Dashboard Properties SQL Statistics Dependencies Dependents Processes Music_Database/postgres@PostgreSQL 15*

Music_Database/postgres@PostgreSQL 15

No limit

Query Query History

```sql
--1. Who is the senior most employee based on job title?

SELECT * FROM EMPLOYEE
ORDER BY LEVELS DESC
LIMIT 1;
```

Data Output Messages Notifications

	employee_id [PK] character	last_name character (50)	first_name character (50)	title character varying (50)	reports_to character varying (30)	levels character varying (10)	birthdate timestamp without time zone	hire_date timestamp without time zone	address character varying (120)	city character varying (50)	state character varying (50)	country character varying	postal_c charact
1	9	Madan ...	Mohan ...	Senior General Manager	[null]	L7	1961-01-26 00:00:00	2016-01-14 00:00:00	1008 Vrinda Ave MT	Edmonton	AB	Canada	T5K 2N

```sql
AS CUST_SALES FROM CUSTOMER
```

Total rows: 1 of 1 Query complete 00:00:00.219 Ln 3, Col 1

Dashboard Properties SQL Statistics Dependencies Dependents Processes Music_Database/postgres@PostgreSQL 15*

Music_Database/postgres@PostgreSQL 15

No limit

Query Query History

```sql
--1. Who is the senior most employee based on job title?

SELECT * FROM EMPLOYEE
ORDER BY LEVELS DESC
LIMIT 1;

--2. Which countries have the most Invoices?

SELECT COUNT (*) AS COUNT_COUNTRY, BILLING_COUNTRY FROM INVOICE
GROUP BY BILLING_COUNTRY
ORDER BY COUNT_COUNTRY DESC ;
```

Data Output Messages Notifications

	count_country bigint	billing_country character varying (30)
1	131	USA
2	76	Canada
3	61	Brazil
4	50	France
5	41	Germany
6	30	Czech Republic
7	29	Portugal
8	28	United Kingdom
9	21	India
10	13	Chile
11	13	Ireland
12	11	Spain
13	11	Finland
14	10	Australia
15	10	Netherlands
16	10	Sweden
17	10	Poland
18	10	Hungary

```sql
ORDER BY CUST_SALES DESC
```

Total rows: 24 of 24 Query complete 00:00:00.093 Ln 9, Col 1

Dashboard Properties SQL Statistics Dependencies Dependents Processes Music_Database/postgres@PostgreSQL 15*

Music_Database/postgres@PostgreSQL 15

No limit

Query Query History

```sql
--1. Who is the senior most employee based on job title?

SELECT * FROM EMPLOYEE
ORDER BY LEVELS DESC
LIMIT 1;

--2. Which countries have the most Invoices?

SELECT COUNT (*) AS COUNT_COUNTRY, BILLING_COUNTRY FROM INVOICE
GROUP BY BILLING_COUNTRY
ORDER BY COUNT_COUNTRY DESC ;

--3. What are top 3 values of total invoice?

SELECT TOTAL FROM INVOICE
ORDER BY TOTAL DESC
LIMIT 3;
```

Data Output Messages Notifications

	total double precision
1	23.759999999999998
2	19.8
3	19.8

```sql
ORDER BY EMAIL ASC;
```

Total rows: 3 of 3 Query complete 00:00:00.051 Ln 15, Col 1

Music_Database/postgres@PostgreSQL 15

No limit

Query Query History

```
SELECT COUNT (*) AS COUNT_COUNTRY, BILLING_COUNTRY FROM INVOICE
GROUP BY BILLING_COUNTRY
ORDER BY COUNT_COUNTRY DESC ;

--3. What are top 3 values of total invoice?

SELECT TOTAL FROM INVOICE
ORDER BY TOTAL DESC
LIMIT 3;

--4. Which city has the best customers? We would like to throw a promotional Music Festival in the city we made the most money.
-- (city that has the highest sum of invoice totals)

SELECT SUM(TOTAL) AS SUM_SALES, BILLING_CITY FROM INVOICE
GROUP BY BILLING_CITY
ORDER BY SUM_SALES DESC
LIMIT 1;
```

Data Output Messages Notifications

	sum_sales double precision	billing_city character varying (30)
1	273.24000000000007	Prague

```
54 WHERE TRACK_ID IN(
```

Music_Database/postgres@PostgreSQL 15

No limit

Query Query History

```
LIMIT 3;

--4. Which city has the best customers? We would like to throw a promotional Music Festival in the city we made the most money.
-- (city that has the highest sum of invoice totals)

SELECT SUM(TOTAL) AS SUM_SALES, BILLING_CITY FROM INVOICE
GROUP BY BILLING_CITY
ORDER BY SUM_SALES DESC
LIMIT 1;

--5. Who is the best customer? The customer who has spent the most money will be declared the best customer.

SELECT CUSTOMER.CUSTOMER_ID, CUSTOMER.FIRST_NAME, CUSTOMER.LAST_NAME, SUM(INVOICE.TOTAL)
AS CUST_SALES FROM CUSTOMER
JOIN INVOICE ON CUSTOMER.CUSTOMER_ID = INVOICE.CUSTOMER_ID
GROUP BY CUSTOMER.CUSTOMER_ID
ORDER BY CUST_SALES DESC
LIMIT 1;
```

Data Output Messages Notifications

	customer_id [PK] integer	first_name character (50)	last_name character (50)	cust_sales double precision
1	5	R ...	Madhav	144.54000000000002

```
--8. Need all the track names that have a song length longer than the average song length.
```

Music_Database/postgres@PostgreSQL 15

No limit

Query Query History

```
30 AS CUST_SALES FROM CUSTOMER
31 JOIN INVOICE ON CUSTOMER.CUSTOMER_ID = INVOICE.CUSTOMER_ID
32 GROUP BY CUSTOMER.CUSTOMER_ID
33 ORDER BY CUST_SALES DESC
34 LIMIT 1;
35
36 --6. Need details of all Rock Music listeners (email, first name, last name, & Genre).
37
38 SELECT DISTINCT EMAIL, FIRST_NAME, LAST_NAME FROM CUSTOMER
39 JOIN INVOICE ON CUSTOMER.CUSTOMER_ID =  INVOICE.CUSTOMER_ID
40 JOIN INVOICE_LINE ON INVOICE.INVOICE_ID =  INVOICE_LINE.INVOICE_ID
41 WHERE TRACK_ID IN(
42     SELECT TRACK_ID FROM TRACK
43     JOIN GENRE ON TRACK.GENRE_ID =  GENRE.GENRE_ID
44     WHERE GENRE.NAME LIKE 'Rock'
45 )
46 ORDER BY EMAIL ASC;
47
```

Data Output Messages Notifications

	email character varying (50)	first_name character (50)	last_name character (50)
1	aaronmitchell@yahoo.ca	Aaron	Mitchell
2	alero@uol.com.br	Alexandre	Rocha
3	astrid.gruber@apple.at	Astrid	Gruber
4	bjorn.hansen@yahoo.no	Bjørn	Hansen
5	camille.bernard@yahoo.fr	Camille	Bernard
6	daan_peeters@apple.be	Daan	Peeters
7	diego.gutierrez@yahoo.ar	Diego	Gutiérrez
8	dmiller@comcast.com	Dan	Miller
9	dominiquelefebvre@gmail.c...	Dominique	Lefebvre
10	edfrancis@yachoo.ca	Edward	Francis
11	eduardo@woodstock.com.br	Eduardo	Martins
12	ellie.sullivan@shaw.ca	Ellie	Sullivan
13	emma_jones@hotmail.com	Emma	Jones
14	enrique_munoz@yahoo.es	Enrique	Muñoz
15	fernadaramos4@uol.com.br	Fernanda	Ramos
16	fharris@google.com	Frank	Harris
17	fralston@gmail.com	Frank	Ralston
18	ftremblay@gmail.com	François	Tremblay

```
75 WITH BEST_SELLING_ARTIST AS (
```

Total rows: 59 of 59 Query complete 00:00:00.052 Ln 38, Col 1

Music_Database/postgres@PostgreSQL 15

No limit

Query | Query History

```
44      WHERE GENRE.NAME LIKE 'Rock'
45  )
46  ORDER BY EMAIL ASC;
47
48  --7. Need to invite the artists who have written the most rock music in our dataset
49  --(Artist name and total track count of the top 10 rock bands)
50
51  SELECT ARTIST.NAME, COUNT(ARTIST.NAME) AS SONGS FROM ARTIST
52  JOIN ALBUM ON ARTIST.ARTIST_ID = ALBUM.ARTIST_ID
53  JOIN TRACK ON ALBUM.ALBUM_ID = TRACK.ALBUM_ID
54  WHERE TRACK_ID IN(
55      SELECT TRACK_ID FROM TRACK
56      JOIN GENRE ON TRACK.GENRE_ID = GENRE.GENRE_ID
57      WHERE GENRE.NAME LIKE 'Rock'
58  )
59  GROUP BY ARTIST.NAME
60  ORDER BY SONGS DESC
61  LIMIT 10;
```

Data Output | Messages | Notifications

	name character varying (120)	songs bigint
1	Led Zeppelin	114
2	U2	112
3	Deep Purple	92
4	Iron Maiden	81
5	Pearl Jam	54
6	Van Halen	52
7	Queen	45
8	The Rolling Stones	41
9	Creedence Clearwater Revival	40
10	Kiss	35

Total rows: 10 of 10 | Query complete 00:00:00.053 | Ln 51, Col 1

Music_Database/postgres@PostgreSQL 15

No limit

Query Query History

```
53 JOIN TRACK ON ALBUM.ALBUM_ID = TRACK.ALBUM_ID
54 WHERE TRACK_ID IN(
55     SELECT TRACK_ID FROM TRACK
56     JOIN GENRE ON TRACK.GENRE_ID = GENRE.GENRE_ID
57     WHERE GENRE.NAME LIKE 'Rock'
58 )
59 GROUP BY ARTIST.NAME
60 ORDER BY SONGS DESC
61 LIMIT 10;
62
63 --8. Need all the track names that have a song length longer than the average song length.
64 --(Name and Milliseconds for each track)
65
66 SELECT TRACK.NAME, TRACK.MILLISECONDS FROM TRACK
67 WHERE TRACK.MILLISECONDS > (
68     SELECT AVG(TRACK.MILLISECONDS) AS AVG_LENGTH FROM TRACK)
69 ORDER BY TRACK.MILLISECONDS DESC;
70
```

Data Output Messages Notifications

	name character varying (150)	milliseconds integer
1	Occupation / Precipice	5286953
2	Through a Looking Glass	5088838
3	Greetings from Earth, Pt. 1	2960293
4	The Man With Nine Lives	2956998
5	Battlestar Galactica, Pt. 2	2956081
6	Battlestar Galactica, Pt. 1	2952702
7	Murder On the Rising Star	2935894
8	Battlestar Galactica, Pt. 3	2927802
9	Take the Celestra	2927677
10	Fire In Space	2926593
11	The Long Patrol	2925008
12	The Magnificent Warriors	2924716
13	The Living Legend, Pt. 1	2924507
14	The Gun On Ice Planet Zero, Pt. 2	2924341
15	The Hand of God	2924007
16	Experiment In Terra	2923548
17	War of the Gods, Pt. 2	2923381
18	The Living Legend, Pt. 2	2923298
19	War of the Gods, Pt. 1	2922630

Dashboard | Properties | SQL | Statistics | Dependencies | Dependents | Processes | Music_Database/postgres@PostgreSQL 15*

Music_Database/postgres@PostgreSQL 15

No limit

Query | Query History

```sql
--9. How much amount spent by each customer on artists?
--(customer name, artist name and total spent)

WITH BEST_SELLING_ARTIST AS (
    SELECT ARTIST.ARTIST_ID AS ARTIST_ID, ARTIST.NAME AS ARTIST_NAME,
    SUM(INVOICE_LINE.UNIT_PRICE*INVOICE_LINE.QUANTITY) AS TOTAL_SALES FROM INVOICE_LINE
    JOIN TRACK ON TRACK.TRACK_ID = INVOICE_LINE.TRACK_ID
    JOIN ALBUM ON ALBUM.ALBUM_ID = TRACK.ALBUM_ID
    JOIN ARTIST ON ARTIST.ARTIST_ID = ALBUM.ARTIST_ID
    GROUP BY 1
    ORDER BY 3 DESC
    LIMIT 1
)
SELECT C.CUSTOMER_ID, C.FIRST_NAME, C.LAST_NAME, BSA.ARTIST_NAME,
SUM(IL.UNIT_PRICE * IL.QUANTITY) AS AMOUNT_SPEND FROM INVOICE I
JOIN CUSTOMER C ON I.CUSTOMER_ID = C.CUSTOMER_ID
JOIN INVOICE_LINE IL ON IL.INVOICE_ID = I.INVOICE_ID
JOIN TRACK T ON T.TRACK_ID = IL.TRACK_ID
JOIN ALBUM ALB ON ALB.ALBUM_ID = T.ALBUM_ID
JOIN BEST_SELLING_ARTIST BSA ON BSA.ARTIST_ID = ALB.ARTIST_ID
GROUP BY 1,2,3,4
ORDER BY 5 DESC;
```

Data Output | Messages | Notifications

	customer_id integer	first_name character (50)	last_name character (50)	artist_name character varying (120)	amount_spend double precision
1	46	Hugh	O'Reilly	Queen	27.719999999999985
2	38	Niklas	Schröder	Queen	18.81
3	3	François	Tremblay	Queen	17.82
4	34	João	Fernandes	Queen	16.830000000000002
5	53	Phil	Hughes	Queen	11.88
6	41	Marc	Dubois	Queen	11.88
7	47	Lucas	Mancini	Queen	10.89
8	33	Ellie	Sullivan	Queen	10.89
9	20	Dan	Miller	Queen	3.96
10	5	R	Madhav	Queen	3.96
11	23	John	Gordon	Queen	2.9699999999999998
12	54	Steve	Murray	Queen	2.9699999999999998
13	31	Martha	Silk	Queen	2.9699999999999998
14	16	Frank	Harris	Queen	1.98
15	17	Jack	Smith	Queen	1.98

Total rows: 43 of 43 | Query complete 00:00:00.044 | Ln 75, Col 1

Dashboard Properties SQL Statistics Dependencies Dependents Processes Music_Database/postgres@PostgreSQL 15*

Music_Database/postgres@PostgreSQL 15

No limit

Query Query History

```
89  JOIN TRACK T ON T.TRACK_ID = IL.TRACK_ID
90  JOIN ALBUM ALB ON ALB.ALBUM_ID = T.ALBUM_ID
91  JOIN BEST_SELLING_ARTIST BSA ON BSA.ARTIST_ID = ALB.ARTIST_ID
92  GROUP BY 1,2,3,4
93  ORDER BY 5 DESC;
94
95
96  --10. Which is the most popular music Genre for each country.
97  -- (The most popular genre is the genre with the highest amount of purchases)
98
99  WITH POPULAR_GENRE AS
100 (
101    SELECT COUNT(INVOICE_LINE.QUANTITY) AS PURCHASES, CUSTOMER.COUNTRY, GENRE.NAME, GENRE.GENRE_ID,
102     ROW_NUMBER() OVER (PARTITION BY CUSTOMER.COUNTRY ORDER BY COUNT(INVOICE_LINE.QUANTITY) DESC) AS ROWNO
103     FROM INVOICE_LINE
104     JOIN INVOICE ON INVOICE.INVOICE_ID = INVOICE_LINE.INVOICE_ID
105     JOIN CUSTOMER ON CUSTOMER.CUSTOMER_ID = INVOICE.CUSTOMER_ID
106     JOIN TRACK ON TRACK.TRACK_ID = INVOICE_LINE.TRACK_ID
107     JOIN GENRE ON GENRE.GENRE_ID = TRACK.GENRE_ID
108     GROUP BY 2,3,4
109     ORDER BY 2 ASC, 1 DESC
110 )
111 SELECT * FROM POPULAR_GENRE WHERE ROWNO <=1;
112
```

Data Output Messages Notifications

	purchases bigint	country character varying (50)	name character varying (120)	genre_id character varying (50)	rowno bigint
1	17	Argentina	Alternative & Punk	4	1
2	34	Australia	Rock	1	1
3	40	Austria	Rock	1	1
4	26	Belgium	Rock	1	1
5	205	Brazil	Rock	1	1
6	333	Canada	Rock	1	1
7	61	Chile	Rock	1	1
8	143	Czech Republic	Rock	1	1
9	24	Denmark	Rock	1	1
10	46	Finland	Rock	1	1
11	211	France	Rock	1	1
12	194	Germany	Rock	1	1
13	44	Hungary	Rock	1	1

Total rows: 24 of 24 Query complete 00:00:00.054 Ln 99, Col 1

Dashboard Properties SQL Statistics Dependencies Dependents Processes Music_Database/postgres@PostgreSQL 15*

Music_Database/postgres@PostgreSQL 15

No limit

Query Query History

```sql

--11. Which customer has spent the most on music for each country.
--(Country, Top customer and how much they spent)

WITH RECURSIVE
   CUSTOMER_WITH_COUNTRY AS (
       SELECT CUSTOMER.CUSTOMER_ID, FIRST_NAME, LAST_NAME, BILLING_COUNTRY, SUM(TOTAL) AS TOTAL_SPENDING
       FROM INVOICE
       JOIN CUSTOMER ON CUSTOMER.CUSTOMER_ID = INVOICE.CUSTOMER_ID
       GROUP BY 1,2,3,4
       ORDER BY 1, 5 DESC),

   COUNTRY_MAX_SPENDING AS (
       SELECT BILLING_COUNTRY, MAX(TOTAL_SPENDING) AS MAX_SPENDING
       FROM CUSTOMER_WITH_COUNTRY
       GROUP BY BILLING_COUNTRY)

   SELECT CC.BILLING_COUNTRY, CC.TOTAL_SPENDING, CC.FIRST_NAME, CC.LAST_NAME, CC.CUSTOMER_ID
   FROM CUSTOMER_WITH_COUNTRY CC
   JOIN COUNTRY_MAX_SPENDING MS ON CC.BILLING_COUNTRY = MS.BILLING_COUNTRY
   WHERE CC.TOTAL_SPENDING = MS.MAX_SPENDING
   ORDER BY 1;
```

Data Output Messages Notifications

	billing_country character varying (30)	total_spending double precision	first_name character (50)	last_name character (50)	customer_id integer
1	Argentina	39.6	Diego	Gutiérrez	56
2	Australia	81.18	Mark	Taylor	55
3	Austria	69.3	Astrid	Gruber	7
4	Belgium	60.38999999999999	Daan	Peeters	8
5	Brazil	108.89999999999998	Luís	Gonçalves	1
6	Canada	99.99	François	Tremblay	3
7	Chile	97.02000000000001	Luis	Rojas	57
8	Czech Republic	144.54000000000002	R	Madhav	5
9	Denmark	37.61999999999999	Kara	Nielsen	9
10	Finland	79.2	Terhi	Hämäläinen	44
11	France	99.99	Wyatt	Girard	42
12	Germany	94.05000000000001	Fynn	Zimmermann	37
13	Hungary	78.21	Ladislav	Kovács	45

Total rows: 24 of 24 Query complete 00:00:00.052 Ln 117, Col 1