

Visual Studio Code interface showing a SQLite database project named CHINOOK. The Explorer pane on the left shows the project structure: PROGRA, CHINOOK, .env, SQLite.sql, chinook.db, and Documento sin título.pdf. The CHINOOK folder is expanded, showing the SQLite.sql file.

The CHINOOK > SQLite.sql file contains the following SQL code:

```
2 CREATE TABLE CustomerComments (
11 VALUES
47 ORDER BY DineroTotal DESC;
48
49 SELECT AlbumId, COUNT(*) AS NumCanciones
50 FROM Tracks
51 GROUP BY AlbumId
52 ORDER BY NumCanciones DESC;
53
54 SELECT Customers.FirstName, Customers.LastName
55 FROM Customers
56 LEFT JOIN Invoices ON Customers.CustomerId = Invoices.CustomerId
57 WHERE Invoices.Total IS NULL;
58
59 SELECT Tracks.Name AS SongName, Albums.Title AS NameAlbum
60 FROM Tracks
61 INNER JOIN Albums ON Tracks.AlbumId = Albums.AlbumId;
62
63 SELECT Invoices.InvoiceId, Invoices.Total, Customers.Country
64 FROM Invoices
65 LEFT JOIN Customers ON Invoices.CustomerId = Customers.CustomerId;
66
67 SELECT Genres.Name AS SongGenre, COUNT(Tracks.TrackId) AS NumTracks
68 FROM Tracks
69 INNER JOIN Genres ON Tracks.GenreId = Genres.GenreId
70 GROUP BY Genres.Name;
```

The right pane shows the SQLite database viewer. The top section displays the SQLite schema, and the bottom section shows the results of a query. The query results are displayed in a table with two columns: SongName and NameAlbum. The table contains 20 rows of data, including songs like "For Those About To Rock (We Salute You)", "Put The Finger On You", "Let's Get It Up", "Inject The Venom", "Snowballed", "Evil Walks", "C.O.D.", "Breaking The Rules", "Night Of The Long Knives", "Spellbound", "Balls to the Wall", "Fast As a Shark", "Restless and Wild", "Princess of the Dawn", "Go Down", "Dog Eat Dog", "Let There Be Rock", "Bad Boy Boogie", "Problem Child", "Overdose", "Hell Ain't A Bad Place To Be", "Whole Lotta Rosie", "Walk On Water", and "Love In An Elevator".

Visual Studio Code interface showing the same SQLite database project. The Explorer pane on the left shows the project structure: PROGRA, CHINOOK, .env, SQLite.sql, chinook.db, and Documento sin título.pdf. The CHINOOK folder is expanded, showing the SQLite.sql file.

The CHINOOK > SQLite.sql file contains the following SQL code:

```
2 CREATE TABLE CustomerComments (
11 VALUES
47 ORDER BY DineroTotal DESC;
48
49 SELECT AlbumId, COUNT(*) AS Nu
50 FROM Tracks
51 GROUP BY AlbumId
52 ORDER BY NumCanciones DESC;
53
54 SELECT Customers.FirstName, Cu
55 FROM Customers
56 LEFT JOIN Invoices ON Customer
57 WHERE Invoices.Total IS NULL;
58
59 SELECT Tracks.Name AS SongName
60 FROM Tracks
61 INNER JOIN Albums ON Tracks.Al
62
63 SELECT Invoices.InvoiceId, Inv
64 FROM Invoices
65 LEFT JOIN Customers ON Invoice
66
67 SELECT Genres.Name AS SongGenr
68 FROM Tracks
69 INNER JOIN Genres ON Tracks.Ge
70 GROUP BY Genres.Name;
```

The right pane shows the SQLite database viewer. The top section displays the SQLite schema, and the bottom section shows the results of a query. The query results are displayed in a table with two columns: SongName and NameAlbum. The table contains 20 rows of data, including songs like "For Those About To Rock (We Salute You)", "Put The Finger On You", "Let's Get It Up", "Inject The Venom", "Snowballed", "Evil Walks", "C.O.D.", "Breaking The Rules", "Night Of The Long Knives", "Spellbound", "Balls to the Wall", "Fast As a Shark", "Restless and Wild", "Princess of the Dawn", "Go Down", "Dog Eat Dog", "Let There Be Rock", "Bad Boy Boogie", "Problem Child", "Overdose", "Hell Ain't A Bad Place To Be", "Whole Lotta Rosie", "Walk On Water", and "Love In An Elevator".

CHINOOK > -- SQLite.sql

```

2 CREATE TABLE CustomerComments (
11 VALUES
47 ORDER BY DineroTotal DESC;
48
49 SELECT AlbumId, COUNT(*) AS NumTracks
50 FROM Tracks
51 GROUP BY AlbumId
52 ORDER BY NumCanciones DESC;
53
54 SELECT Customers.FirstName, Customers.LastName
55 FROM Customers
56 LEFT JOIN Invoices ON CustomerId = Invoices.CustomerId
57 WHERE Invoices.Total IS NULL;
58
59 SELECT Tracks.Name AS SongName
60 FROM Tracks
61 INNER JOIN Albums ON Tracks.AlbumId = Albums.AlbumId
62
63 SELECT Invoices.InvoiceId, Invoices.Total
64 FROM Invoices
65 LEFT JOIN Customers ON InvoiceId = Customers.CustomerId
66
67 SELECT Genres.Name AS SongGenre
68 FROM Tracks
69 INNER JOIN Genres ON Tracks.GenreId = Genres.GenreId
70 GROUP BY Genres.Name;

```

InvoiceId	Total	Country
1	1.98	Germany
2	3.96	Norway
3	5.94	Belgium
4	8.91	Canada
5	13.86	USA
6	0.99	Germany
7	1.98	Germany
8	1.98	France
9	3.96	France
10	5.94	Ireland
11	8.91	United Kingdom
12	13.86	Germany
13	0.99	USA
14	1.98	USA
15	1.98	USA
16	3.96	USA
17	5.94	USA
18	8.91	Canada
19	13.86	France
20	0.99	United Kingdom
21	1.98	Australia
22	1.98	Chile
23	3.96	India
24	5.94	Norway
25	8.91	Brazil
26	13.86	USA
27	0.99	Canada
28	1.98	Portugal

CHINOOK > -- SQLite.sql

```

2 CREATE TABLE CustomerComments (
11 VALUES
47 ORDER BY DineroTotal DESC;
48
49 SELECT AlbumId, COUNT(*) AS NumTracks
50 FROM Tracks
51 GROUP BY AlbumId
52 ORDER BY NumCanciones DESC;
53
54 SELECT Customers.FirstName, Customers.LastName
55 FROM Customers
56 LEFT JOIN Invoices ON CustomerId = Invoices.CustomerId
57 WHERE Invoices.Total IS NULL;
58
59 SELECT Tracks.Name AS SongName
60 FROM Tracks
61 INNER JOIN Albums ON Tracks.AlbumId = Albums.AlbumId
62
63 SELECT Invoices.InvoiceId, Invoices.Total
64 FROM Invoices
65 LEFT JOIN Customers ON InvoiceId = Customers.CustomerId
66
67 SELECT Genres.Name AS SongGenre
68 FROM Tracks
69 INNER JOIN Genres ON Tracks.GenreId = Genres.GenreId
70 GROUP BY Genres.Name;

```

SongGenre	NumTracks
Alternative	40
Alternative & Punk	332
Blues	81
Bossa Nova	15
Classical	74
Comedy	17
Drama	64
Easy Listening	24
Electronica/Dance	30
Heavy Metal	28
Hip Hop/Rap	35
Jazz	130
Latin	579
Metal	374
Opera	1
Pop	48
R&B/Soul	61
Reggae	58
Rock	1297
Rock And Roll	12
Sci Fi & Fantasy	26
Science Fiction	13
Soundtrack	43
TV Shows	93
World	28

1: Selecciona el primer nombre y apellido del cliente de la sección de customers y hace un LEFT JOIN de Invoices en el CustomerID de los clientes y hace Customer ID en Invoices y pone una condición en donde sea NULL el total de los Invoices

2: Selecciona el nombre de la canción y la nombra como Song Name y selecciona el nombre del album y lo nombra como NameAlbum desde la sección de Tracks y hace un INNER JOIN de la sección albums en el AlbumId de Tracks y pone un Album ID en Albums

3: Selecciona el Invoice Id de Invoices , el total de Invoices y el país de costumers desde Invoices y hace un LEFT JOIN en customer's de el Customer Id de invoices y hace un customer Id en costumers

4: Selecciona el nombre del genero y lo nombre como SongGenre y cuenta los ids de Tracks y los puse con NumTracks desde la seccion Tracks y hice un INNER