Table of Contents

1. **MUSIC LIBRARY MANAGEMENT SYSTEM** 2
    1.1. **Introduction** 2
2. **MUSIC LIBRARY MANAGEMENT SYSTEM CASE STUDY** 3
3. **MUSIC LIBRARY MANAGEMENT SYSTEM ER DIAGRAM** 4
4. **MUSIC LIBRARY MANAGEMENT SYSTEM NORMALIZATION** 5
    4.1. **Normalization** 5
    4.2. **Finalization** 7
    4.3. **Final Tables** 8
5. **MUSIC LIBRARY MANAGEMENT SYSTEM TABLE CREATION** 9
6. **MUSIC LIBRARY MANAGEMENT SYSTEM DATA INSERTION** 15
7. **MUSIC LIBRARY MANAGEMENT SYSTEM QUERIES** 22
    7.1. **Single Row Subquery** 22
    7.2. **Multiple Row Subquery** 23
    7.3. **Aggregate Function** 24
    7.4. **Joining** 24
        7.4.1.**Equijoin** 24
        7.4.2.**Outer Join** 25
    7.5. **View Creation:** 26
        7.5.1.**Simple View** 26
        7.5.2.**Complex View** 27

MUSIC LIBRARY MANAGEMENT SYSTEM

Introduction:

TrackTrove is a music library management system. This database management system is created using Oracle_10g_XE application.

TrackTrove contains a vast collection of tracks. It is enriched with tracks of different genres. TrackTrove users can create playlists containing their favorite tracks.

TrackTrove strives to provide information of a track to the best extent possible. The artist of a track, Album the track belongs to are all included in this music library.

MUSIC LIBRARY MANAGEMENT SYSTEM CASE STUDY

In a music library management system, a user can listen to tracks/songs. A user can listen to many tracks and one track can also be listened to by many users. The system stores the user id (primary key), username, password, and user email. A track is identified by the track id, track name, track creation date, and duration. Users can make playlists.

One user can make many playlists, but one playlist can only be made by exactly one user. The playlist
contains tracks. One playlist can contain many tracks and one track can be in many playlists. The system stores the playlist id, playlist name, and playlist created date. Artist composes track.

A track may be composed by many artists and one artist can compose many tracks. The system stores the artist id, artist name, and nationality of the artists who created at least one track.

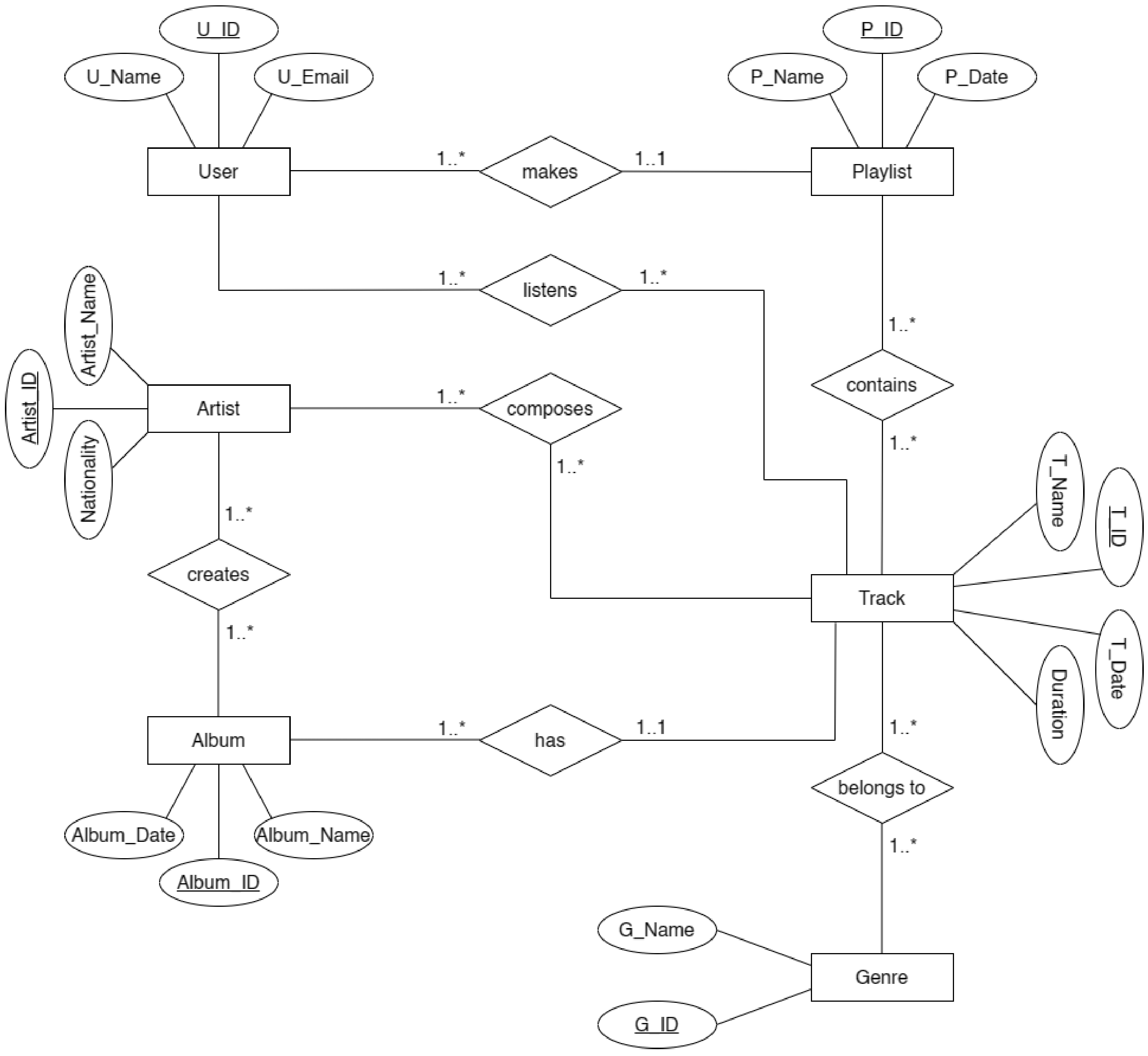
An artist creates an Album. An album may be created by many artists and one artist can create many albums. Albums are identified by album id, album name, and album date created.

One artist has to create at least one track or an album.

Album has tracks. One album may contain many tracks and one track can only belong to one album.

Tracks belong to genres. One track can belong to multiple genres and for one genre there can be multiple tracks. Genre is specified by genre id, and genre name.

MUSIC LIBRARY MANAGEMENT SYSTEM ER DIAGRAM



MUSIC LIBRARY MANAGEMENT SYSTEM NORMALIZATION

Normalization

UNF (makes): U_ID, U_Name, U_Email, Password, P_ID, P_Name, P_Date

1NF: **U-ID**, U_Name, U_Email, Password

P-ID, P_Name, P_Date

2NF: 1) **U-ID**, U_Name, U_Email, Password, P_ID (FK)

2) **P-ID**, P_Name, P_Date

3NF: Same as 2NF

UNF (contains): P_ID, P_Name, P_Date, T_Date, T_Name, T_ID, Duration

1NF: **P-ID**, P_Name, P_Date

T-ID, T_Name, T_Date, Duration

2NF: 1) **P-ID**, P_Name, P_Date

2) **T-ID**, T_Name, T_Date, Duration

3) P_ID (PK), T_ID (FK) / P_ID (FK), T_ID (PK)

3NF: Same as 2NF

UNF (listens): U_ID, U_Name, U_Email, Password, T_Date, T_Name, T_ID, Duration

1NF: **U-ID**, U_Name, U_Email, Password

T-ID, T_Date, T_Name, Duration

2NF: 1) **U-ID**, U_Name, U_Email, Password

2) **T-ID**, T_Date, T_Name, Duration

3) U_ID (PK), T_ID (FK) / U_ID (FK), T_ID (PK)

3NF: Same as 2NF

UNF (belongs): T_Date, T_Name, T_ID, Duration, G_ID, G_Name

1NF: **T-ID**, T_Date, T_Name, Duration, **G-ID**, G_Name

2NF: 1) **T-ID**, T_Date, T_Name, Duration

2) **G-ID**, G_Name

3) T_ID (PK), G_ID (FK) / T_ID (FK), G_ID (PK)

3NF: Same as 2NF

UNF (composes): Artist_ID, Artist_Name, Nationality, T_Date, T_ID, T_Name, Duration

1NF: **Artist-ID**, Artist_Name, Nationality

T-ID, T_Date, T_Name, Duration

2NF: 1) **Artist-ID**, Artist_Name, Nationality

2) **T-ID** ,T_Date, T_Name, Duration

3) Artist_ID (PK), T_ID (FK) / Artist_ID (FK), T_ID (PK)

3NF: Same as 2NF

UNF (creates): Artist_ID, Artist_Name, Nationality, Album_ID, Album_Name, Album_date

1NF: **Artist-ID**, Artist_Name, Nationality

Album-ID, Album_Name, Album_date

2NF: 1) **Artist-ID**, Artist_Name, Nationality

2) **Album-ID**, Album_Name, Album_date

3) Artist_ID (PK), Album_ID (FK) / Artist_ID(FK), Album_ID (PK)

3NF: Same as 2NF

UNF (has): Album_ID, Album_Name, Album_Date, T_Date, T_Name, T_ID, Duration

1NF: **Album-ID**, Album_Name, Album_Date

T-ID, T_Name, T_Date, Duration

2NF: 1) **Album-ID**, Album_Name, Album_Date, T_ID (FK)

2) **T-ID**, T_Name, T_Date, Duration

3NF: Same as 2NF

Finalization

1) U-ID, U_Name, U_Email, Password, P_ID (FK)

2) P-ID, P_Name, P_Date

~~3) P-ID, P_Name, P_Date~~

4) T-ID, T_Name, T_Date, Duration

5) P_ID (PK), T_ID (FK)

6) U-ID, U_Name, U_Email, Password

~~7) T-ID, T_Date, T_Name, Duration~~

8) U_ID (PK), T_ID (FK)

~~9) T-ID, T_Date, T_Name, Duration~~

10) G-ID, G_Name

11) T_ID (PK), G_ID (FK)

12) Artist-ID, Artist_Name, Nationality

~~13) T-ID ,T_Date, T_Name, Duration~~

14) Artist_ID (PK), T_ID (FK)

~~15) Artist-ID, Artist_Name, Nationality~~

16) Album-ID, Album_Name, Album_date

17) Artist_ID (PK), Album_ID (FK)

18) Album-ID, Album_Name, Album_Date, T_ID (FK)

~~19) T-ID, T_Name, T_Date, Duration~~

Final Tables

1)**U-ID**, U_Name, U_Email, Password

2) **P-ID**, P_Name, P_Date

3) **U-ID**, U_Name, U_Email, Password, P_ID (FK)

4) **T-ID**, T_Name, T_Date, Duration

5 **Artist-ID**, Artist_Name, Nationality

6)**Album-ID**, Album_Name, Album_Date

7) **Album-ID**, Album_Name, Album_Date, T_ID (FK)

8) **G-ID**, G_Name

9) P_ID (PK), T_ID (FK)

10) U_ID (PK), T_ID (FK)

11) T_ID (PK), G_ID (FK)

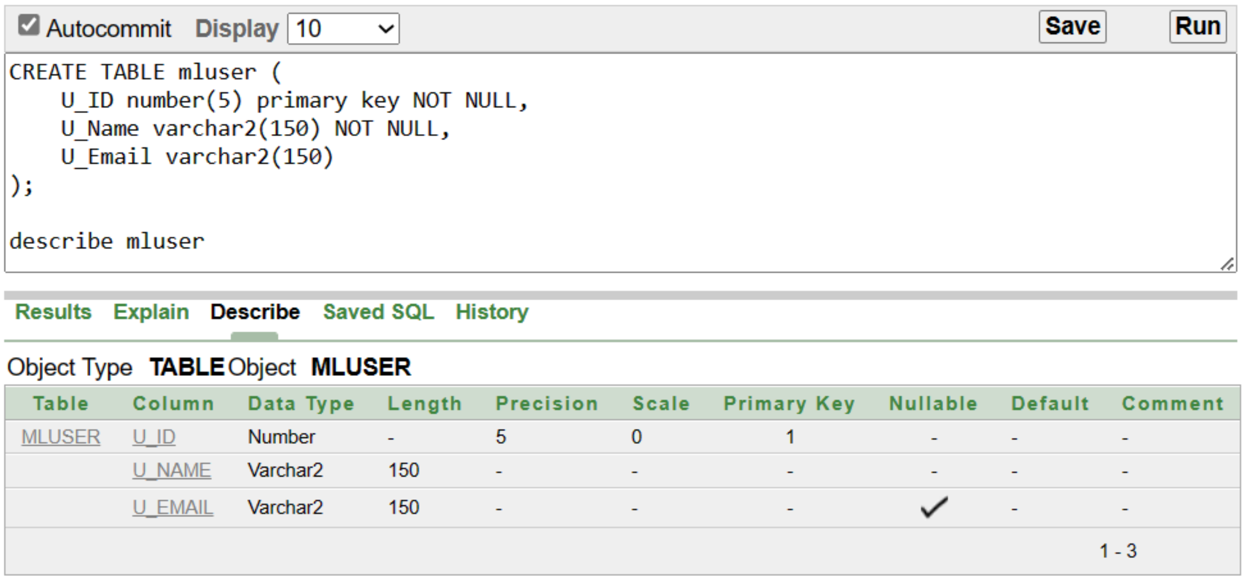
12) Artist_ID (PK), T_ID (FK)

13) Artist_ID (PK), Album_ID (FK)

Total tables: 13

MUSIC LIBRARY MANAGEMENT SYSTEM TABLE CREATION

1) mluser:



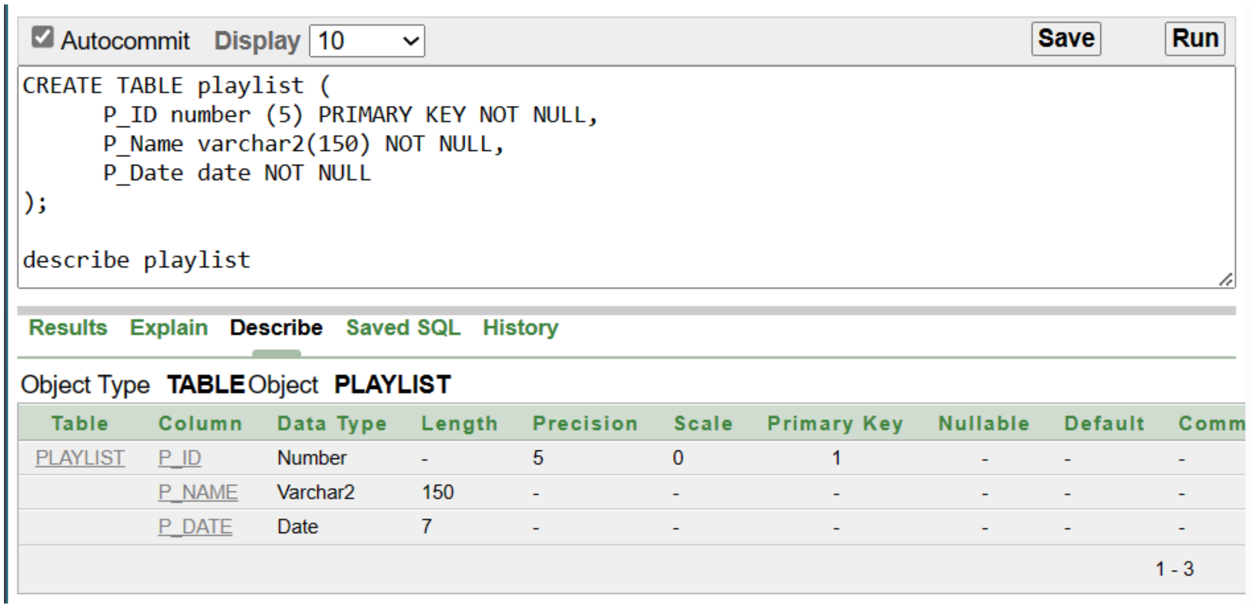
```
CREATE TABLE mluser (
    U_ID number(5) primary key NOT NULL,
    U_Name varchar2(150) NOT NULL,
    U_Email varchar2(150)
);

describe mluser
```

Object Type **TABLE** Object **MLUSER**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MLUSER	U_ID	Number	-	5	0	1	-	-	-
	U_NAME	Varchar2	150	-	-	-	-	-	-
	U_EMAIL	Varchar2	150	-	-	-	✓	-	-

1 - 3

2) playlist:



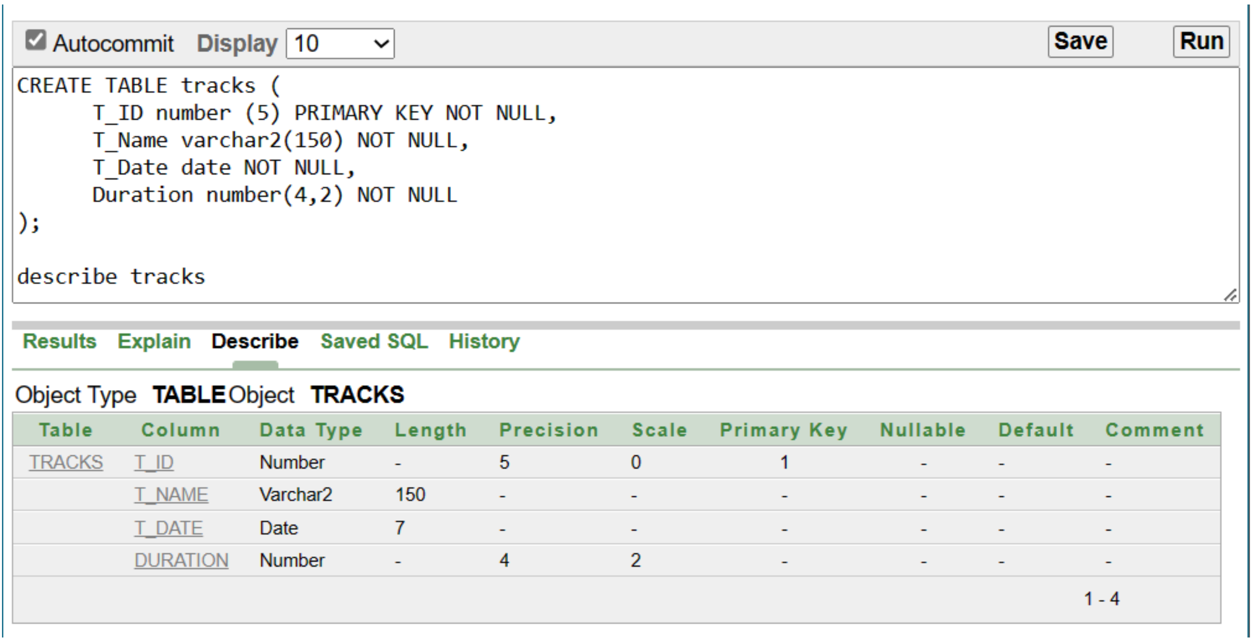
```
CREATE TABLE playlist (
      P_ID number (5) PRIMARY KEY NOT NULL,
      P_Name varchar2(150) NOT NULL,
      P_Date date NOT NULL
);

describe playlist
```

Object Type **TABLE** Object **PLAYLIST**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comm
PLAYLIST	P_ID	Number	-	5	0	1	-	-	-
	P_NAME	Varchar2	150	-	-	-	-	-	-
	P_DATE	Date	7	-	-	-	-	-	-

1 - 3

3) track:



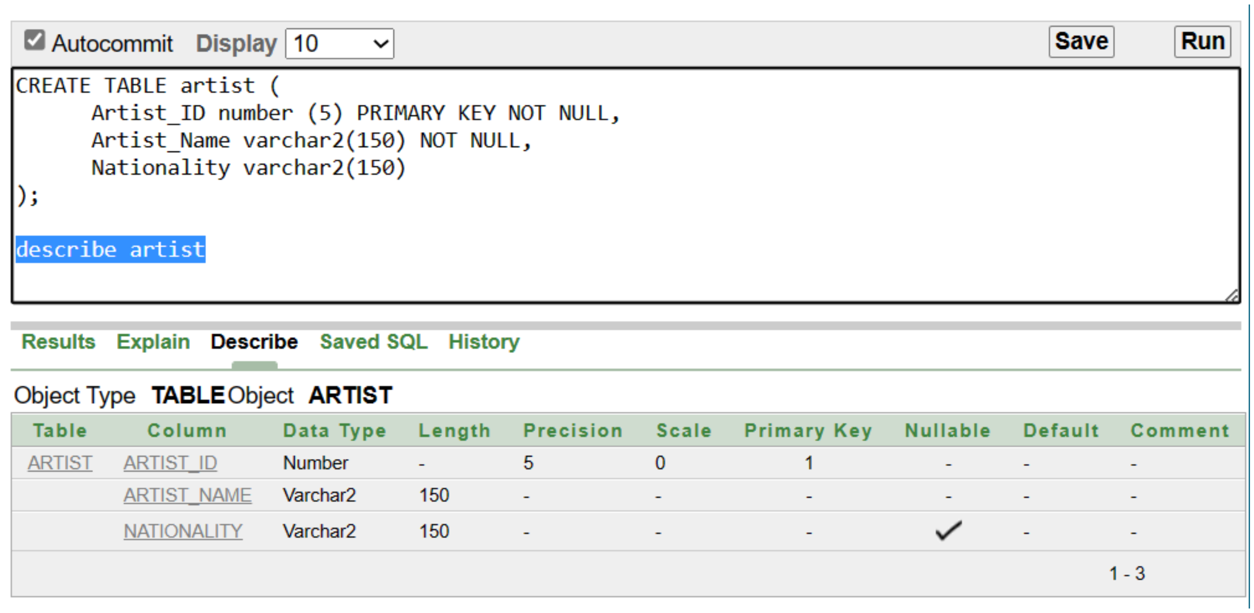
```
CREATE TABLE tracks (
      T_ID number (5) PRIMARY KEY NOT NULL,
      T_Name varchar2(150) NOT NULL,
      T_Date date NOT NULL,
      Duration number(4,2) NOT NULL
);

describe tracks
```

Object Type **TABLE** Object **TRACKS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
TRACKS	T_ID	Number	-	5	0	1	-	-	-
	T_NAME	Varchar2	150	-	-	-	-	-	-
	T_DATE	Date	7	-	-	-	-	-	-
	DURATION	Number	-	4	2	-	-	-	-

1 - 4

4) artist:



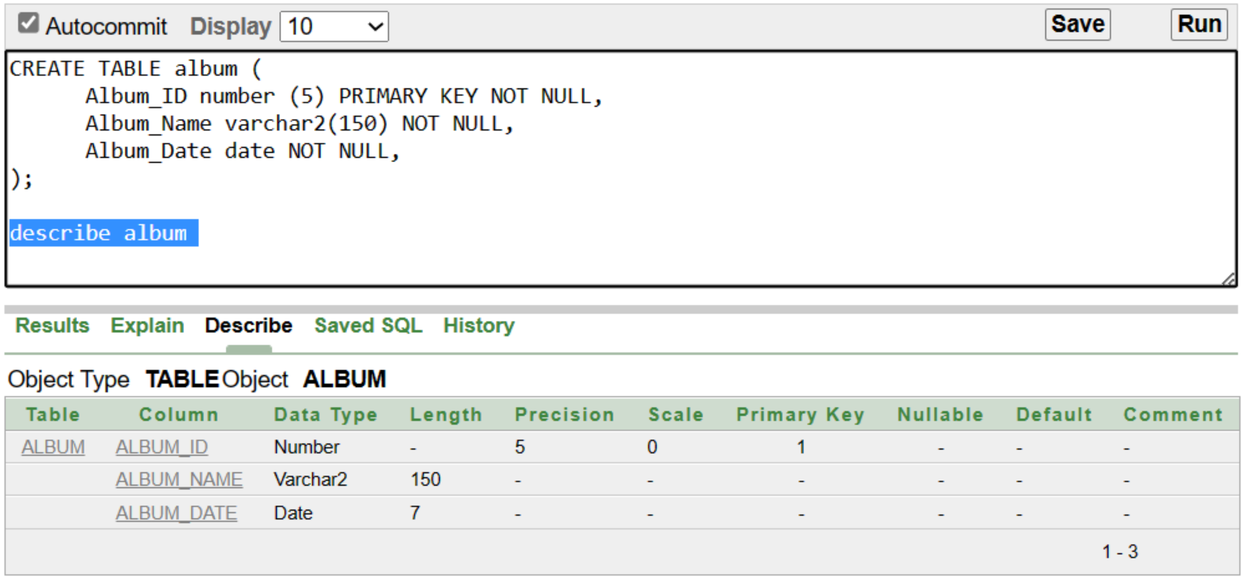
```
CREATE TABLE artist (
      Artist_ID number (5) PRIMARY KEY NOT NULL,
      Artist_Name varchar2(150) NOT NULL,
      Nationality varchar2(150)
);

describe artist
```

Object Type **TABLE** Object **ARTIST**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ARTIST	ARTIST_ID	Number	-	5	0	1	-	-	-
	ARTIST_NAME	Varchar2	150	-	-	-	-	-	-
	NATIONALITY	Varchar2	150	-	-	-	✓	-	-

1 - 3

5) album:



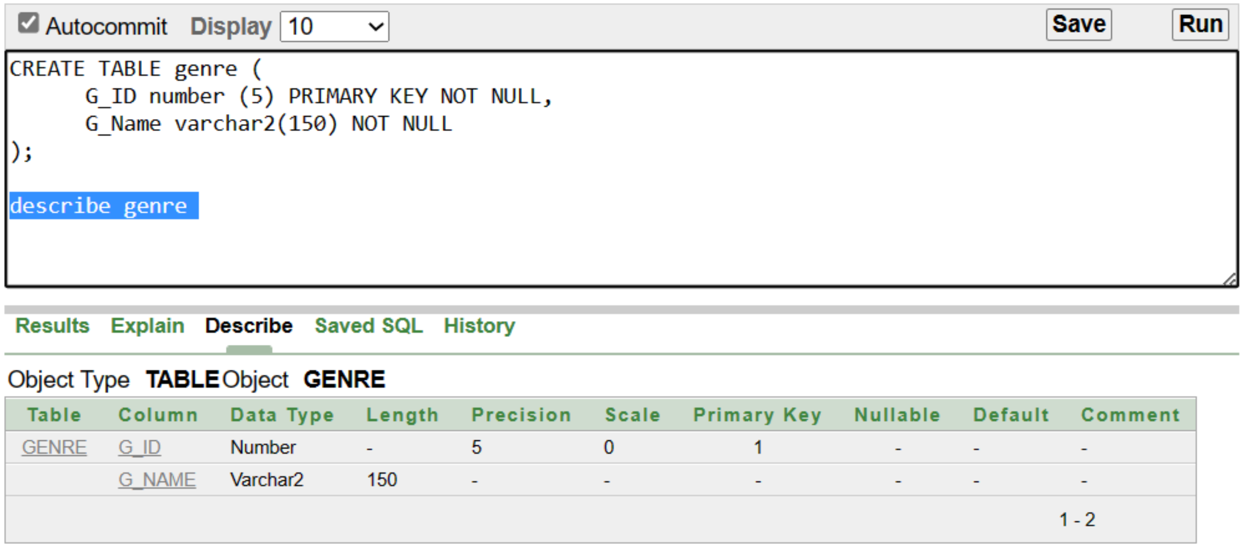
```
CREATE TABLE album (
      Album_ID number (5) PRIMARY KEY NOT NULL,
      Album_Name varchar2(150) NOT NULL,
      Album_Date date NOT NULL,
);

describe album
```

Object Type **TABLE** Object **ALBUM**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
ALBUM	ALBUM_ID	Number	-	5	0	1	-	-	-
	ALBUM_NAME	Varchar2	150	-	-	-	-	-	-
	ALBUM_DATE	Date	7	-	-	-	-	-	-

1 - 3

6) genre:



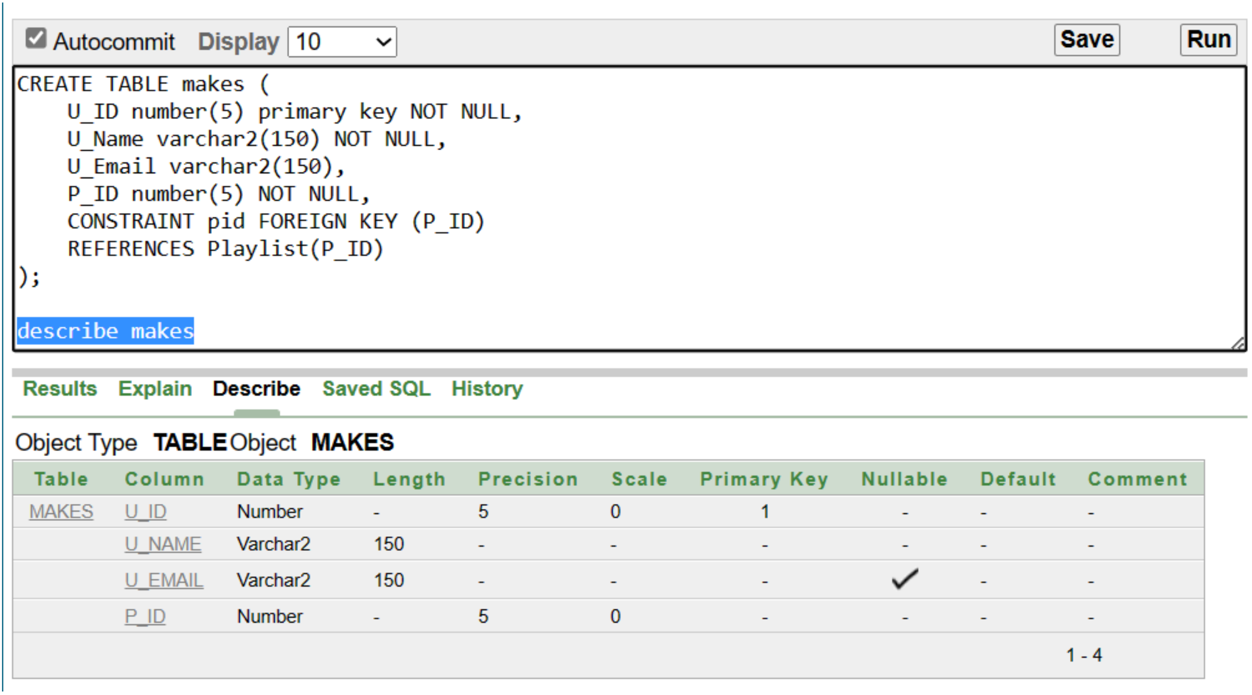
```
CREATE TABLE genre (
      G_ID number (5) PRIMARY KEY NOT NULL,
      G_Name varchar2(150) NOT NULL
);

describe genre
```

Object Type **TABLE** Object **GENRE**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
GENRE	G_ID	Number	-	5	0	1	-	-	-
	G_NAME	Varchar2	150	-	-	-	-	-	-

1 - 2

7) makes:



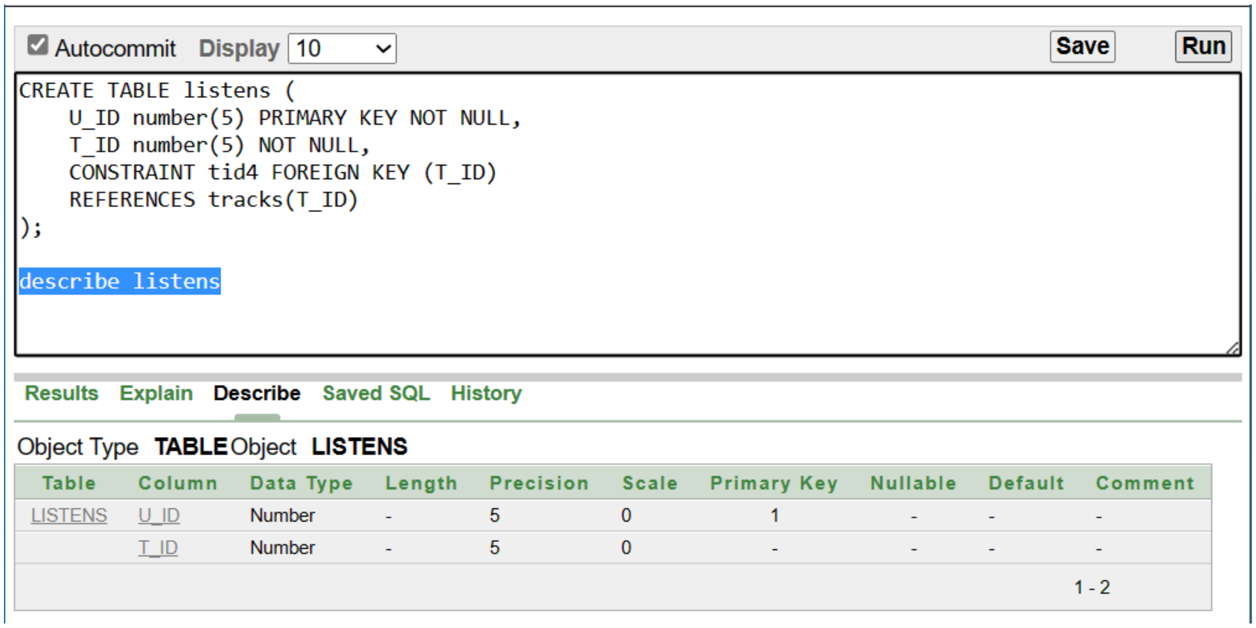
```
CREATE TABLE makes (
    U_ID number(5) primary key NOT NULL,
    U_Name varchar2(150) NOT NULL,
    U_Email varchar2(150),
    P_ID number(5) NOT NULL,
    CONSTRAINT pid FOREIGN KEY (P_ID)
    REFERENCES Playlist(P_ID)
);

describe makes
```

Object Type **TABLE** Object **MAKES**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
MAKES	U_ID	Number	-	5	0	1	-	-	-
	U_NAME	Varchar2	150	-	-	-	-	-	-
	U_EMAIL	Varchar2	150	-	-	-	✓	-	-
	P_ID	Number	-	5	0	-	-	-	-

1 - 4

8) listens:



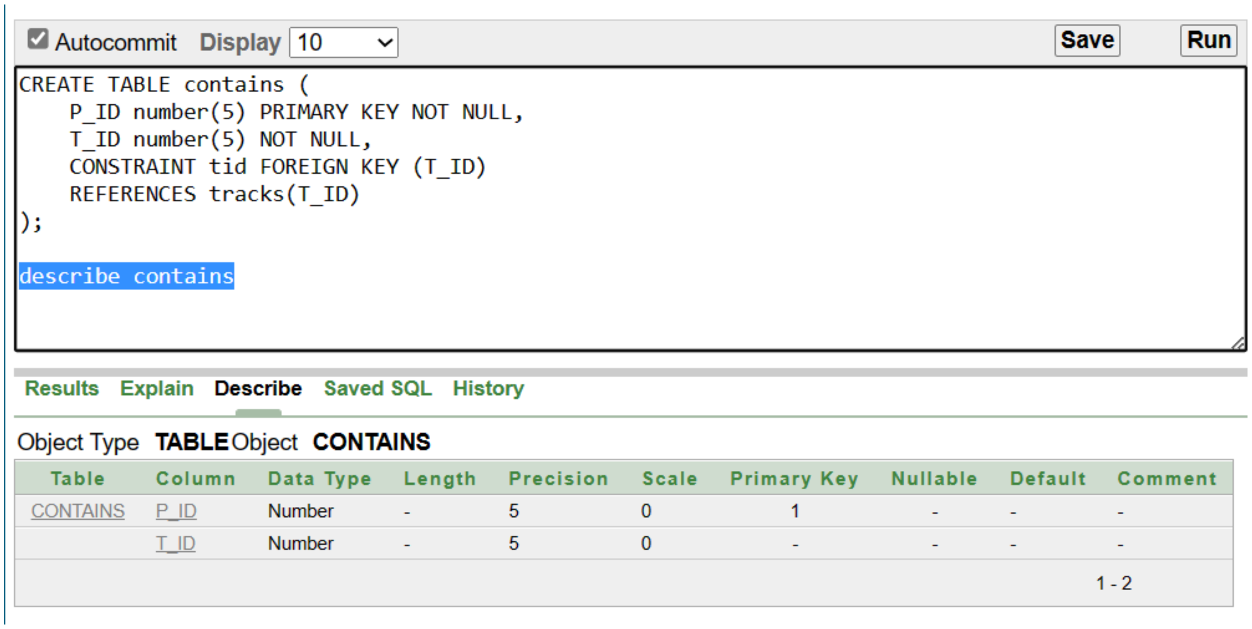
```
CREATE TABLE listens (
    U_ID number(5) PRIMARY KEY NOT NULL,
    T_ID number(5) NOT NULL,
    CONSTRAINT tid4 FOREIGN KEY (T_ID)
    REFERENCES tracks(T_ID)
);

describe listens
```

Object Type **TABLE** Object **LISTENS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
LISTENS	U_ID	Number	-	5	0	1	-	-	-
	T_ID	Number	-	5	0	-	-	-	-

1 - 2

9) contains:



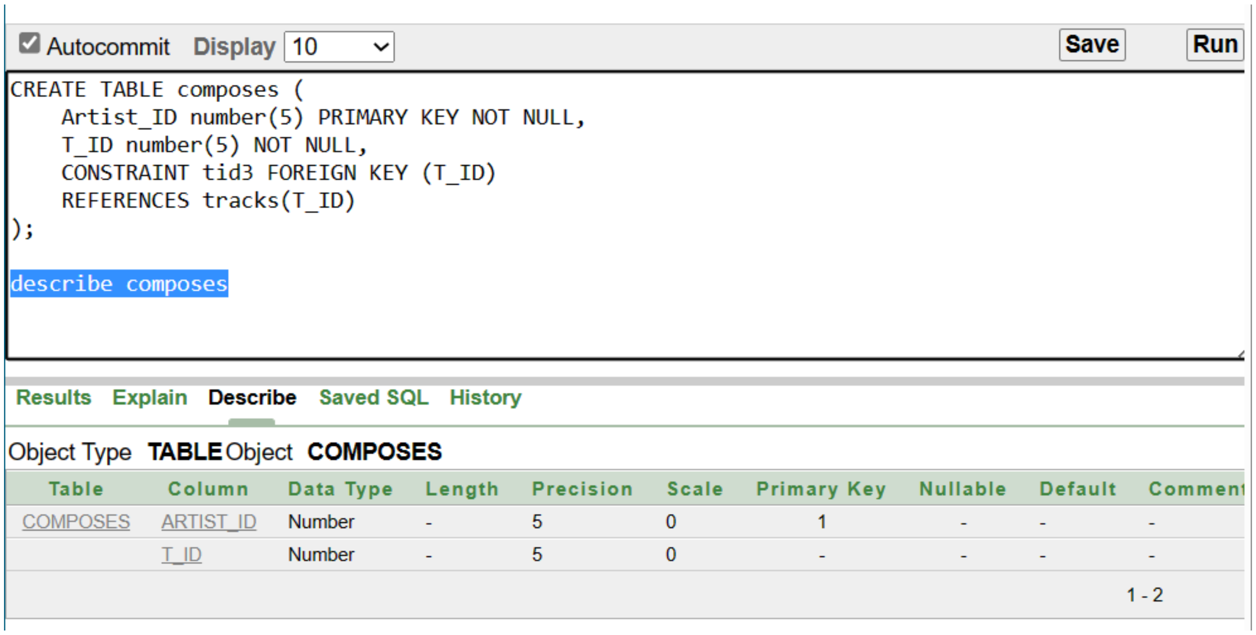
```
CREATE TABLE contains (
    P_ID number(5) PRIMARY KEY NOT NULL,
    T_ID number(5) NOT NULL,
    CONSTRAINT tid FOREIGN KEY (T_ID)
    REFERENCES tracks(T_ID)
);

describe contains
```

Object Type **TABLE** Object **CONTAINS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CONTAINS	P_ID	Number	-	5	0	1	-	-	-
	T_ID	Number	-	5	0	-	-	-	-

1 - 2

10) composes:



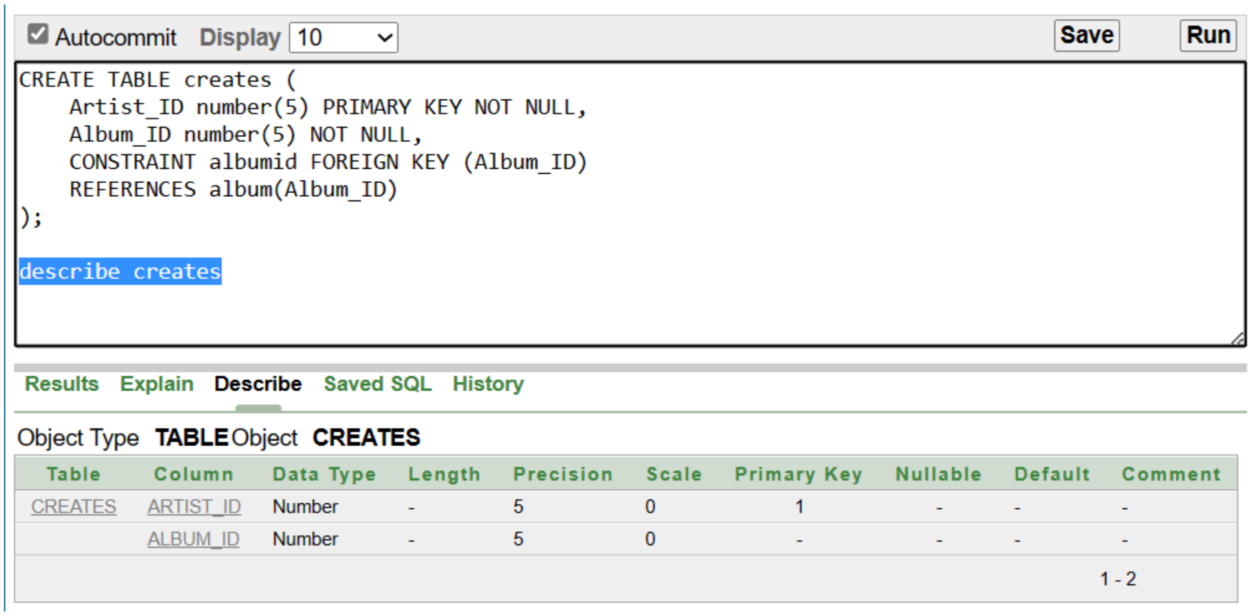
```
CREATE TABLE composes (
    Artist_ID number(5) PRIMARY KEY NOT NULL,
    T_ID number(5) NOT NULL,
    CONSTRAINT tid3 FOREIGN KEY (T_ID)
    REFERENCES tracks(T_ID)
);

describe composes
```

Object Type **TABLE** Object **COMPOSES**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
COMPOSES	ARTIST_ID	Number	-	5	0	1	-	-	-
	T_ID	Number	-	5	0	-	-	-	-

1 - 2

11) creates:



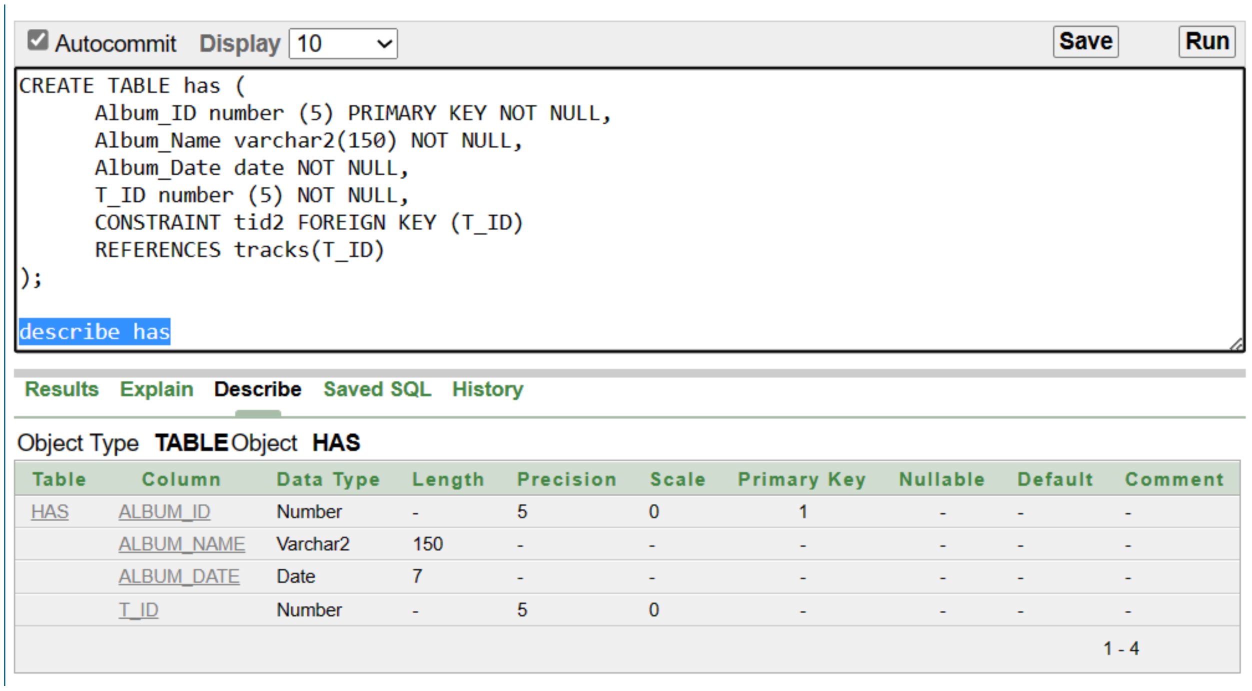
```
CREATE TABLE creates (
    Artist_ID number(5) PRIMARY KEY NOT NULL,
    Album_ID number(5) NOT NULL,
    CONSTRAINT albumid FOREIGN KEY (Album_ID)
    REFERENCES album(Album_ID)
);

describe creates
```

Object Type **TABLE** Object **CREATES**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
CREATES	ARTIST_ID	Number	-	5	0	1	-	-	-
	ALBUM_ID	Number	-	5	0	-	-	-	-

1 - 2

12) has:



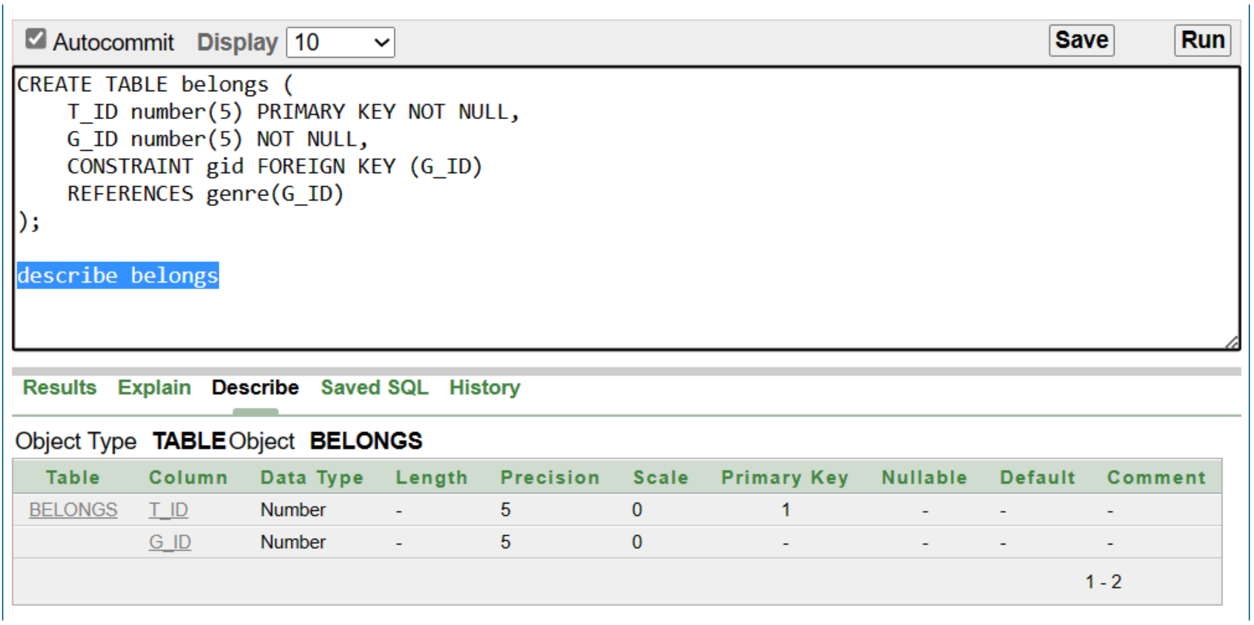
```
CREATE TABLE has (
      Album_ID number (5) PRIMARY KEY NOT NULL,
      Album_Name varchar2(150) NOT NULL,
      Album_Date date NOT NULL,
      T_ID number (5) NOT NULL,
      CONSTRAINT tid2 FOREIGN KEY (T_ID)
      REFERENCES tracks(T_ID)
);

describe has
```

Object Type **TABLE** Object **HAS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
HAS	ALBUM_ID	Number	-	5	0	1	-	-	-
	ALBUM_NAME	Varchar2	150	-	-	-	-	-	-
	ALBUM_DATE	Date	7	-	-	-	-	-	-
	T_ID	Number	-	5	0	-	-	-	-

1 - 4

13) belongs:



Autocommit Display 10 Save Run

```
CREATE TABLE belongs (
    T_ID number(5) PRIMARY KEY NOT NULL,
    G_ID number(5) NOT NULL,
    CONSTRAINT gid FOREIGN KEY (G_ID)
    REFERENCES genre(G_ID)
);

describe belongs
```

Results Explain Describe Saved SQL History

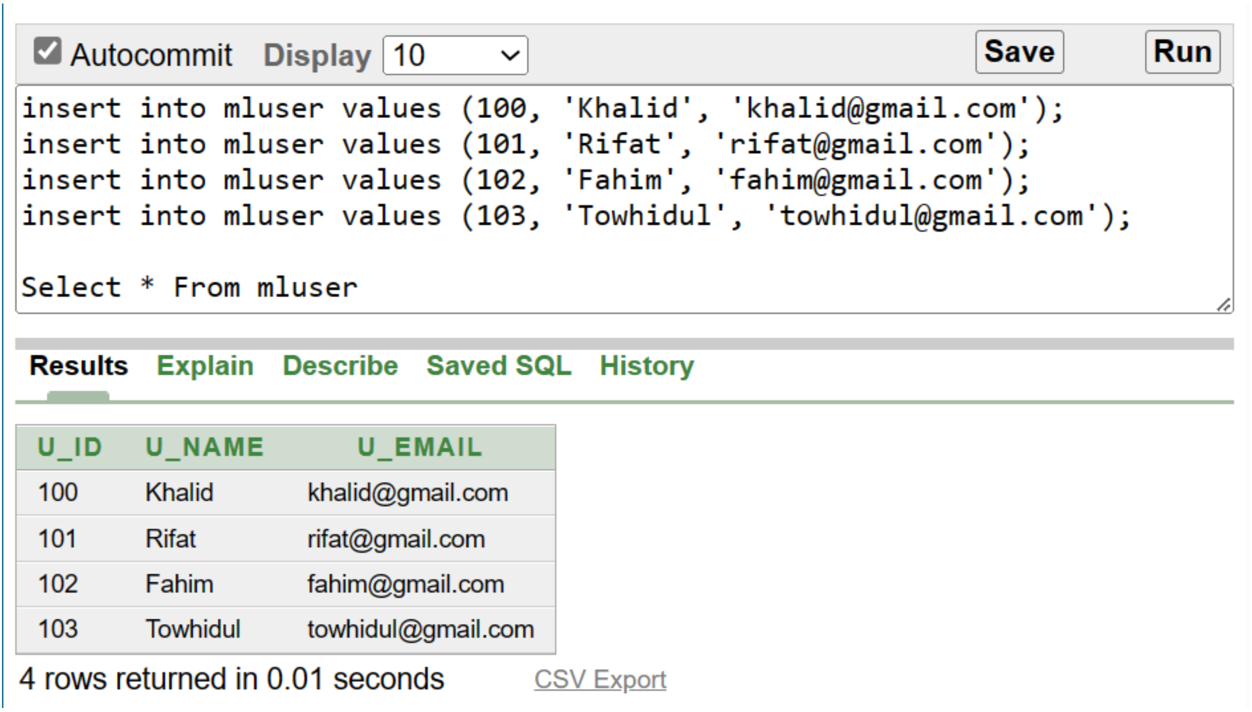
Object Type **TABLE** Object **BELONGS**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
BELONGS	T_ID	Number	-	5	0	1	-	-	-
	G_ID	Number	-	5	0	-	-	-	-

1 - 2

MUSIC LIBRARY MANAGEMENT SYSTEM DATA INSERTION

1) mluser:



Autocommit Display 10 Save Run

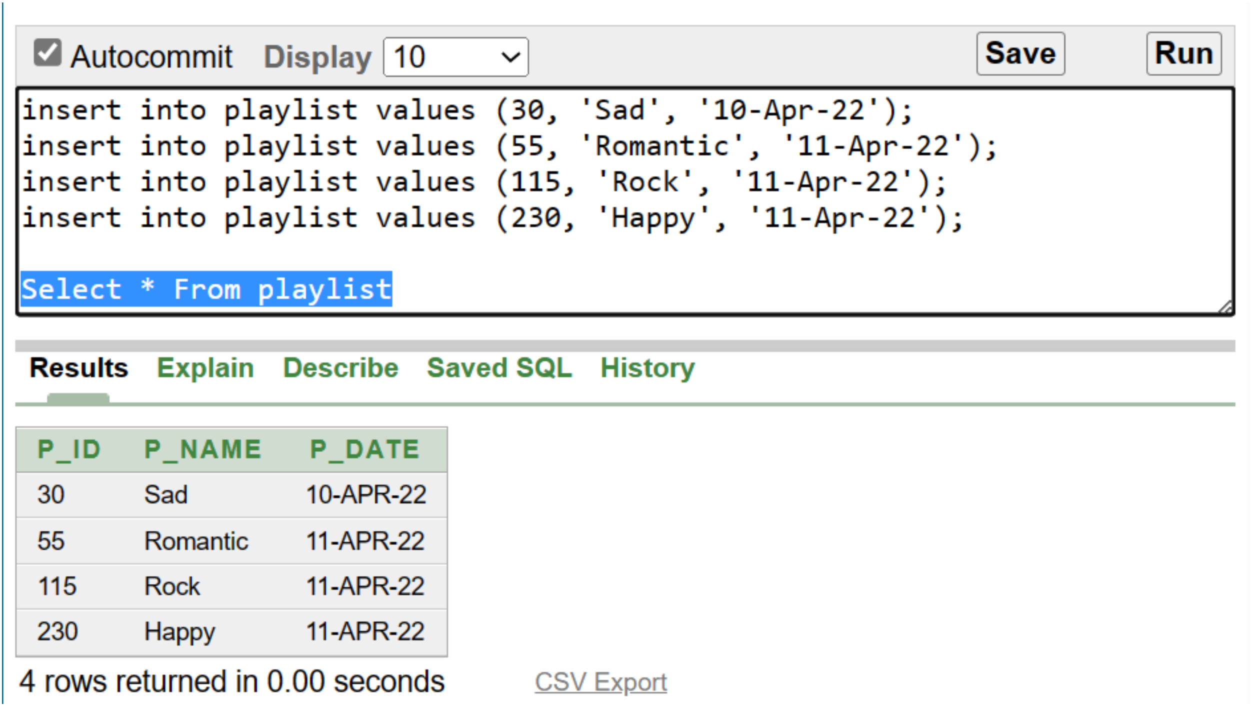
```
insert into mluser values (100, 'Khalid', 'khalid@gmail.com');
insert into mluser values (101, 'Rifat', 'rifat@gmail.com');
insert into mluser values (102, 'Fahim', 'fahim@gmail.com');
insert into mluser values (103, 'Towhidul', 'towhidul@gmail.com');

Select * From mluser
```

Results Explain Describe Saved SQL History

U_ID	U_NAME	U_EMAIL
100	Khalid	khalid@gmail.com
101	Rifat	rifat@gmail.com
102	Fahim	fahim@gmail.com
103	Towhidul	towhidul@gmail.com

4 rows returned in 0.01 seconds CSV Export

2) playlist:

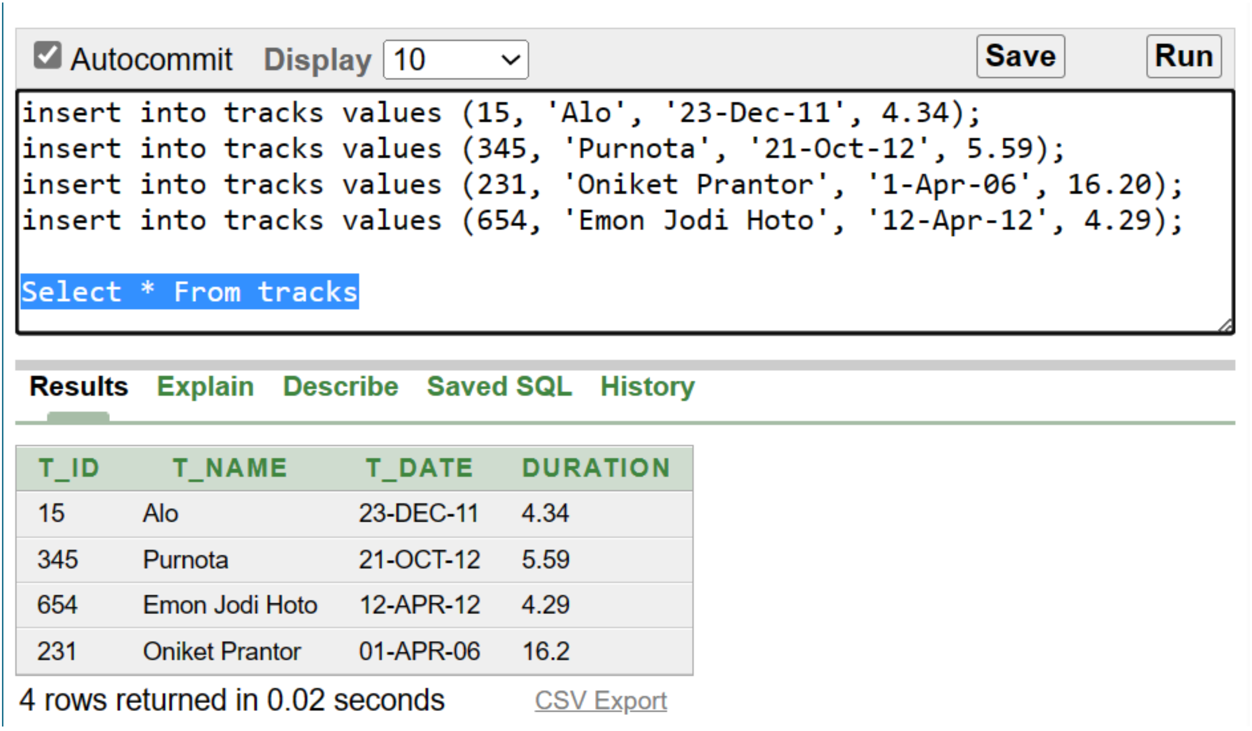


```
insert into playlist values (30, 'Sad', '10-Apr-22');
insert into playlist values (55, 'Romantic', '11-Apr-22');
insert into playlist values (115, 'Rock', '11-Apr-22');
insert into playlist values (230, 'Happy', '11-Apr-22');

Select * From playlist
```

P_ID	P_NAME	P_DATE
30	Sad	10-APR-22
55	Romantic	11-APR-22
115	Rock	11-APR-22
230	Happy	11-APR-22

4 rows returned in 0.00 seconds

3) tracks:

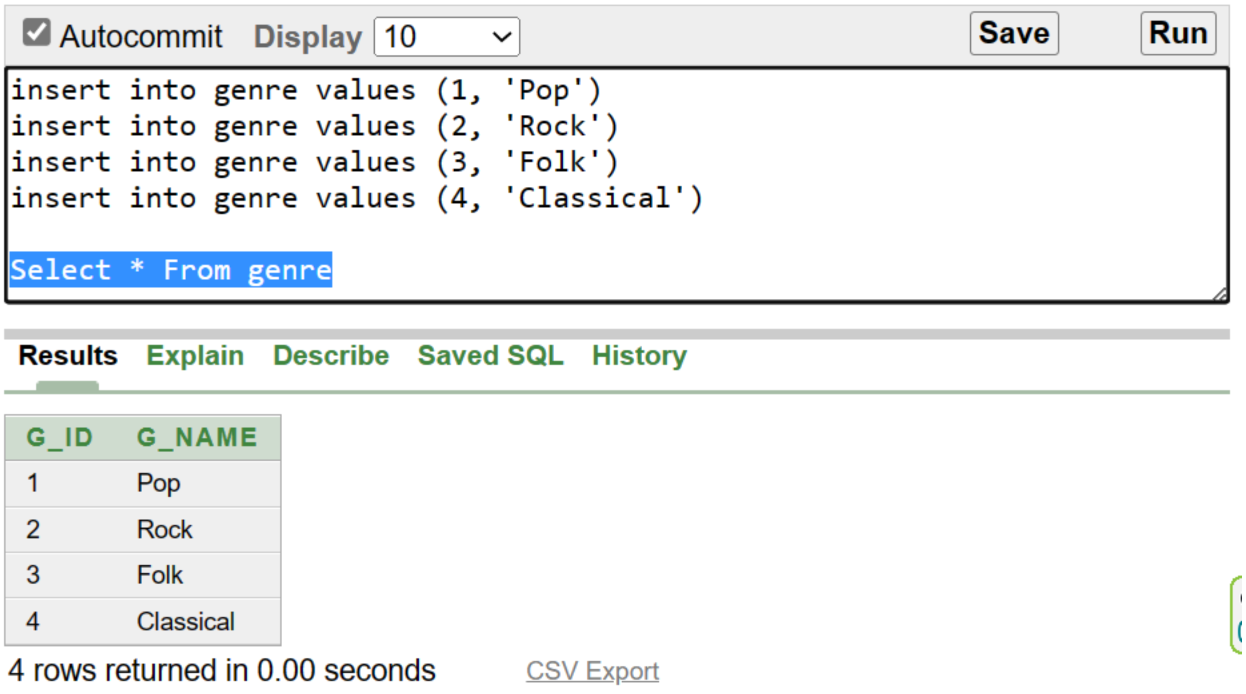


```
insert into tracks values (15, 'Alo', '23-Dec-11', 4.34);
insert into tracks values (345, 'Purnota', '21-Oct-12', 5.59);
insert into tracks values (231, 'Oniket Prantor', '1-Apr-06', 16.20);
insert into tracks values (654, 'Emon Jodi Hoto', '12-Apr-12', 4.29);

Select * From tracks
```

T_ID	T_NAME	T_DATE	DURATION
15	Alo	23-DEC-11	4.34
345	Purnota	21-OCT-12	5.59
654	Emon Jodi Hoto	12-APR-12	4.29
231	Oniket Prantor	01-APR-06	16.2

4 rows returned in 0.02 seconds

4) genre:

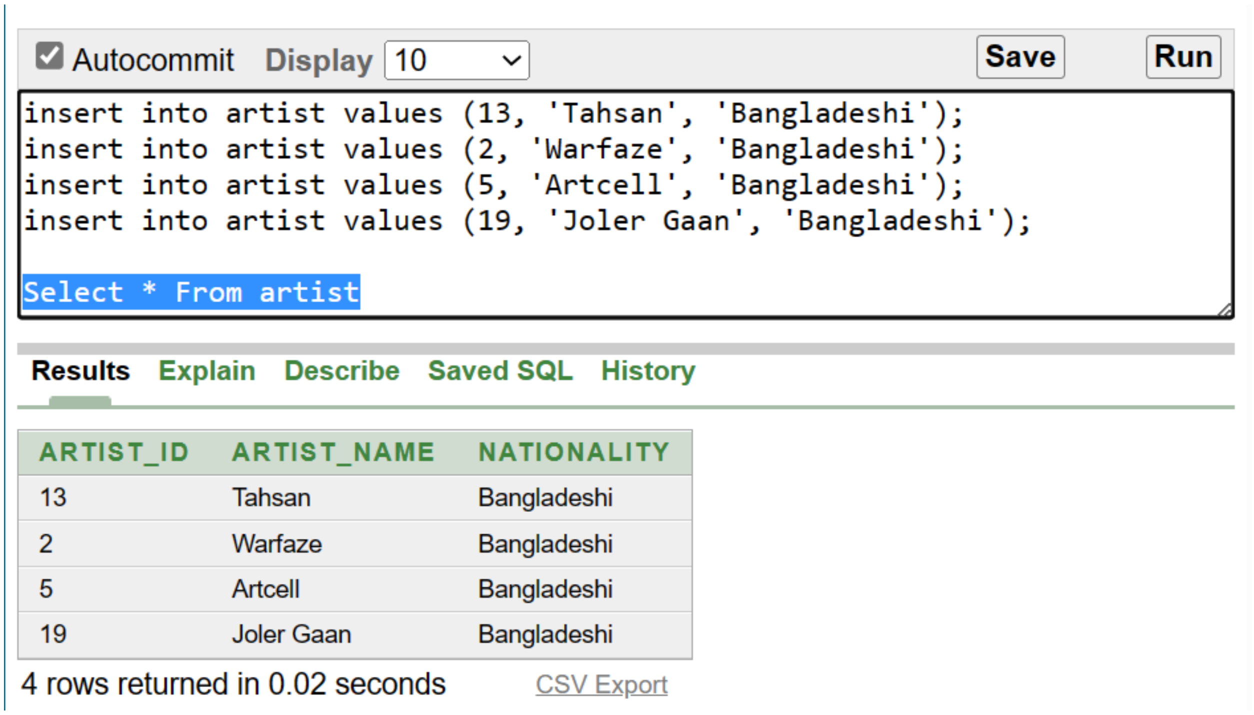


```
insert into genre values (1, 'Pop')
insert into genre values (2, 'Rock')
insert into genre values (3, 'Folk')
insert into genre values (4, 'Classical')

Select * From genre
```

G_ID	G_NAME
1	Pop
2	Rock
3	Folk
4	Classical

4 rows returned in 0.00 seconds

5) artist:

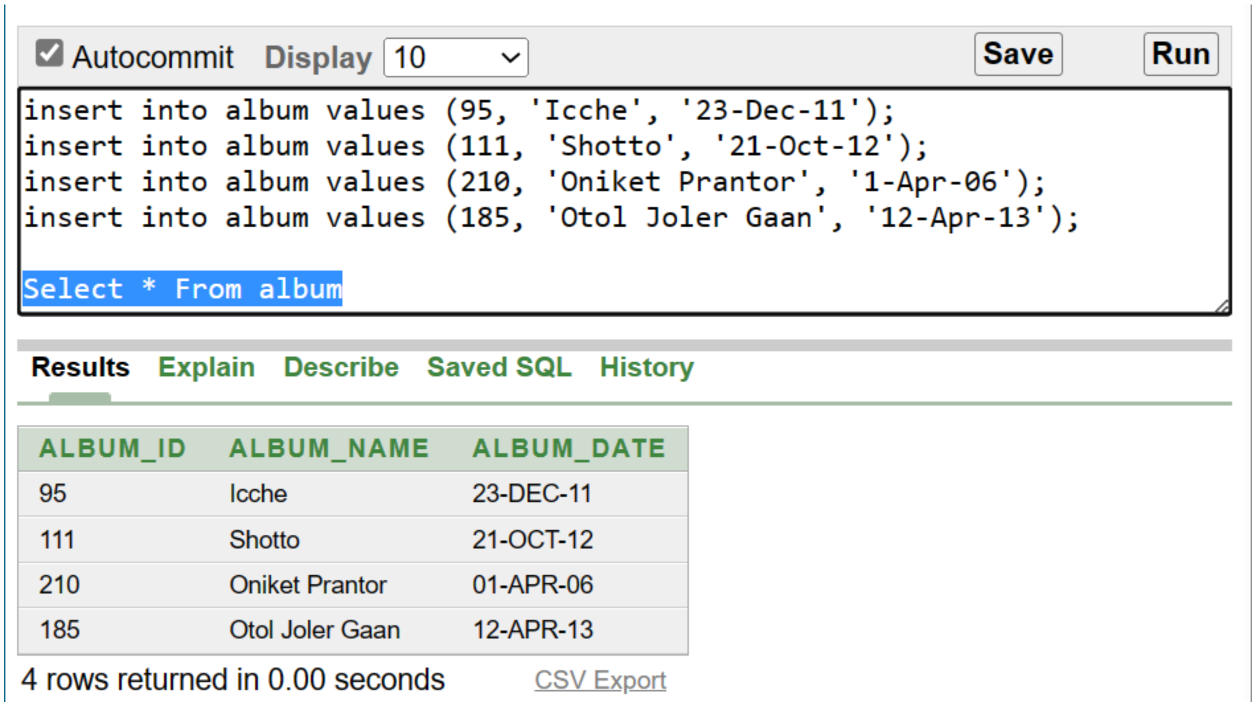


```
insert into artist values (13, 'Tahsan', 'Bangladeshi');
insert into artist values (2, 'Warfaze', 'Bangladeshi');
insert into artist values (5, 'Artcell', 'Bangladeshi');
insert into artist values (19, 'Joler Gaan', 'Bangladeshi');

Select * From artist
```

ARTIST_ID	ARTIST_NAME	NATIONALITY
13	Tahsan	Bangladeshi
2	Warfaze	Bangladeshi
5	Artcell	Bangladeshi
19	Joler Gaan	Bangladeshi

4 rows returned in 0.02 seconds

6) album:

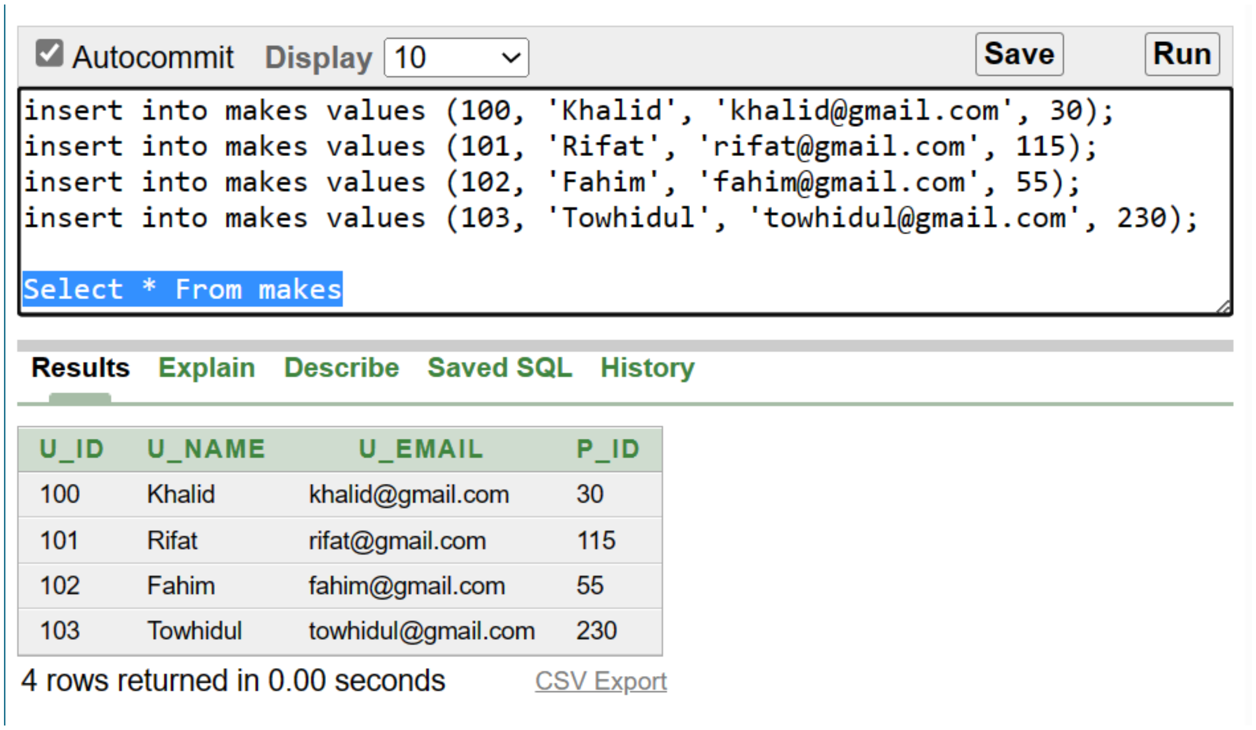


```
insert into album values (95, 'Icche', '23-Dec-11');
insert into album values (111, 'Shotto', '21-Oct-12');
insert into album values (210, 'Oniket Prantor', '1-Apr-06');
insert into album values (185, 'Otol Joler Gaan', '12-Apr-13');

Select * From album
```

ALBUM_ID	ALBUM_NAME	ALBUM_DATE
95	Icche	23-DEC-11
111	Shotto	21-OCT-12
210	Oniket Prantor	01-APR-06
185	Otol Joler Gaan	12-APR-13

4 rows returned in 0.00 seconds

7) makes:

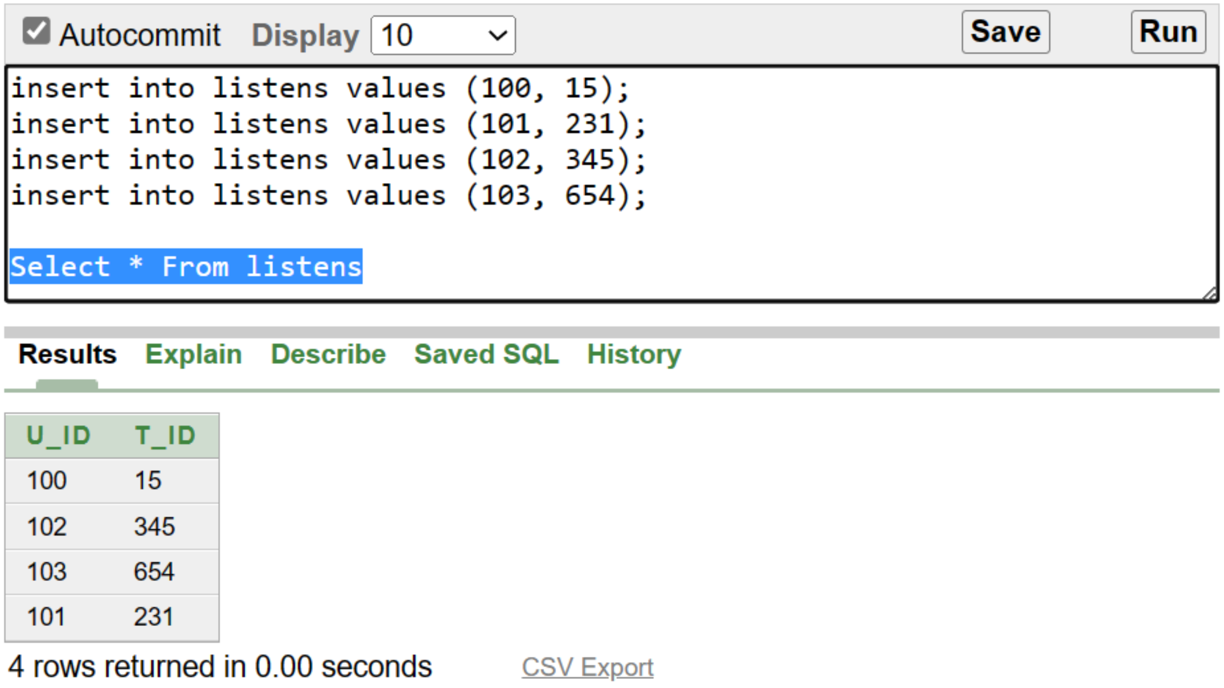


```
insert into makes values (100, 'Khalid', 'khalid@gmail.com', 30);
insert into makes values (101, 'Rifat', 'rifat@gmail.com', 115);
insert into makes values (102, 'Fahim', 'fahim@gmail.com', 55);
insert into makes values (103, 'Towhidul', 'towhidul@gmail.com', 230);

Select * From makes
```

U_ID	U_NAME	U_EMAIL	P_ID
100	Khalid	khalid@gmail.com	30
101	Rifat	rifat@gmail.com	115
102	Fahim	fahim@gmail.com	55
103	Towhidul	towhidul@gmail.com	230

4 rows returned in 0.00 seconds

8) listens:



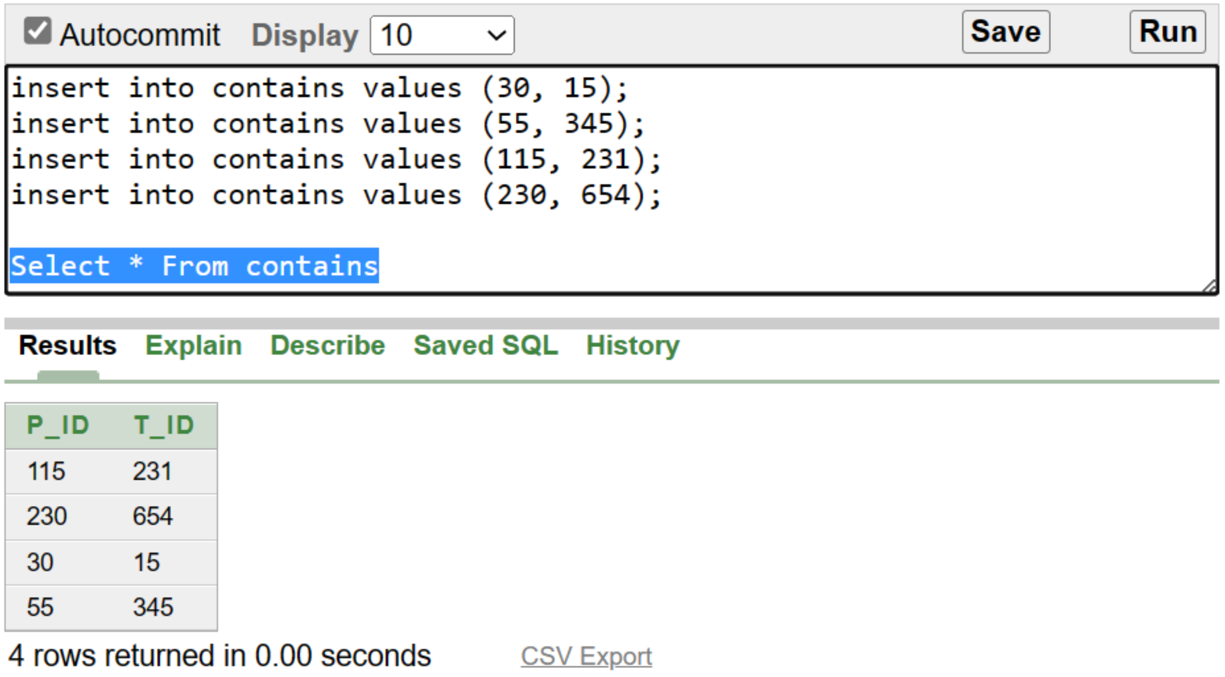
Display: 10

```
insert into listens values (100, 15);
insert into listens values (101, 231);
insert into listens values (102, 345);
insert into listens values (103, 654);

Select * From listens
```

U_ID	T_ID
100	15
102	345
103	654
101	231

4 rows returned in 0.00 seconds

9) contains:



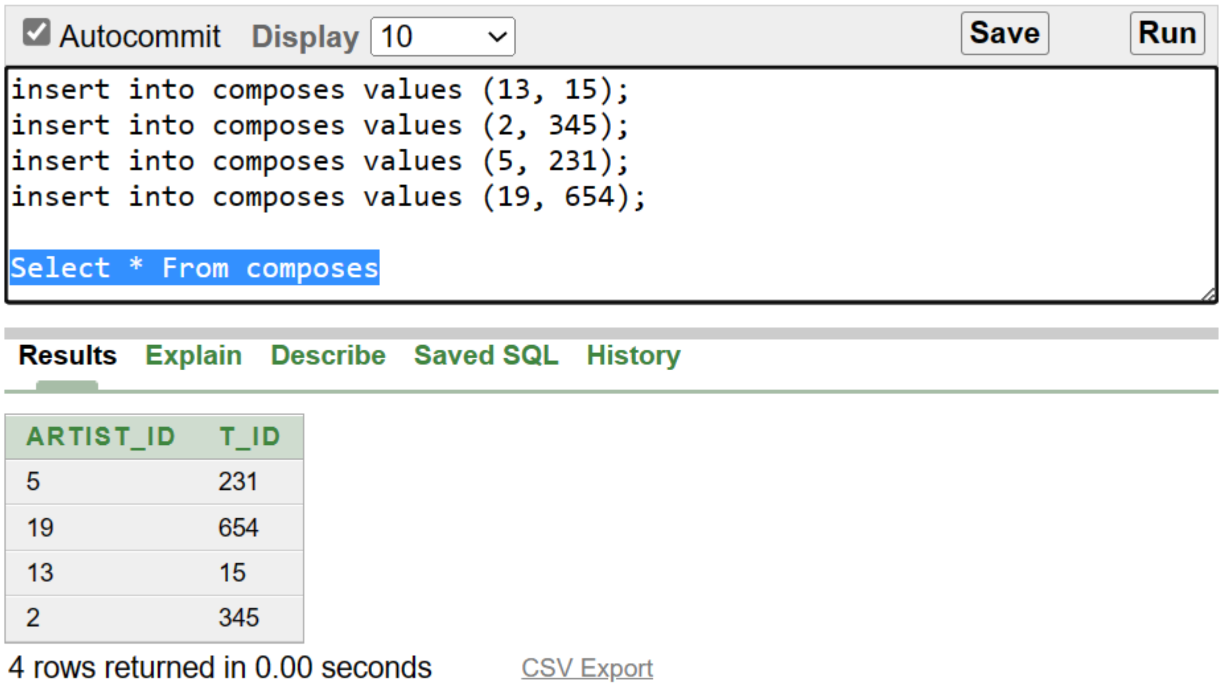
Display: 10

```
insert into contains values (30, 15);
insert into contains values (55, 345);
insert into contains values (115, 231);
insert into contains values (230, 654);

Select * From contains
```

P_ID	T_ID
115	231
230	654
30	15
55	345

4 rows returned in 0.00 seconds

10) composes:

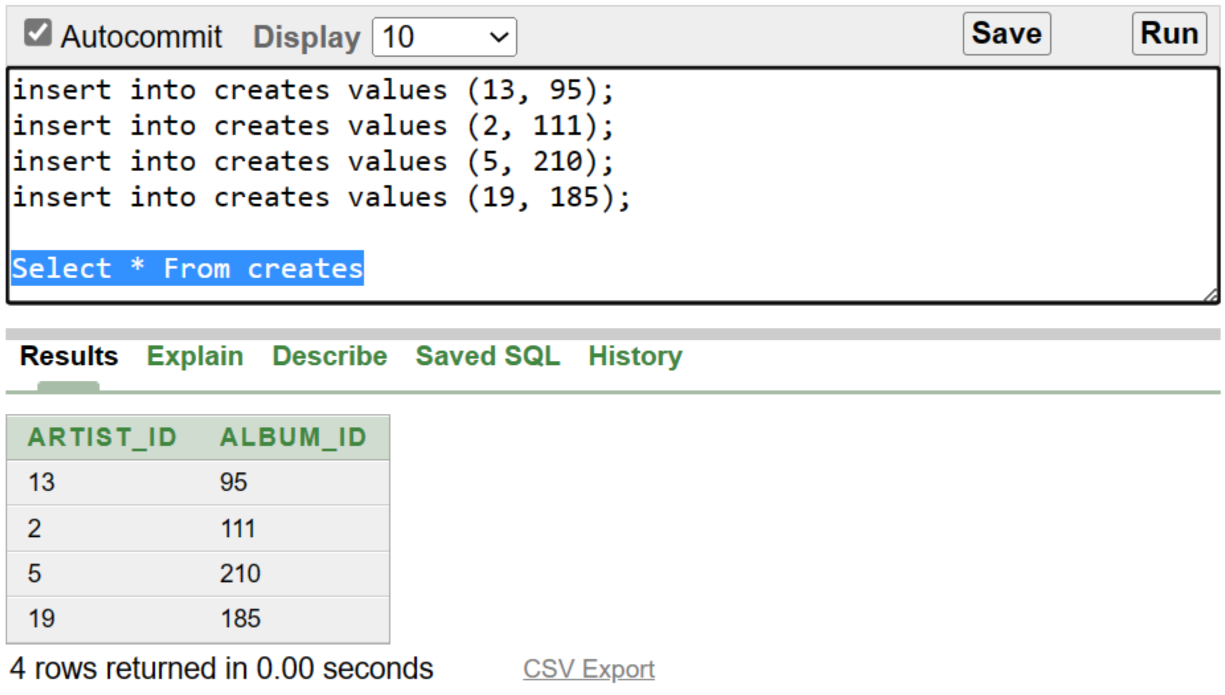


```
insert into composes values (13, 15);
insert into composes values (2, 345);
insert into composes values (5, 231);
insert into composes values (19, 654);

Select * From composes
```

ARTIST_ID	T_ID
5	231
19	654
13	15
2	345

4 rows returned in 0.00 seconds

11) creates:

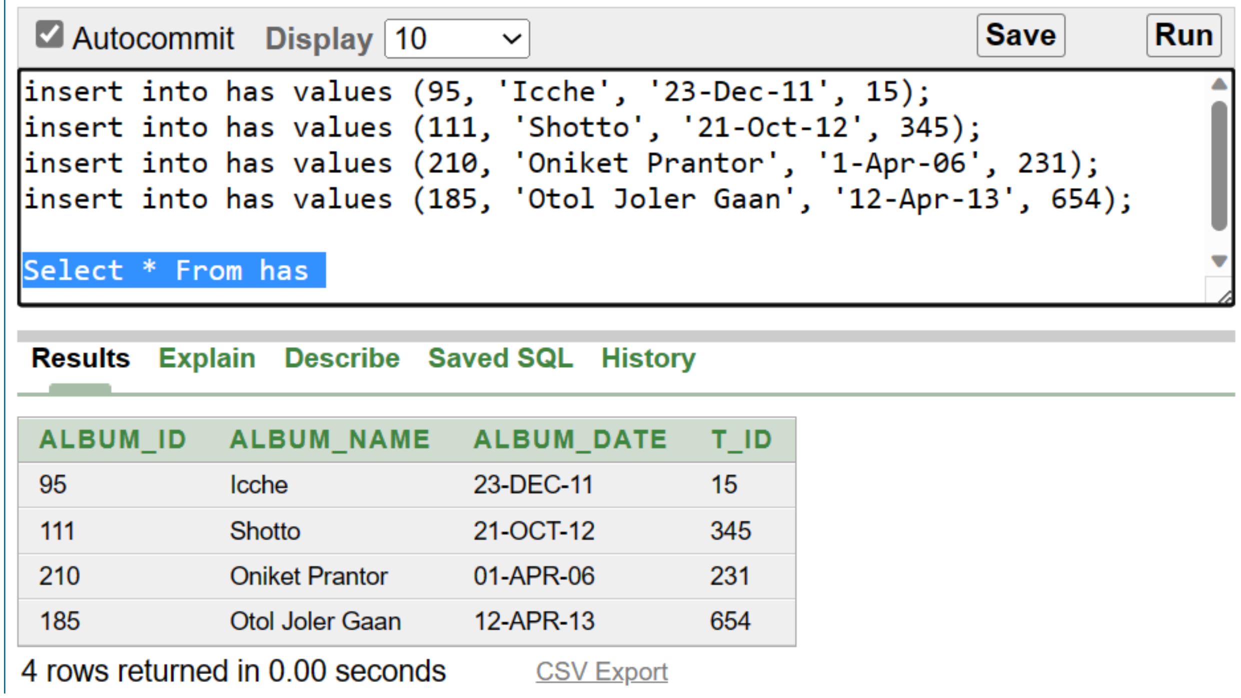


```
insert into creates values (13, 95);
insert into creates values (2, 111);
insert into creates values (5, 210);
insert into creates values (19, 185);

Select * From creates
```

ARTIST_ID	ALBUM_ID
13	95
2	111
5	210
19	185

4 rows returned in 0.00 seconds

12) has:

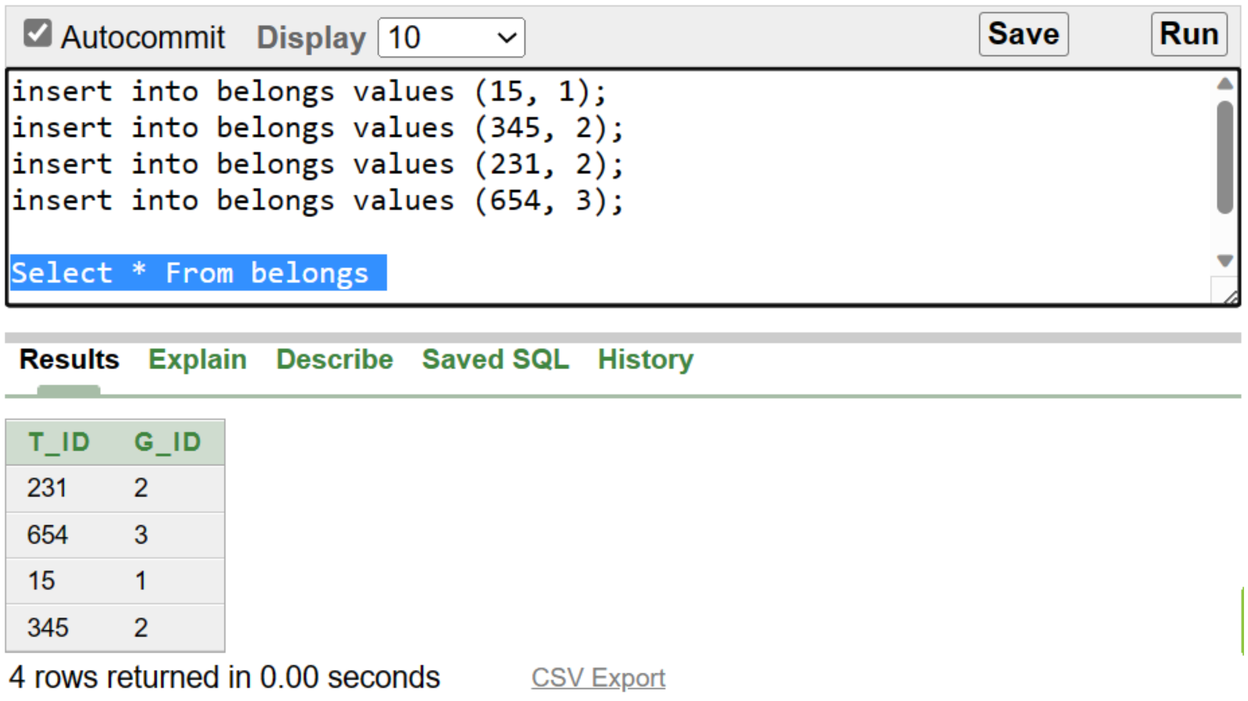


```
insert into has values (95, 'Icche', '23-Dec-11', 15);
insert into has values (111, 'Shotto', '21-Oct-12', 345);
insert into has values (210, 'Oniket Prantor', '1-Apr-06', 231);
insert into has values (185, 'Otol Joler Gaan', '12-Apr-13', 654);

Select * From has
```

ALBUM_ID	ALBUM_NAME	ALBUM_DATE	T_ID
95	Icche	23-DEC-11	15
111	Shotto	21-OCT-12	345
210	Oniket Prantor	01-APR-06	231
185	Otol Joler Gaan	12-APR-13	654

4 rows returned in 0.00 seconds

13) belongs:



```
insert into belongs values (15, 1);
insert into belongs values (345, 2);
insert into belongs values (231, 2);
insert into belongs values (654, 3);

Select * From belongs
```

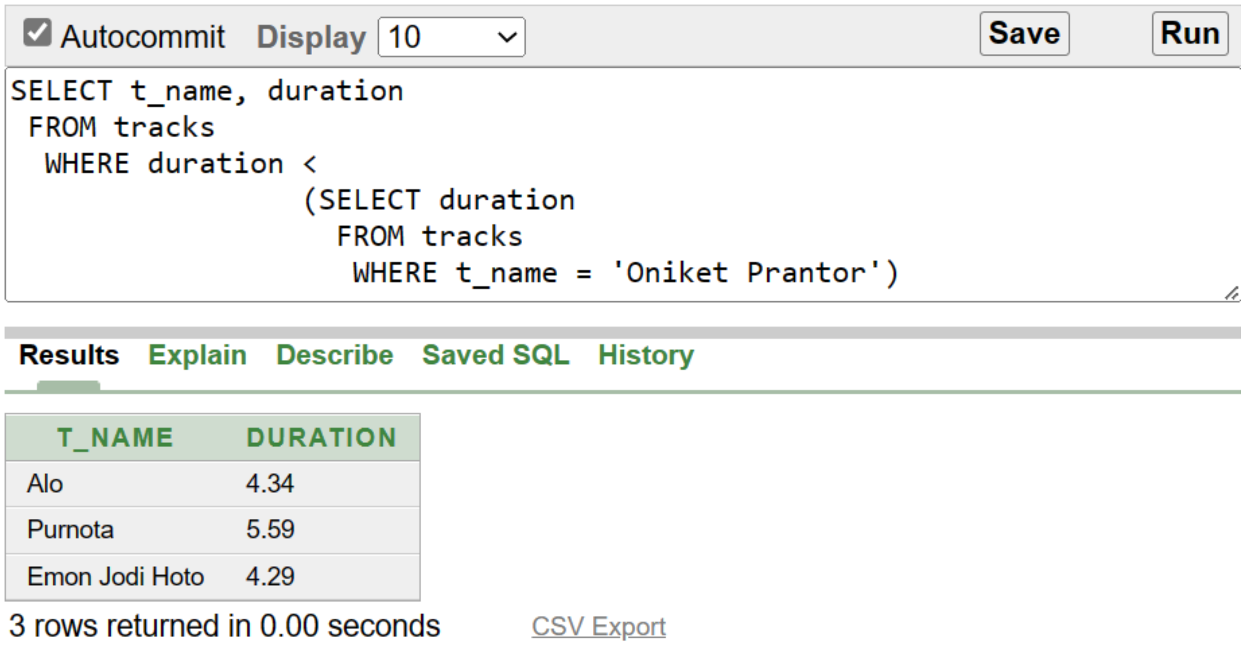
T_ID	G_ID
231	2
654	3
15	1
345	2

4 rows returned in 0.00 seconds

MUSIC LIBRARY MANAGEMENT SYSTEM QUERIES

Single Row Subquery:

1) Show the track names with duration less than that of the track, 'Oniket Prantor'

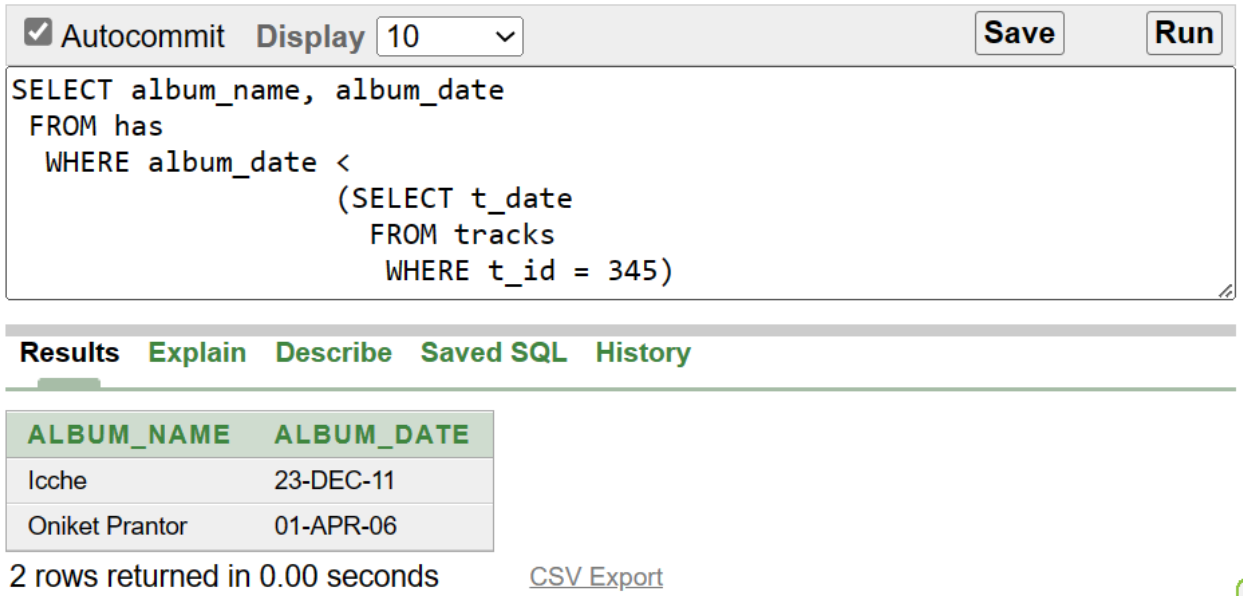


```
SELECT t_name, duration
 FROM tracks
  WHERE duration <
                (SELECT duration
                  FROM tracks
                   WHERE t_name = 'Oniket Prantor')
```

T_NAME	DURATION
Alo	4.34
Purnota	5.59
Emon Jodi Hoto	4.29

3 rows returned in 0.00 seconds

2) Show the album names and creation date that was created before t_id 345



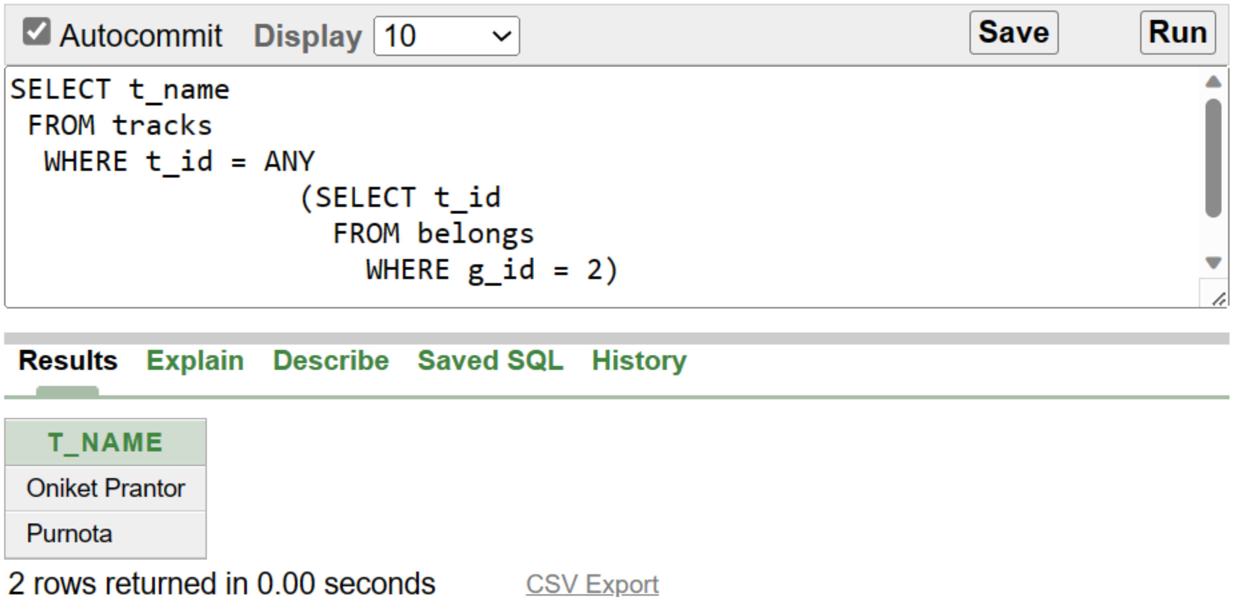
```
SELECT album_name, album_date
 FROM has
  WHERE album_date <
                (SELECT t_date
                  FROM tracks
                   WHERE t_id = 345)
```

ALBUM_NAME	ALBUM_DATE
Icche	23-DEC-11
Oniket Prantor	01-APR-06

2 rows returned in 0.00 seconds

Multiple Row Subquery:

3) Show the track names that belongs to the same genre as g_id 2

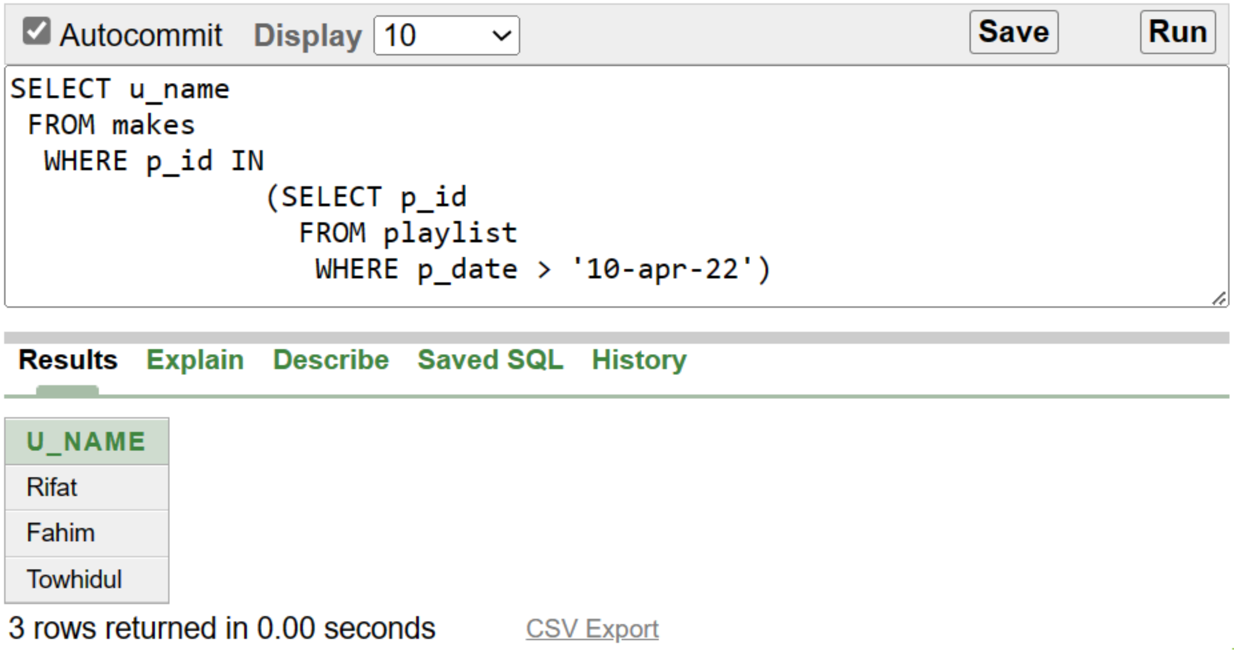


```
SELECT t_name
 FROM tracks
  WHERE t_id = ANY
               (SELECT t_id
                 FROM belongs
                   WHERE g_id = 2)
```

T_NAME
Oniket Prantor
Purnota

2 rows returned in 0.00 seconds

4) Show the user names who created a playlist after 10-apr-22



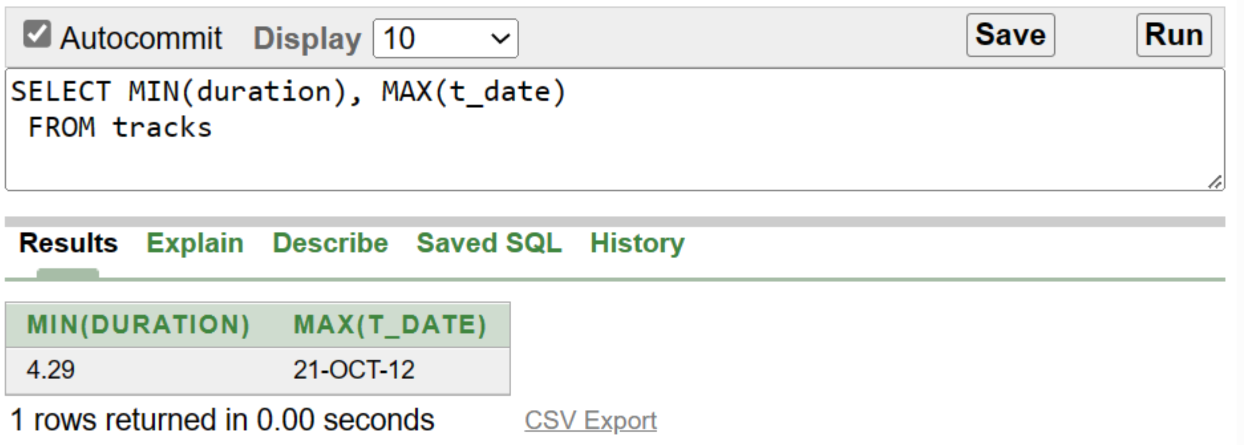
```
SELECT u_name
 FROM makes
  WHERE p_id IN
             (SELECT p_id
               FROM playlist
                WHERE p_date > '10-apr-22')
```

U_NAME
Rifat
Fahim
Towhidul

3 rows returned in 0.00 seconds

Aggregate Function:

5) Show the minimum duration and the date of the last created track



```
SELECT MIN(duration), MAX(t_date)
 FROM tracks
```

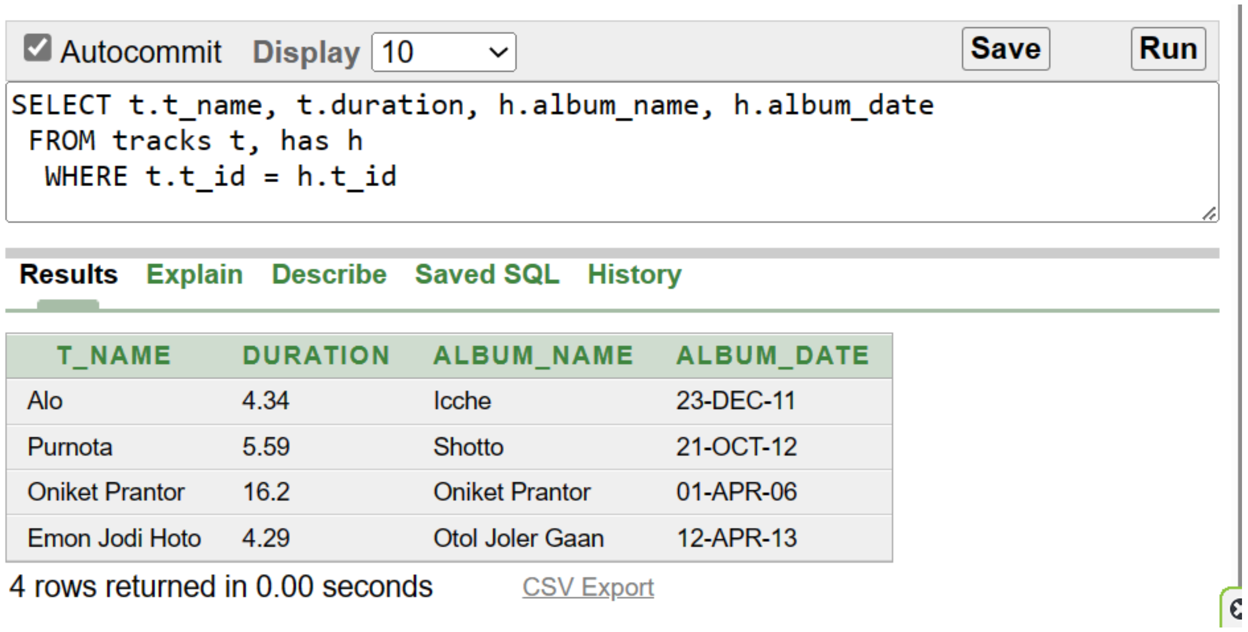
MIN(DURATION)	MAX(T_DATE)
4.29	21-OCT-12

1 rows returned in 0.00 seconds

Joining:

Equijoin:

6) Show the track name, track duration, album name, album creation date with proper joining condition



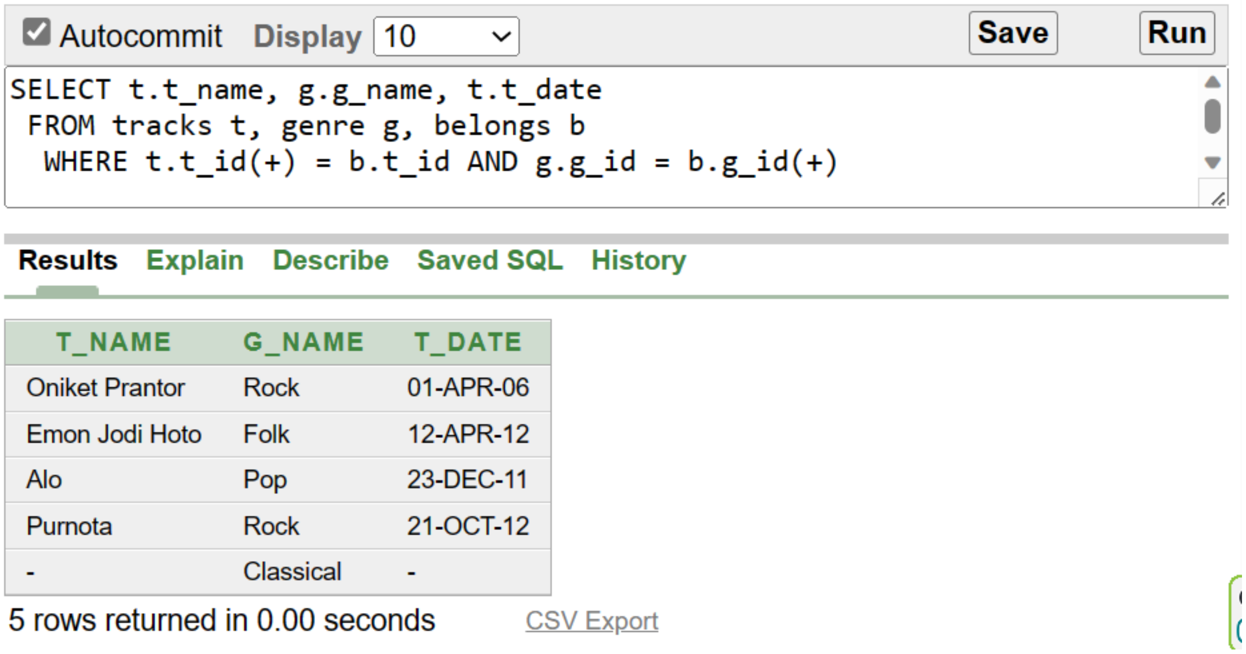
```
SELECT t.t_name, t.duration, h.album_name, h.album_date
 FROM tracks t, has h
  WHERE t.t_id = h.t_id
```

T_NAME	DURATION	ALBUM_NAME	ALBUM_DATE
Alo	4.34	Icche	23-DEC-11
Purnota	5.59	Shotto	21-OCT-12
Oniket Prantor	16.2	Oniket Prantor	01-APR-06
Emon Jodi Hoto	4.29	Otol Joler Gaan	12-APR-13

4 rows returned in 0.00 seconds

Outer Join:

7) Show the track name, track creation date, genre name with joining condition



```
SELECT t.t_name, g.g_name, t.t_date
 FROM tracks t, genre g, belongs b
  WHERE t.t_id(+) = b.t_id AND g.g_id = b.g_id(+)
```

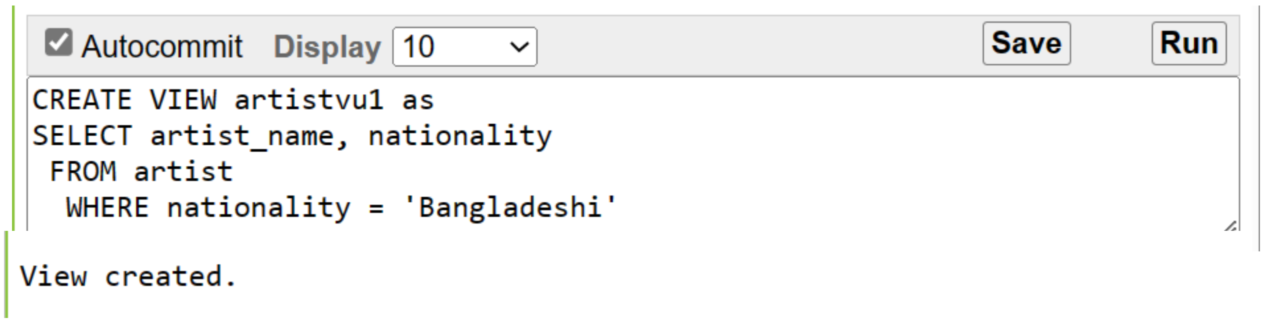
T_NAME	G_NAME	T_DATE
Oniket Prantor	Rock	01-APR-06
Emon Jodi Hoto	Folk	12-APR-12
Alo	Pop	23-DEC-11
Purnota	Rock	21-OCT-12
-	Classical	-

5 rows returned in 0.00 seconds

View Creation:

Simple View:

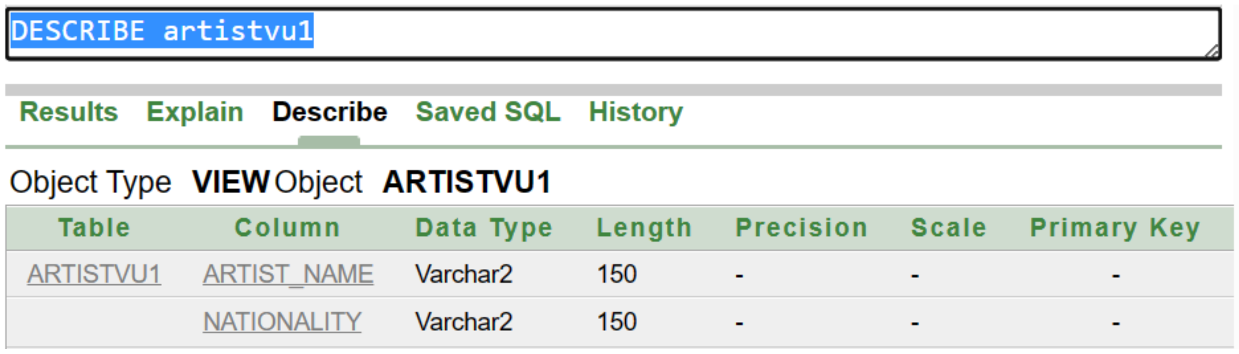
- Create a view to show the artist names and nationality who are Bangladeshi



```
CREATE VIEW artistvu1 as
SELECT artist_name, nationality
 FROM artist
  WHERE nationality = 'Bangladeshi'
```

```
View created.
```

Figure: View Creation

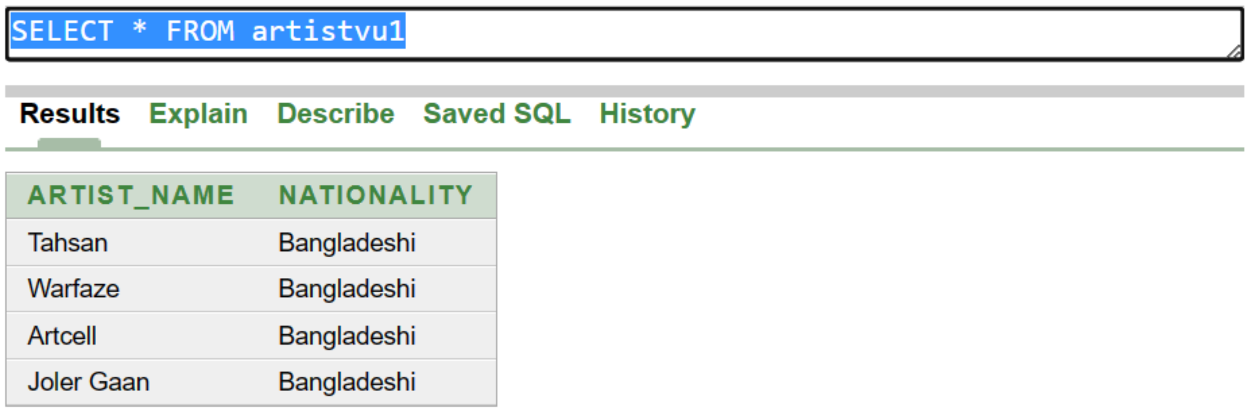


```
DESCRIBE artistvu1
```

Object Type **VIEW** Object **ARTISTVU1**

Table	Column	Data Type	Length	Precision	Scale	Primary Key
ARTISTVU1	ARTIST_NAME	Varchar2	150	-	-	-
	NATIONALITY	Varchar2	150	-	-	-

Figure: Description of the created view



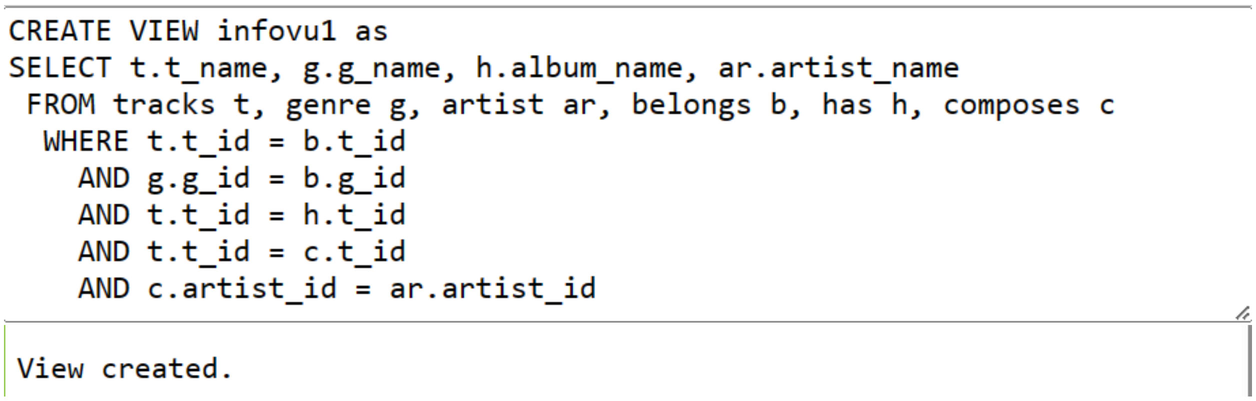
```
SELECT * FROM artistvu1
```

ARTIST_NAME	NATIONALITY
Tahsan	Bangladeshi
Warfaze	Bangladeshi
Artcell	Bangladeshi
Joler Gaan	Bangladeshi

Figure: Result of the created view

Complex View:

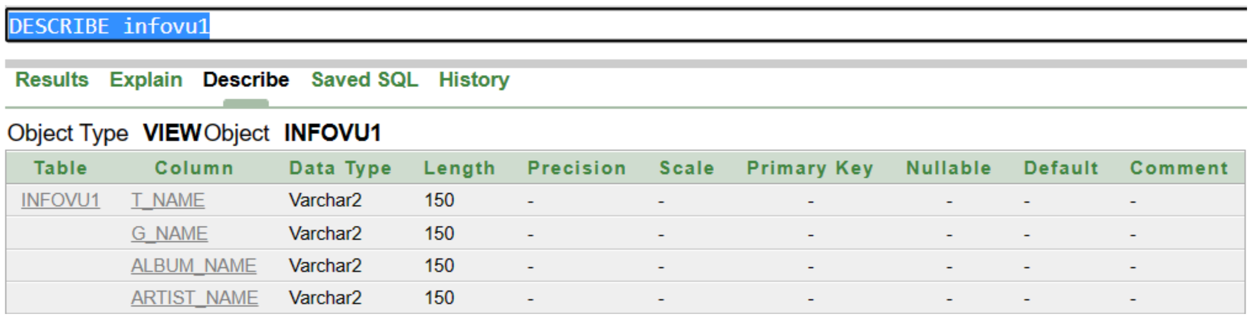
- Create a view to show the track name, genre name, album name and artist name



```
CREATE VIEW infovu1 as
SELECT t.t_name, g.g_name, h.album_name, ar.artist_name
 FROM tracks t, genre g, artist ar, belongs b, has h, composes c
  WHERE t.t_id = b.t_id
    AND g.g_id = b.g_id
    AND t.t_id = h.t_id
    AND t.t_id = c.t_id
    AND c.artist_id = ar.artist_id
```

```
View created.
```

Figure: View Creation

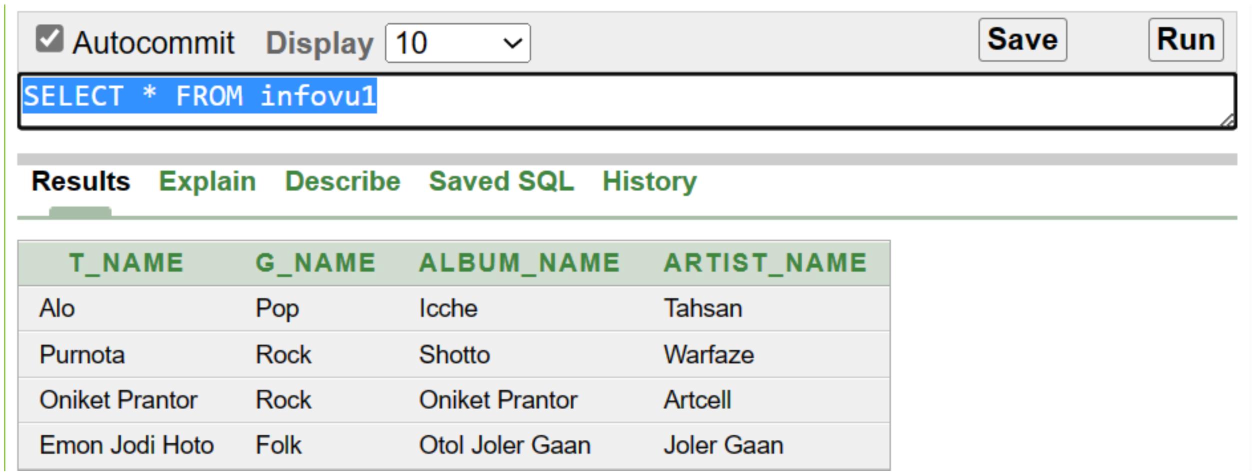


```
DESCRIBE infovu1
```

Object Type **VIEW** Object **INFOVU1**

Table	Column	Data Type	Length	Precision	Scale	Primary Key	Nullable	Default	Comment
INFOVU1	T_NAME	Varchar2	150	-	-	-	-	-	-
	G_NAME	Varchar2	150	-	-	-	-	-	-
	ALBUM_NAME	Varchar2	150	-	-	-	-	-	-
	ARTIST_NAME	Varchar2	150	-	-	-	-	-	-

Figure: Description of created view



```
SELECT * FROM infovu1
```

T_NAME	G_NAME	ALBUM_NAME	ARTIST_NAME
Alo	Pop	Icche	Tahsan
Purnota	Rock	Shotto	Warfaze
Oniket Prantor	Rock	Oniket Prantor	Artcell
Emon Jodi Hoto	Folk	Otol Joler Gaan	Joler Gaan

Figure: Result of the created view