SQL Assignment
Galacz Barnabas

Everything was done in sqlite because I started in that before the the sql meeting. But the last assignment couldn’t be done in sqlite so I did that in Postgresql.

Assignment 1

```
CREATE TABLE "Employer"(
"Employer_ID" INTEGER PRIMARY KEY AUTOINCREMENT,
"Full Name" TEXT,
"Joining Date" TEXT,
"Current Position" TEXT,
"Department" TEXT,
"Assigned Project(Client)" TEXT

);
----------------------------------------------------------------------------------------------------------------

CREATE TABLE "Services"(
"Software_ID" INTEGER PRIMARY KEY AUTOINCREMENT,
"Name" TEXT,
"Category" TEXT,
"Size" TEXT,
"Number of installments" INT

);
----------------------------------------------------------------------------------------------------------------


CREATE TABLE "Software_Request"(
"Employer_ID" INTEGER,
"Software_ID" INTEGER,
"Request_Start_Date" TEXT,
"Request_Close_Date" TEXT,
"Status" TEXT

);
----------------------------------------------------------------------------------------------------------------


CREATE TRIGGER newRequest AFTER INSERT ON Software_Request
	BEGIN
		UPDATE Services SET "Number of installments" ="Number of installments"+1
where Software_Request.Software_ID=NEW.Software_ID;
	END
;
----------------------------------------------------------------------------------------------------------------
```

This subtracts one when the a request is invalidated. Assuming invalidated requests won’t get updated again.
If they do it could be solved by counting the valid requests in a subselect but the assignment mentioned reduction so I went with this.

```
CREATE TRIGGER newInvalidRequest2 AFTER UPDATE ON Software_Request
        BEGIN
                UPDATE Services SET "Number of installments" ="Number of installments"-1
where Software_Request.Software_ID=NEW.Software_ID and NEW.Status="Invalid";
        END
;
```

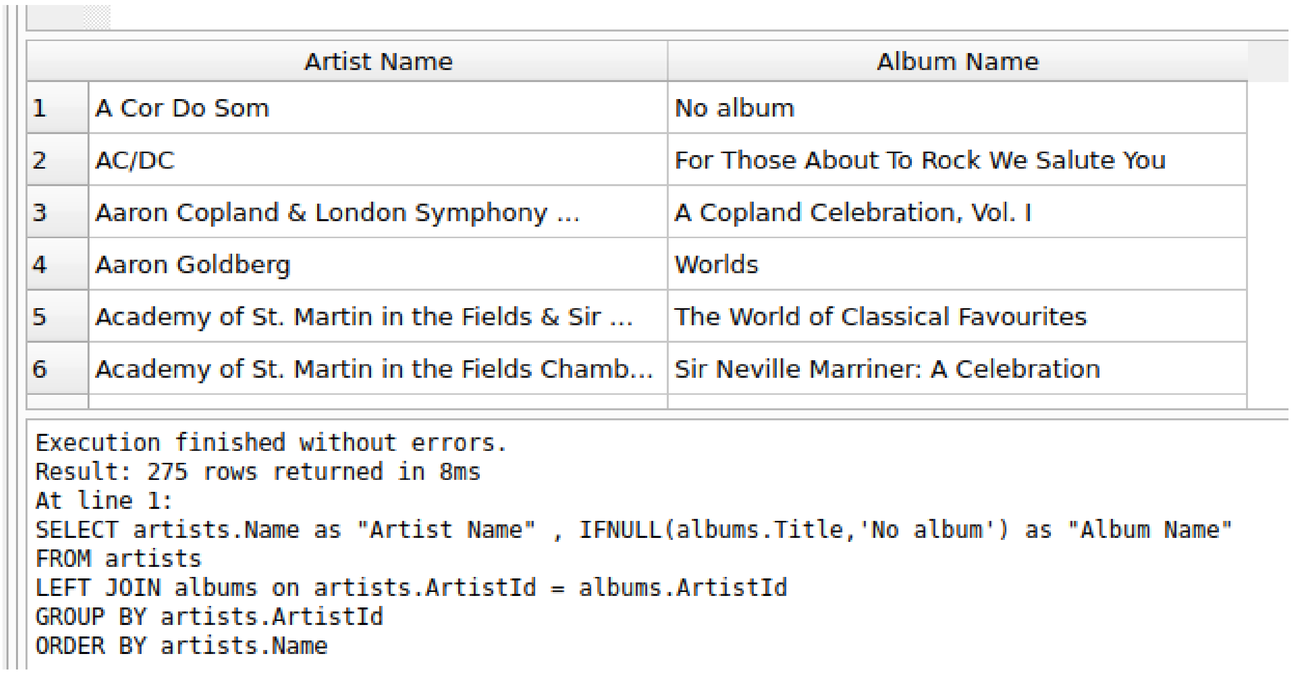
```
INSERT INTO "Employer"
("Full Name","Joining Date","Current Position",Department,"Assigned Project(Client)")
VALUES
("John Smith","2020-10-10","Leader","Finance","Leading");

INSERT INTO "Employer"
("Full Name","Joining Date","Current Position",Department,"Assigned Project(Client)")
VALUES
("Johnathan Smithson","2020-11-10","Snow shoveler","HR","Shoveling snow");

INSERT INTO "Services"
(Name,Category,Size,"Number of installments")
VALUES
("Project tracking","B","10GB","122");

INSERT INTO "Services"
(Name,Category,Size,"Number of installments")
VALUES
("Snow tracking","A","533GB","215643");

INSERT INTO "Services"
(Name,Category,Size,"Number of installments")
VALUES
("Snow tracking","A","533GB","215643");

INSERT INTO "Software_Request"
(Employer_ID,Software_ID,Request_Start_Date,Request_Close_Date,Status)
VALUES
("1","2","2012-12-12","2022-12-25","Valid");
```

Assigment 2

1

```
SELECT artists.Name as "Artist Name" , IFNULL(albums.Title,'No album') as "Album Name"
FROM artists
LEFT JOIN albums on artists.ArtistId = albums.ArtistId
GROUP BY artists.ArtistId
```

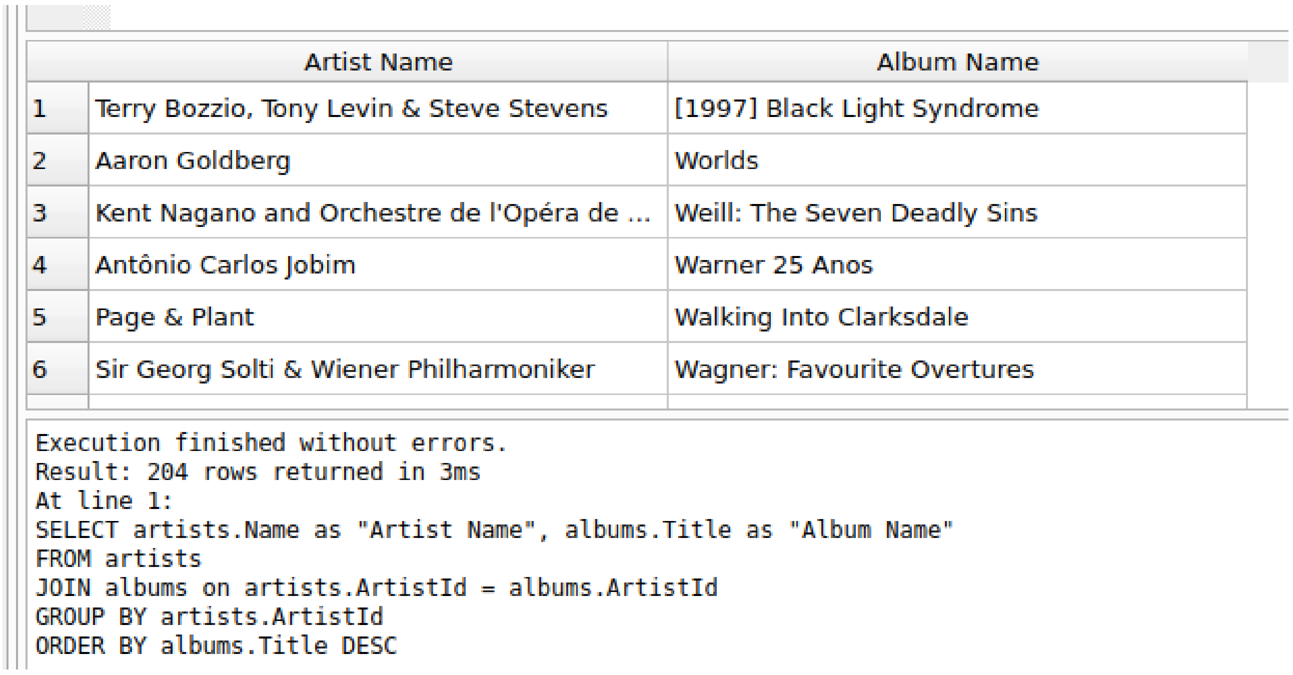
ORDER BY artists.Name



	Artist Name	Album Name
1	A Cor Do Som	No album
2	AC/DC	For Those About To Rock We Salute You
3	Aaron Copland & London Symphony ...	A Copland Celebration, Vol. I
4	Aaron Goldberg	Worlds
5	Academy of St. Martin in the Fields & Sir ...	The World of Classical Favourites
6	Academy of St. Martin in the Fields Chamb...	Sir Neville Marriner: A Celebration

```
Execution finished without errors.
Result: 275 rows returned in 8ms
At line 1:
SELECT artists.Name as "Artist Name" , IFNULL(albums.Title,'No album') as "Album Name"
FROM artists
LEFT JOIN albums on artists.ArtistId = albums.ArtistId
GROUP BY artists.ArtistId
ORDER BY artists.Name
```

2
SELECT artists.Name as "Artist Name", albums.Title as "Album Name"
FROM artists
JOIN albums on artists.ArtistId = albums.ArtistId
GROUP BY artists.ArtistId
ORDER BY albums.Title DESC

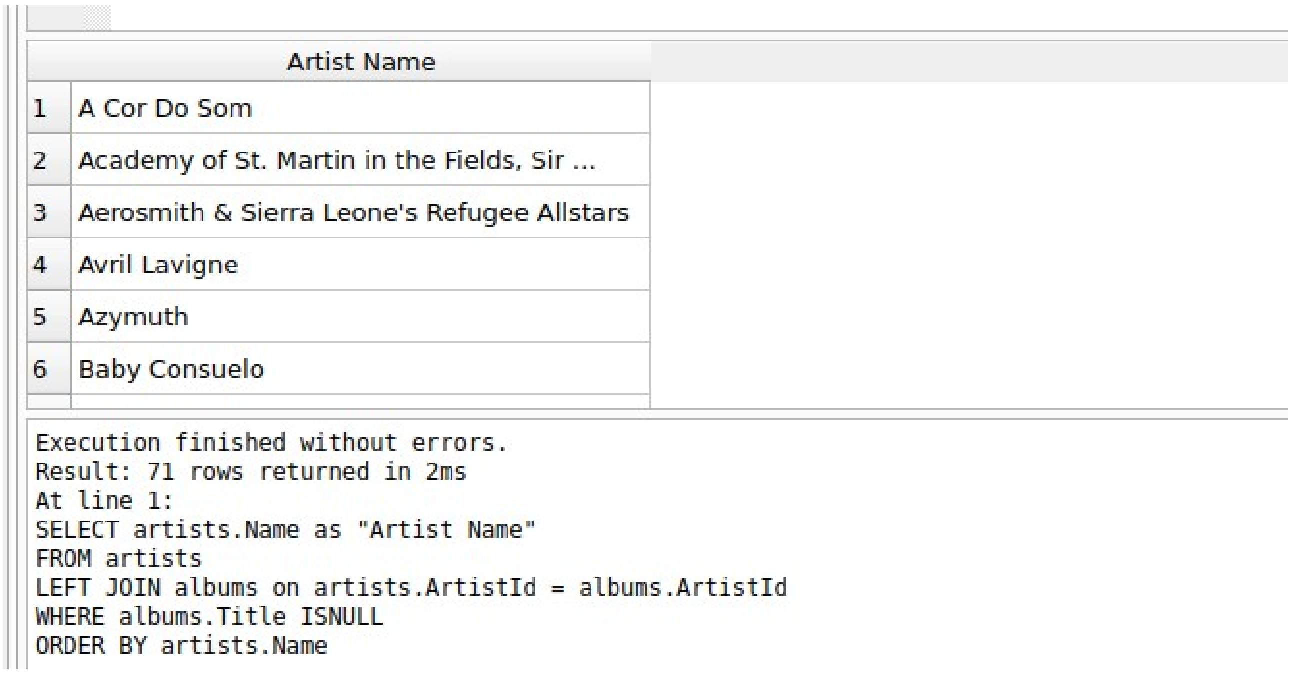


	Artist Name	Album Name
1	Terry Bozzio, Tony Levin & Steve Stevens	[1997] Black Light Syndrome
2	Aaron Goldberg	Worlds
3	Kent Nagano and Orchestre de l'Opéra de ...	Weill: The Seven Deadly Sins
4	Antônio Carlos Jobim	Warner 25 Anos
5	Page & Plant	Walking Into Clarksdale
6	Sir Georg Solti & Wiener Philharmoniker	Wagner: Favourite Overtures

```
Execution finished without errors.
Result: 204 rows returned in 3ms
At line 1:
SELECT artists.Name as "Artist Name", albums.Title as "Album Name"
FROM artists
JOIN albums on artists.ArtistId = albums.ArtistId
GROUP BY artists.ArtistId
ORDER BY albums.Title DESC
```

3
SELECT artists.Name as "Artist Name"
FROM artists
LEFT JOIN albums on artists.ArtistId = albums.ArtistId
WHERE albums.Title ISNULL

ORDER BY artists.Name

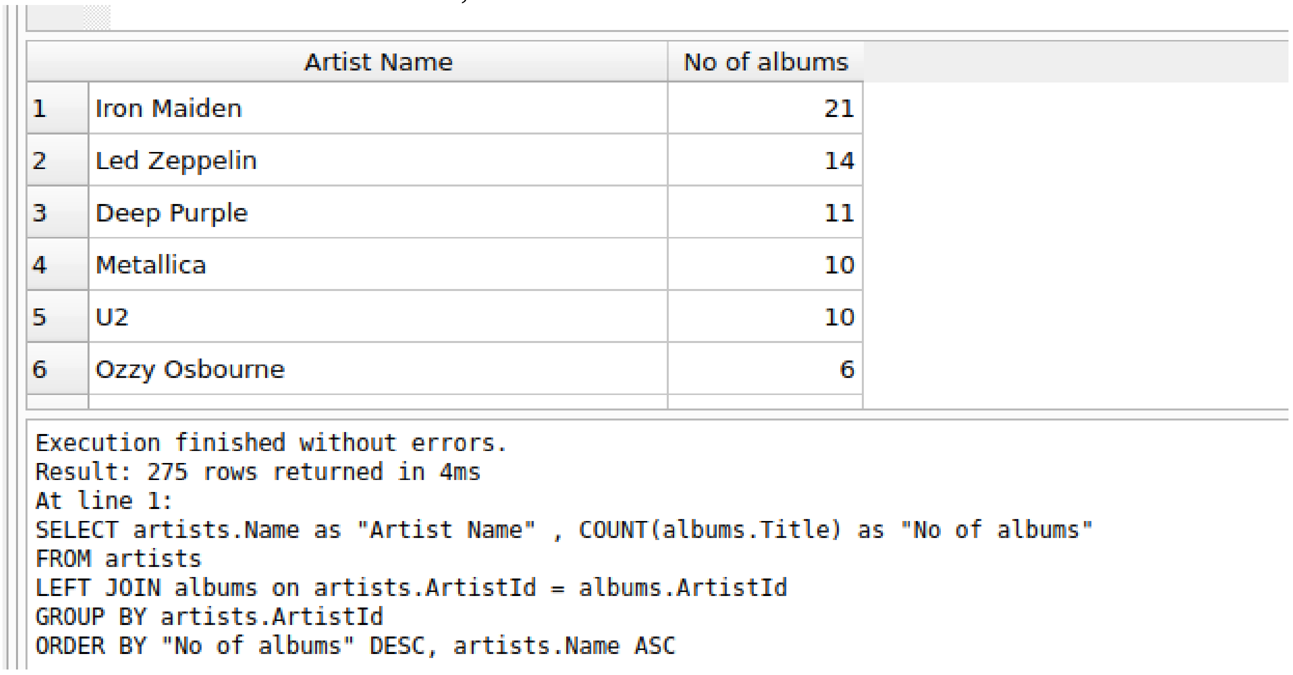


	Artist Name
1	A Cor Do Som
2	Academy of St. Martin in the Fields, Sir ...
3	Aerosmith & Sierra Leone's Refugee Allstars
4	Avril Lavigne
5	Azymuth
6	Baby Consuelo

```
Execution finished without errors.
Result: 71 rows returned in 2ms
At line 1:
SELECT artists.Name as "Artist Name"
FROM artists
LEFT JOIN albums on artists.ArtistId = albums.ArtistId
WHERE albums.Title ISNULL
ORDER BY artists.Name
```

4

SELECT artists.Name as "Artist Name" , COUNT(albums.Title) as "No of albums"
FROM artists
LEFT JOIN albums on artists.ArtistId = albums.ArtistId
GROUP BY artists.ArtistId
ORDER BY "No of albums" DESC, artists.Name ASC



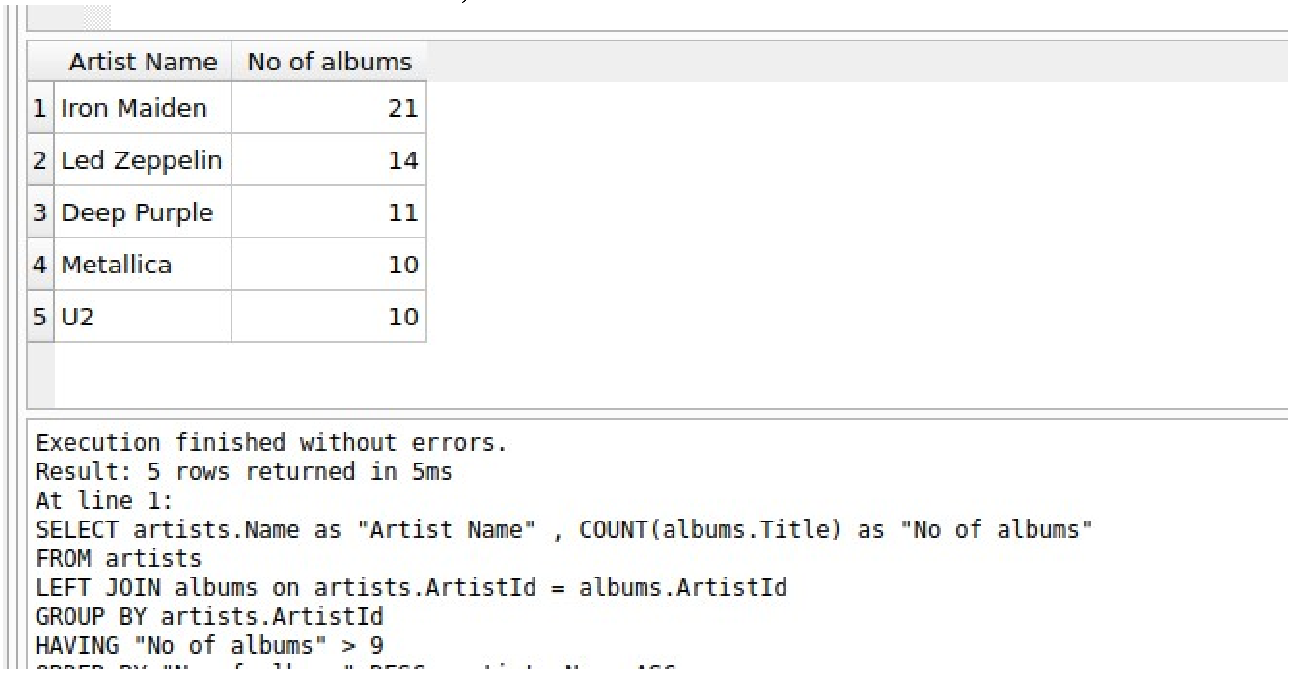
	Artist Name	No of albums
1	Iron Maiden	21
2	Led Zeppelin	14
3	Deep Purple	11
4	Metallica	10
5	U2	10
6	Ozzy Osbourne	6

```
Execution finished without errors.
Result: 275 rows returned in 4ms
At line 1:
SELECT artists.Name as "Artist Name" , COUNT(albums.Title) as "No of albums"
FROM artists
LEFT JOIN albums on artists.ArtistId = albums.ArtistId
GROUP BY artists.ArtistId
ORDER BY "No of albums" DESC, artists.Name ASC
```

5

SELECT artists.Name as "Artist Name" , COUNT(albums.Title) as "No of albums"
FROM artists
LEFT JOIN albums on artists.ArtistId = albums.ArtistId
GROUP BY artists.ArtistId

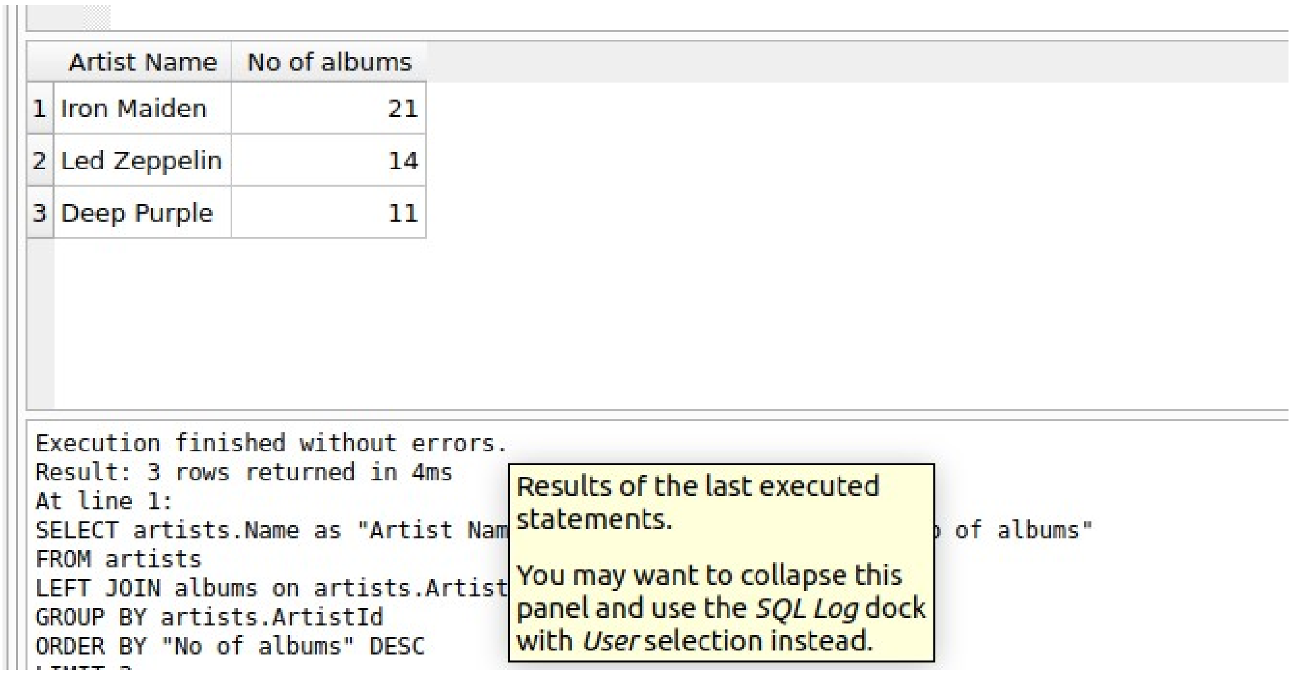
HAVING "No of albums" > 9
ORDER BY "No of albums" DESC, artists.Name ASC



	Artist Name	No of albums
1	Iron Maiden	21
2	Led Zeppelin	14
3	Deep Purple	11
4	Metallica	10
5	U2	10

```
Execution finished without errors.
Result: 5 rows returned in 5ms
At line 1:
SELECT artists.Name as "Artist Name" , COUNT(albums.Title) as "No of albums"
FROM artists
LEFT JOIN albums on artists.ArtistId = albums.ArtistId
GROUP BY artists.ArtistId
HAVING "No of albums" > 9
```

6
SELECT artists.Name as "Artist Name" , COUNT(albums.Title) as "No of albums"
FROM artists
LEFT JOIN albums on artists.ArtistId = albums.ArtistId
GROUP BY artists.ArtistId
ORDER BY "No of albums" DESC
LIMIT 3



	Artist Name	No of albums
1	Iron Maiden	21
2	Led Zeppelin	14
3	Deep Purple	11

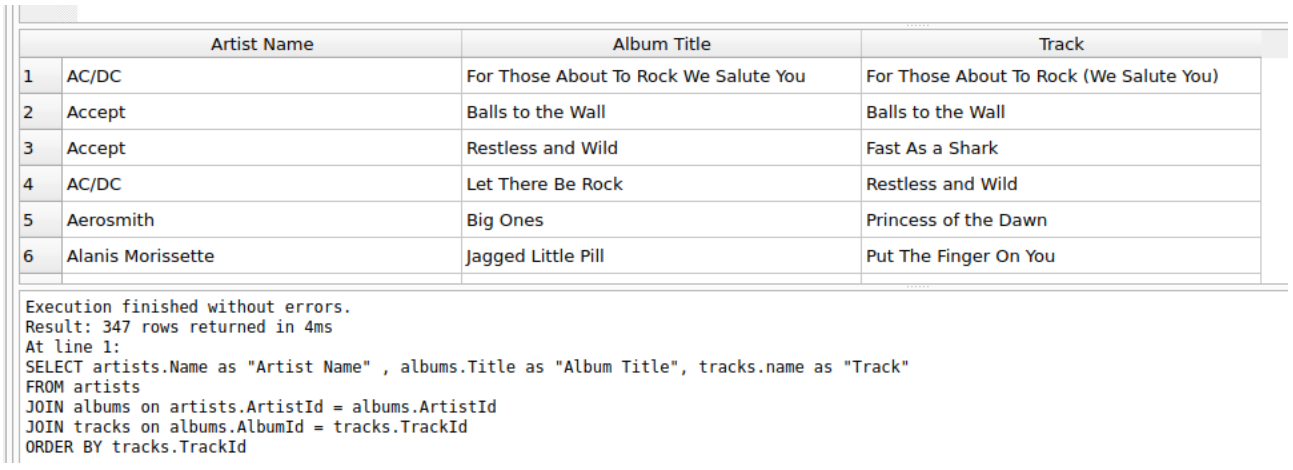
```
Execution finished without errors.
Result: 3 rows returned in 4ms
At line 1:
SELECT artists.Name as "Artist Nam
FROM artists
LEFT JOIN albums on artists.Artist
GROUP BY artists.ArtistId
ORDER BY "No of albums" DESC
```

Results of the last executed statements.

You may want to collapse this panel and use the *SQL Log* dock with *User* selection instead.

7
SELECT artists.Name as "Artist Name" , albums.Title as "Album Title", tracks.name as "Track"
FROM artists

JOIN albums on artists.ArtistId = albums.ArtistId
JOIN tracks on albums.AlbumId = tracks.TrackId
ORDER BY tracks.TrackId

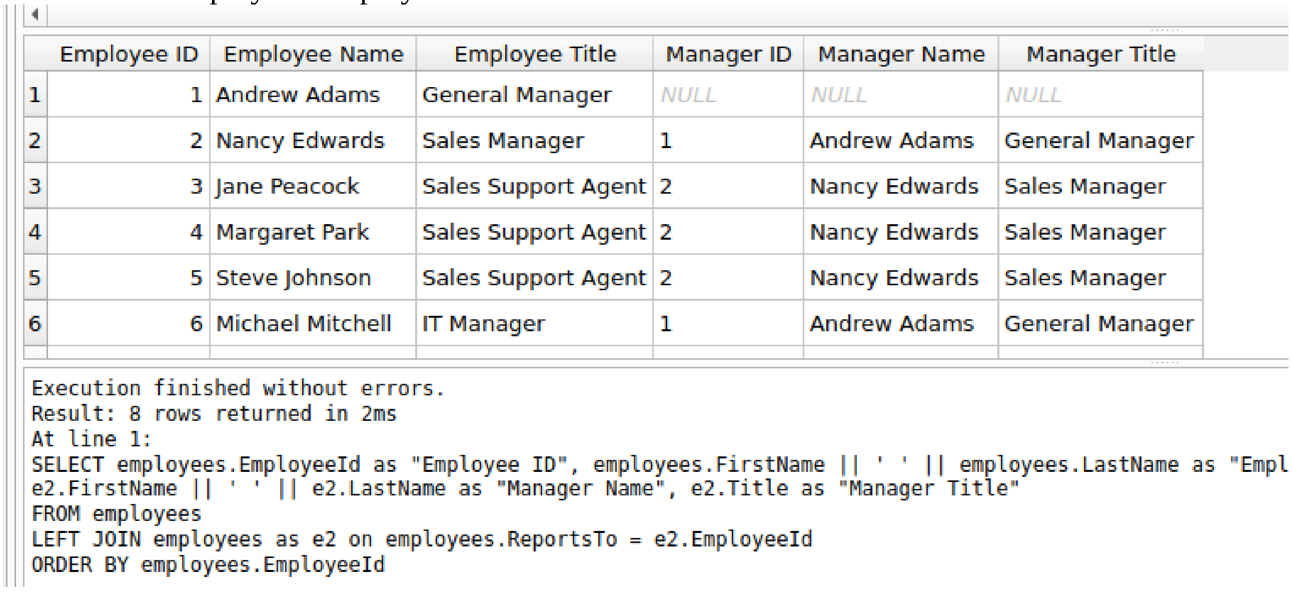


	Artist Name	Album Title	Track
1	AC/DC	For Those About To Rock We Salute You	For Those About To Rock (We Salute You)
2	Accept	Balls to the Wall	Balls to the Wall
3	Accept	Restless and Wild	Fast As a Shark
4	AC/DC	Let There Be Rock	Restless and Wild
5	Aerosmith	Big Ones	Princess of the Dawn
6	Alanis Morissette	Jagged Little Pill	Put The Finger On You

```
Execution finished without errors.
Result: 347 rows returned in 4ms
At line 1:
SELECT artists.Name as "Artist Name" , albums.Title as "Album Title", tracks.name as "Track"
FROM artists
JOIN albums on artists.ArtistId = albums.ArtistId
JOIN tracks on albums.AlbumId = tracks.TrackId
ORDER BY tracks.TrackId
```

8

SELECT employees.EmployeeId as "Employee ID", employees.FirstName || ' ' || employees.LastName as "Employee Name", employees.Title as "Employee Title", employees.ReportsTo "Manager ID", e2.FirstName || ' ' || e2.LastName as "Manager Name", e2.Title as "Manager Title"
FROM employees
LEFT JOIN employees as e2 on employees.ReportsTo = e2.EmployeeId
ORDER BY employees.EmployeeId



	Employee ID	Employee Name	Employee Title	Manager ID	Manager Name	Manager Title
1	1	Andrew Adams	General Manager	NULL	NULL	NULL
2	2	Nancy Edwards	Sales Manager	1	Andrew Adams	General Manager
3	3	Jane Peacock	Sales Support Agent	2	Nancy Edwards	Sales Manager
4	4	Margaret Park	Sales Support Agent	2	Nancy Edwards	Sales Manager
5	5	Steve Johnson	Sales Support Agent	2	Nancy Edwards	Sales Manager
6	6	Michael Mitchell	IT Manager	1	Andrew Adams	General Manager

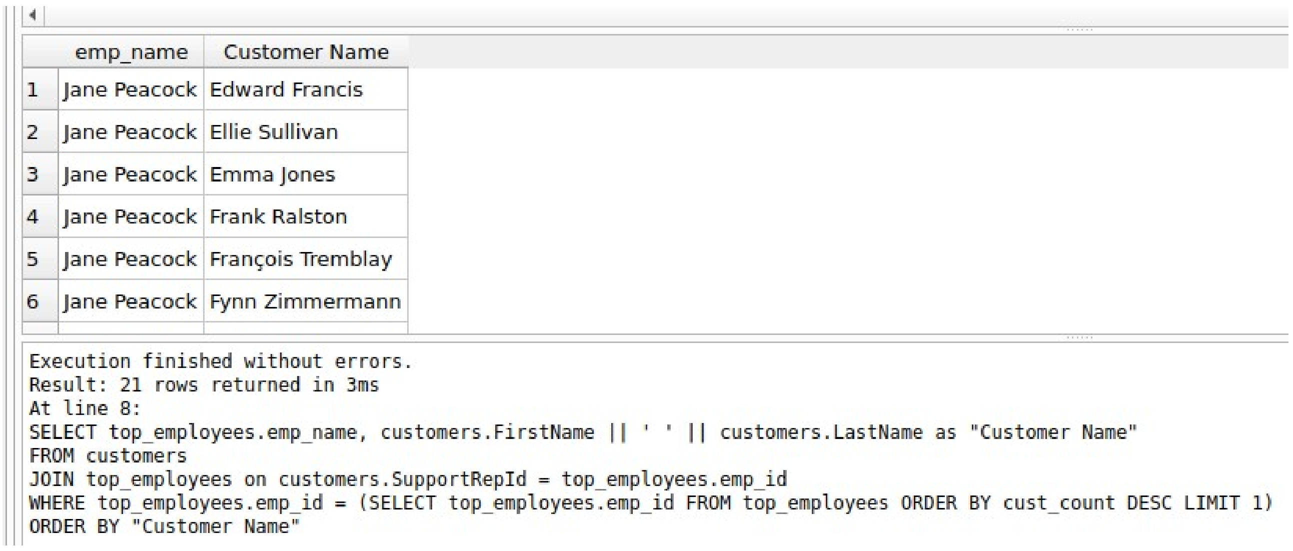
```
Execution finished without errors.
Result: 8 rows returned in 2ms
At line 1:
SELECT employees.EmployeeId as "Employee ID", employees.FirstName || ' ' || employees.LastName as "Empl
e2.FirstName || ' ' || e2.LastName as "Manager Name", e2.Title as "Manager Title"
FROM employees
LEFT JOIN employees as e2 on employees.ReportsTo = e2.EmployeeId
ORDER BY employees.EmployeeId
```

9

CREATE VIEW top_employees AS
SELECT employees.EmployeeId as emp_id, employees.FirstName || ' ' || employees.LastName as emp_name, count(customers.CustomerId) as cust_count
FROM employees
JOIN customers on employees.EmployeeId = customers.SupportRepId
GROUP BY EmployeeId

ORDER BY cust_count DESC;

SELECT top_employees.emp_name, customers.FirstName || ' ' || customers.LastName as "Customer Name"
FROM customers
JOIN top_employees on customers.SupportRepId = top_employees.emp_id
WHERE top_employees.emp_id = (SELECT top_employees.emp_id FROM top_employees
ORDER BY cust_count DESC LIMIT 1)
ORDER BY "Customer Name"



	emp_name	Customer Name
1	Jane Peacock	Edward Francis
2	Jane Peacock	Ellie Sullivan
3	Jane Peacock	Emma Jones
4	Jane Peacock	Frank Ralston
5	Jane Peacock	François Tremblay
6	Jane Peacock	Fynn Zimmermann

```
Execution finished without errors.
Result: 21 rows returned in 3ms
At line 8:
SELECT top_employees.emp_name, customers.FirstName || ' ' || customers.LastName as "Customer Name"
FROM customers
JOIN top_employees on customers.SupportRepId = top_employees.emp_id
WHERE top_employees.emp_id = (SELECT top_employees.emp_id FROM top_employees ORDER BY cust_count DESC LIMIT 1)
ORDER BY "Customer Name"
```

10

INSERT INTO media_types
(Name)
VALUES
('MP3')

I could have chosen the id of mp3 with a subselect but that seemed unnecessarily complicated. Also if I am right this one only works with transactions, another option would be to delete the inserted row instantly after it was inserted but in the tutorial we received this was mentioned so I chose this option.

```
CREATE TRIGGER newMP3Insert BEFORE INSERT ON tracks
	BEGIN
		SELECT RAISE (ROLLBACK, 'cannot insert mp3 media type into tracks') FROM
tracks
			WHERE TrackId = NEW.TrackId AND MediaTypeId = '6';
	END
```

11

```
CREATE TABLE "tracks_audit_log"(
	id serial primary key,
        operation character varying(200),
        datetime date DEFAULT current_date,
        username character varying(200),
        old_value character varying(200),
        new_value character varying(200)
);



CREATE OR REPLACE FUNCTION process_tracks_audit_log() RETURNS TRIGGER AS
$tracks_audit_log$
  BEGIN

    IF (TG_OP = 'DELETE') THEN
      INSERT INTO tracks_audit_log (operation,username,old_value) VALUES
('DELETE',current_user,CONCAT(OLD.Name,' ',OLD.AlbumId,' ',OLD.MediaTypeId,'
',OLD.GenreId,' ',OLD.Composer,' ',OLD.Milliseconds,' ',OLD.Bytes,' ',OLD.UnitPrice));
    ELSIF (TG_OP = 'INSERT') THEN
                INSERT INTO tracks_audit_log (operation,username,new_value) VALUES
('INSERT',current_user,CONCAT(NEW.Name,' ',NEW.AlbumId,' ',NEW.MediaTypeId,'
',NEW.GenreId,' ',NEW.Composer,' ',NEW.Milliseconds,' ',NEW.Bytes,' ',NEW.UnitPrice));
        ELSIF (TG_OP = 'INSERT') THEN
                INSERT INTO tracks_audit_log (operation,username,old_value,new_value)
VALUES ('UPDATE',current_user,CONCAT(OLD.Name,' ',OLD.AlbumId,' ',OLD.MediaTypeId,'
',OLD.GenreId,' ',OLD.Composer,' ',OLD.Milliseconds,' ',OLD.Bytes,'
',OLD.UnitPrice),CONCAT(NEW.Name,' ',NEW.AlbumId,' ',NEW.MediaTypeId,' ',NEW.GenreId,'
',NEW.Composer,' ',NEW.Milliseconds,' ',NEW.Bytes,' ',NEW.UnitPrice));
    END IF;
        RETURN NULL;
  END;
$tracks_audit_log$ LANGUAGE plpgsql;

CREATE TRIGGER tracks_audit
AFTER INSERT OR UPDATE OR DELETE ON tracks
   FOR EACH ROW EXECUTE FUNCTION process_tracks_audit_log();
```