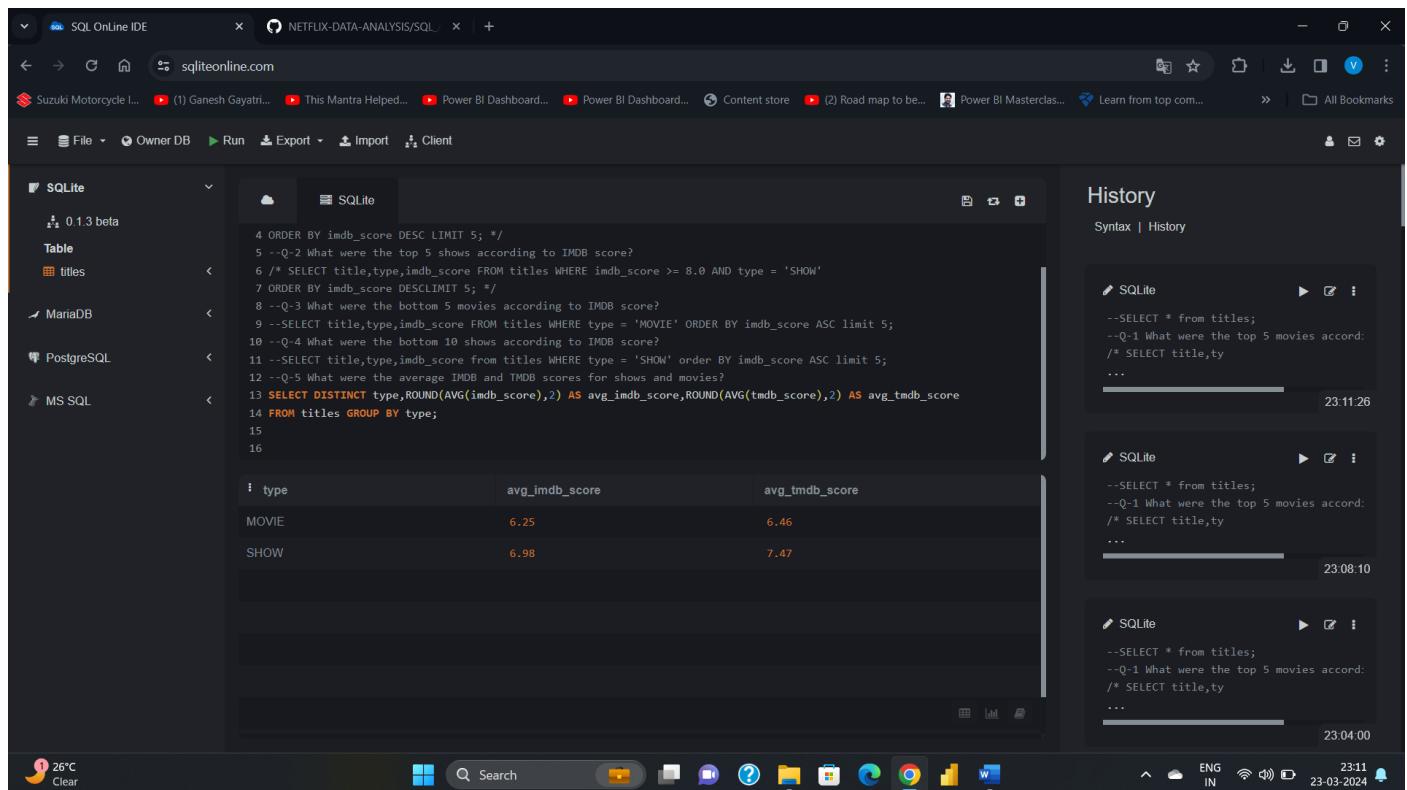
Below the code, the results table shows the following data:

title	type	imdb_score
Thomas & Friends: All Engines Go!	SHOW	2
He's Expecting	SHOW	2
Hype House	SHOW	2.1
A House of Blocks	SHOW	2.3
Until Dawn	SHOW	2.4

The right panel, titled "History", shows the command history with timestamp 23:08:10.

Q-5 What were the average IMDB and TMDB scores for shows and movies?

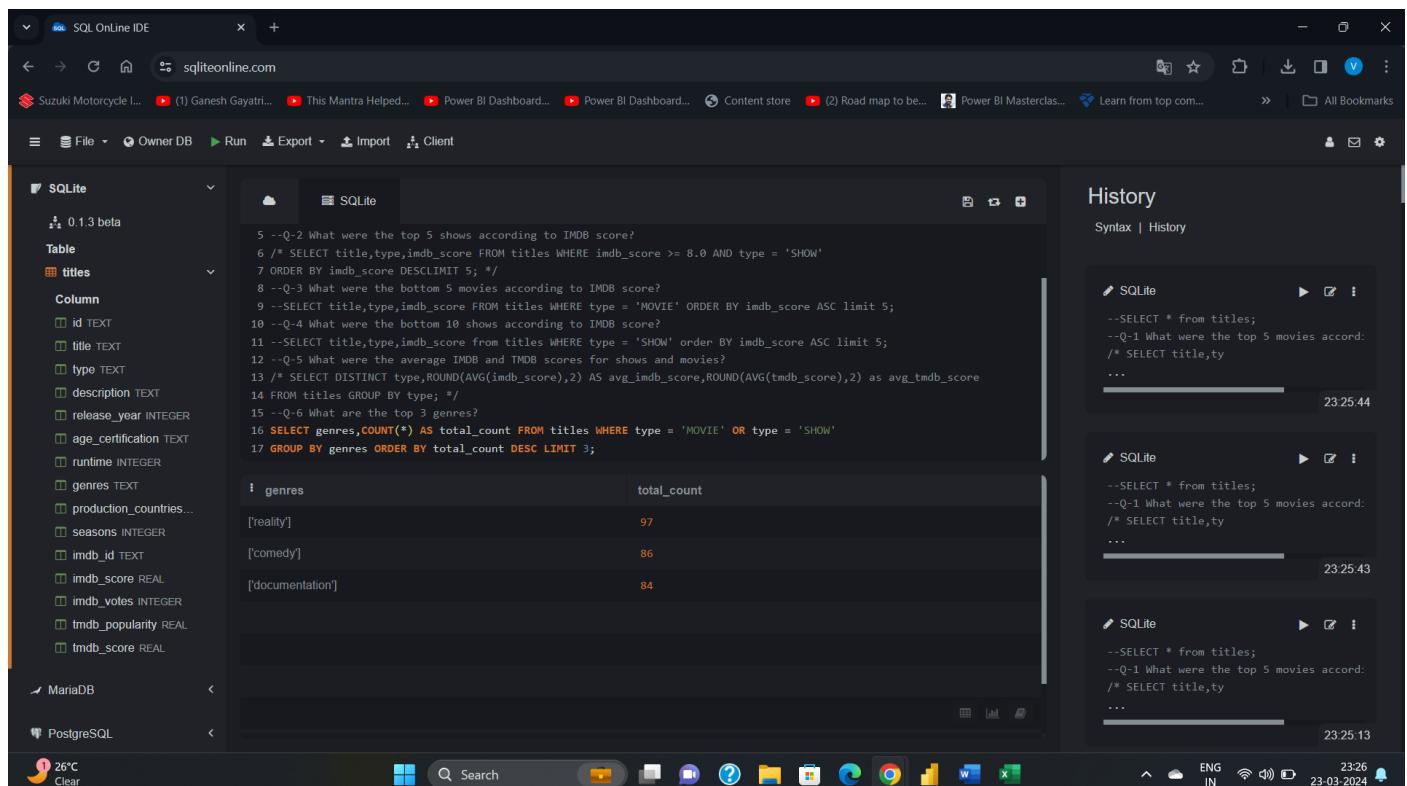
```
SELECT DISTINCT type,ROUND(AVG(imdb_score),2) AS avg_imdb_score ,ROUND(AVG(tmdb_score),2) as avg_tmdb_score FROM titles GROUP BY type;
```



type	avg_imdb_score	avg_tmdb_score
MOVIE	6.25	6.46
SHOW	6.98	7.47

Q-6 What are the top 3 genres?

```
SELECT genres,COUNT(*) AS total_count FROM titles WHERE type = 'MOVIE' OR type = 'SHOW' GROUP BY genres ORDER BY total_count DESC Limit 3;
```



genres	total_count
[reality]	97
[comedy]	86
[documentation]	84