IT 214 DBMS

Lab 8

Prepared by: Group S6_T9

ID	Name
201901076	Utsav Ladani
201901090	Pandar Mayur
201901131	Bhavya Solanki
201901304	Dev Joshi

Dhirubhai Ambani Institute of Information and Communication Technology

09 Nov, 2021

Original Design of the Database

1. **User** (***User_ID***, User_Name, Password, Email_ID, Mobile_Number, Plan_ID, User_Type)
2. **Music** (***Music_ID***, Music_Name, Music_Language, Premium_Information, Views,Likes, Music_Type, Artist_ID, Album_ID)
3. **Artist (*Artist_ID***, Artist_Name, Password, Total_song_made, Company_ID)
4. **Album** (***Album_ID***, Album_Name, Album_rating, Profit, Artist_ID, Company_ID)
5. **Production Company** (***Company_ID***, Company_Name, Base_Salary)
6. **Admin** (***Admin_ID***, Admin_Name, Password)
7. **Plan** (***Plan_ID***, Plan_Name, Plan_Duration, Plan_Price)
8. **Transaction** (***Transaction_ID***, Transaction_Mode, Sender_ID, Receiver_ID, Sender_Type, Receiver_Type)

List of All Dependencies

1. **User** (***User ID***, User_Name, Password, Email_ID, Mobile_Number, Plan_ID, User_Type)

   ➔ **Primary Key :** User_ID
   ➔ **Foreign Key :** Plan_ID

   ➔ **Functional Dependencies :**

   ❖ **User_ID→**
   User_Name, Password, Email_ID, Mobile_Number,Plan_ID, User_Type

   ❖ **Email_ID→**
   User_ID, User_Name, Password, Mobile_Number, Plan_ID, User_Type.

   ❖ **Mobile_Number→**
   User_ID, User_Name, Password, Email_ID, Plan_ID, User_Type

2. **Music** (***Music_ID***, Music_Name, Music_Language, Premium_Information, Views,Likes, Music_Type, Artist_ID, Album_ID)

   - **Primary Key :** Music_ID
   - **Foreign Key :** Artist_ID, Album_ID

   - **Functional Dependencies :**

     - **Music_ID→** Music_Name, Music_Language, Premium_Information, Views,Likes, Music_Type, Artist_ID, Album_ID

3. **Artist (*Artist_ID***, Artist_Name, Password, Total_song_made, Company_ID)

   - **Primary Key :** Artist_ID
   - **Foreign Key :** Company_ID

   - **Functional Dependencies :**

     - **Artist_ID→** Artist_Name, Password, Total_song_made, Company_ID

4. **Album** (***Album_ID***, Album_Name, Album_rating, Profit, Artist_ID, Company_ID)

   - **Primary Key :** Album_ID
   - **Foreign Key :** Artist_ID, Company_ID

   - **Functional Dependencies :**

     - **Album_ID→** Album_Name, Album_rating, Profit, Artist_ID, Company_ID

5. **Production Company** (***Company_ID***, Company_Name, Base_Salary)

   - **Primary Key :** Company_ID
   - **Foreign Key :** -----------------

   - **Functional Dependencies :**

     - **Company_ID→** Company_Name, Base Salary

6. **Admin** (***Admin_ID***, Admin_Name, Password)
    - ➔ **Primary Key :** Admin_ID
    - ➔ **Foreign Key :**
    - ➔ **Functional Dependencies :**
        - ❖ **Admin_ID→** Admin_Name, Password

7. **Plan** (***Plan_ID***, Plan_Name, Plan_Duration, Plan_Price)
    - ➔ **Primary Key :** Plan_ID
    - ➔ **Foreign Key :**
    - ➔ **Functional Dependencies :**
        - ❖ **Plan_ID→** Plan_Name, Plan_Duration, Plan_Price

8. **Transaction** (***Transaction_ID***, Transaction_Mode, Sender_ID, Receiver_ID, Sender_Type, Receiver_Type)
    - ➔ **Primary Key :** Transaction_ID
    - ➔ **Foreign Key :** -------------------
    - ➔ **Functional Dependencies :**
        - ❖ **Transaction_ID→** Transaction_Mode, Sender_ID, Receiver_ID, Sender_Type, Receiver_Type

Anomalies and redundancy:

- There are no anomalies and redundancy in our schema.

Normalize the database up to 1NF

1. **User** (***User_ID***, User_Name, Password, Email_ID, Mobile_Numb9er, Plan_ID, User_Type)

- None of the attributes User_Name, User_ID, Password, Email_ID, Mobile_Number(assume Unique(1per User)), Plan_ID,User_Type are multivalued. So, **User** is already normalized to 1NF.

2. **Music** (***<u>Music_ID</u>***, Music_Name, Music_Language, Premium_Information, Views,Likes, Music_Type, Artist_ID, Album_ID)

- **Music** is already normalized to 1NF because None of the attributes Music_ID, Music_Name, Music_Language, Premium_Information, Views,Likes, Music_Type, Artist_ID, Album_ID are multivalued.

3. **Artist (*<u>Artist_ID</u>***, Artist_Name, Password, Total_song_made, Company_ID)

- None of the attributes Artist_ID, Artist_Name, Password, Total_song_made, Company_ID are multivalued. So, **Artist** is already normalized to 1NF.

4. **Album** (***<u>Album_ID</u>***, Album_Name, Album_rating, Profit, Artist_ID, Company_ID)

- **Album** is already normalized to 1NF because None of the attributes Album_ID, Album_Name, Album_rating, Profit, Artist_ID, Company_ID are multivalued.

5. **Production Company** (***<u>Company_ID</u>***, Company_Name, Base_Salary)

- None of the attributes Company_ID, Company_Name, Base_Salary are multivalued. So, **Production Company** is already normalized to 1NF.

6. **Admin** (***<u>Admin_ID</u>***, Admin_Name, Password)

- **Admin** is already normalized to 1NF because None of the attributes Admin_ID, Admin_Name, Password are multivalued.

7. **Plan** (***<u>Plan_ID</u>***, Plan_Name, Plan_Duration, Plan_Price)

- None of the attributes Plan_ID, Plan_Name, Plan_Duration, Plan_Price are multivalued. So, **Plan** is already normalized to 1NF.

8. **Transaction** (***<u>Transaction_ID</u>***, Transaction_Mode, Sender_ID, Receiver_ID, Sender_Type, Receiver_Type)

- **Transaction** is already normalized to 1NF because None of the attributes Transaction_ID, Transaction_Mode, Sender_ID, Receiver_ID, Sender_Type, Receiver_Type are multivalued.

Normalize the database to 2NF

1. **User** (***User_ID***, User_Name, Password, Email_ID, Mobile_Number, Plan_ID, User_Type)

- **User** is already in 1NF and also no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table (Hence no Partial Dependency). So **User** is already normalized to 2NF.

2. **Music** (***Music_ID***, Music_Name, Music_Language, Premium_Information, Views,Likes, Music_Type, Artist_ID, Album_ID)

- **Music** is already normalized to 2NF because it is already normalized to 1NF and also no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table (Hence no Partial Dependency).

3. **Artist (*Artist_ID***, Artist_Name, Password, Total_song_made, Company_ID)

- **Artist** is already in 1NF and also no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table (Hence no Partial Dependency). So **Artist** is already normalized to 2NF.

4. **Album** (***Album_ID***, Album_Name, Album_rating, Profit, Artist_ID, Company_ID)

- **Album** is already normalized to 2NF because it is already normalized to 1NF and also no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table (Hence no Partial Dependency).

5. **Production Company** (***Company_ID***, Company_Name, Base_Salary)

- **Production Company** is already in 1NF and also no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table (Hence no Partial Dependency). So **Production Comapany** is already normalized to 2NF.

6. **Admin** (***Admin_ID***, Admin_Name, Password)

- **Admin** is already normalized to 2NF because it is already normalized to 1NF and also no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table (Hence no Partial Dependency).

7. **Plan** (***Plan_ID***, Plan_Name, Plan_Duration, Plan_Price)

- **Plan** is already in 1NF and also no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table (Hence no Partial Dependency). So **Plan** is already normalized to 2NF.

8. **Transaction** (***<u>Transaction_ID</u>***, Transaction_Mode, Sender_ID, Receiver_ID, Sender_Type, Receiver_Type)

- **Transaction** is already normalized to 2NF because it is already normalized to 1NF and also no non-prime attribute (attributes which are not part of any candidate key) is dependent on any proper subset of any candidate key of the table (Hence no Partial Dependency).

Normalize the database to 3NF/BCNF

1. **User** (***<u>User_ID</u>***, User_Name, Password, Email_ID, Mobile_Number, Plan_ID, User_Type)

- **User** is already normalized to 2NF and no non key attributes are transitively dependant on the primary key attribute (Hence no transitive dependencies) so **User** is already normalized to 3NF and also in all A → B type relation has A as Candidate Key So it is also in BCNF.

2. **Music** (***<u>Music_ID</u>***, Music_Name, Music_Language, Premium_Information, Views,Likes, Music_Type, Artist_ID, Album_ID)

- **Music** is already normalized to 3NF because it is already normalized to 2NF and no non key attributes are transitively dependent on the primary key attribute (Hence no transitive dependencies). It is also normalized to BCNF because all A → B type relations have A as Candidate key.

3. **Artist (*<u>Artist_ID</u>***, Artist_Name, Password, Total_song_made, Company_ID)

- **Artist** is already normalized to 2NF and no non key attributes are transitively dependant on the primary key attribute (Hence no transitive dependencies) so **Artist** is already normalized to 3NF and also in all A → B type relation has A as Candidate Key So it is also in BCNF.

4. **Album** (***<u>Album_ID</u>***, Album_Name, Album_rating, Profit, Artist_ID, Company_ID)

- **Album** is already normalized to 3NF because it is already normalized to 2NF and no non key attributes are transitively dependent on the primary key attribute (Hence no transitive dependencies). It is also normalized to BCNF because all A → B type relations have A as Candidate key.

5. **Production Company** (***Company_ID***, Company_Name, Base_Salary)

- **Production Company** is already normalized to 2NF and no non key attributes are transitively dependant on the primary key attribute (Hence no transitive dependencies) so **Production Company** is already normalized to 3NF and also in all A → B type relation has A as Candidate Key So it is also in BCNF.

6. **Admin** (***Admin_ID***, Admin_Name, Password)

- **Admin** is already normalized to 3NF because it is already normalized to 2NF and no non key attributes are transitively dependent on the primary key attribute (Hence no transitive dependencies). It is also normalized to BCNF because all A → B type relations have A as Candidate key.

7. **Plan** (***Plan_ID***, Plan_Name, Plan_Duration, Plan_Price)

- **Plan** is already normalized to 2NF and no non key attributes are transitively dependant on the primary key attribute (Hence no transitive dependencies) so **Plan** is already normalized to 3NF and also in all A → B type relation has A as Candidate Key So it is also in BCNF.

8. **Transaction** (***Transaction_ID***, Transaction_Mode, Sender_ID, Receiver_ID, Sender_Type, Receiver_Type)

- **Transaction** is already normalized to 3NF because it is already normalized to 2NF and no non key attributes are transitively dependent on the primary key attribute (Hence no transitive dependencies). It is also normalized to BCNF because all A → B type relations have A as Candidate key.

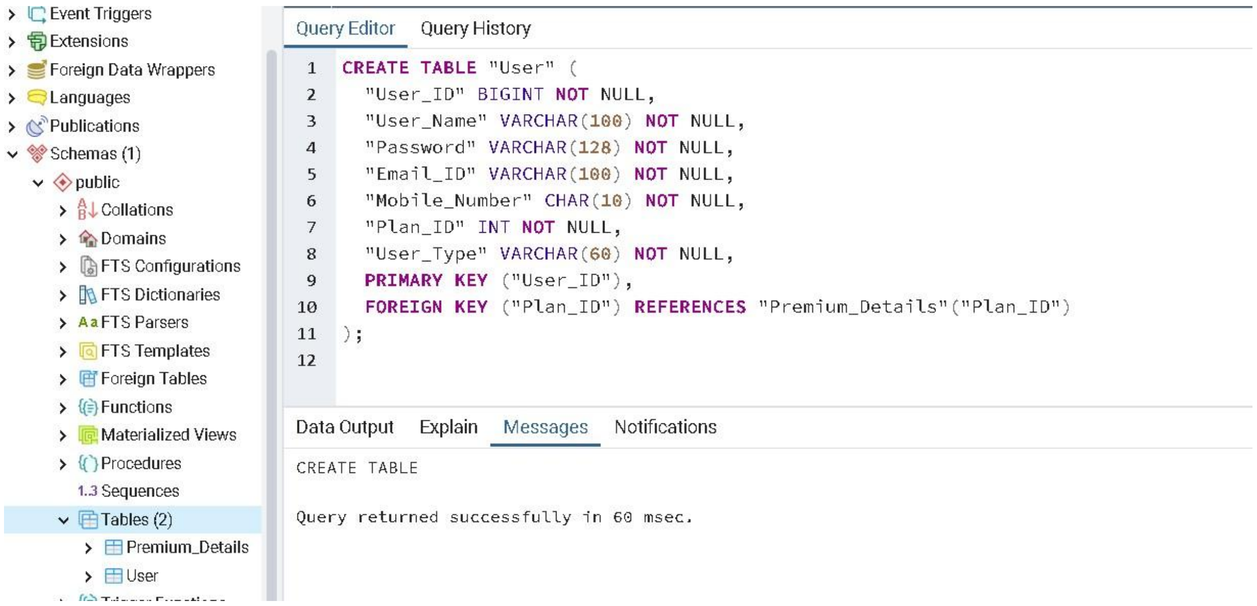
Updated DDL Script

```
CREATE TABLE "User" (
  "User_ID" BIGINT NOT NULL,
  "User_Name" VARCHAR(100) NOT NULL,
  "Password" VARCHAR(128) NOT NULL,
  "Email_ID" VARCHAR(100) NOT NULL,
  "Mobile_Number" CHAR(10) NOT NULL,
  "Plan_ID" INT NOT NULL,
  "User_Type" VARCHAR(60) NOT NULL,
  PRIMARY KEY ("User_ID"),
  FOREIGN KEY ("Plan_ID") REFERENCES
"Premium_Details"("Plan_ID")
);


CREATE TABLE "Production_Company"
(
  "Company_ID" BIGINT NOT NULL,
  "Company_Name" VARCHAR(100) NOT NULL,
  "Base_salary" BIGINT NOT NULL,
  PRIMARY KEY ("Company_ID")
);

CREATE TABLE "Admin"
(
  "Admin_ID" BIGINT NOT NULL,
  "Admin_Name" VARCHAR(100) NOT NULL,
  "Password" VARCHAR(128) NOT NULL,
  PRIMARY KEY ("Admin_ID")
);

CREATE TABLE "Premium_Details"
(
  "Plan_ID" BIGINT NOT NULL,
  "Plan_Name" VARCHAR(100) NOT NULL,
  "Plan_Duration" INT NOT NULL,
  "Plan_Price" INT NOT NULL,
  PRIMARY KEY ("Plan_ID")
);

CREATE TABLE "Artist"
(
```

```sql
  "Artist_ID" BIGINT NOT NULL,
  "Artist_Name" VARCHAR(100) NOT NULL,
  "Password" VARCHAR(128) NOT NULL,
  "Total_Song_Made" INT NOT NULL,
  "Company_ID" BIGINT NOT NULL,
  PRIMARY KEY ("Artist_ID"),
  FOREIGN KEY ("Company_ID") REFERENCES
"Production_Company"("Company_ID")
);

CREATE TABLE "Album"
(
  "Album_ID" BIGINT NOT NULL,
  "Album_Name" VARCHAR(100) NOT NULL,
  "Album_Rating" INT NOT NULL,
  "Profit" BIGINT NOT NULL,
  "Artist_ID" BIGINT NOT NULL,
  "Company_ID" BIGINT NOT NULL,
  PRIMARY KEY ("Album_ID"),
  FOREIGN KEY ("Artist_ID") REFERENCES
"Artist"("Artist_ID"),
  FOREIGN KEY ("Company_ID") REFERENCES
"Production_Company"("Company_ID")
);

CREATE TABLE "Music"
(
  "Music_ID" BIGINT NOT NULL,
  "Music_Name" VARCHAR(100) NOT NULL,
  "Music_Language" VARCHAR(100) NOT NULL,
  "Premium_Information" BOOLEAN NOT NULL,
  "Views" BIGINT NOT NULL,
  "Likes" BIGINT NOT NULL,
  "Music_Type" VARCHAR(60) NOT NULL,
  "Artist_ID" BIGINT NOT NULL,
  "Album_ID" BIGINT NOT NULL,
  PRIMARY KEY ("Music_ID"),
  FOREIGN KEY ("Artist_ID") REFERENCES
"Artist"("Artist_ID"),
  FOREIGN KEY ("Album_ID") REFERENCES "Album"("Album_ID")
);

CREATE TABLE "Transaction"
(
  "Transaction_ID" BIGINT NOT NULL,
  "Transaction_Mode" VARCHAR(20) NOT NULL,
  "Sender_ID" BIGINT NOT NULL,
  "Receiver_ID" BIGINT NOT NULL,
  "Sender_Type" VARCHAR(20) NOT NULL,
  "Receiver_Type" VARCHAR(20) NOT NULL,
  PRIMARY KEY ("Transaction_ID")
);
```

Snapshot of Create Table using DDL

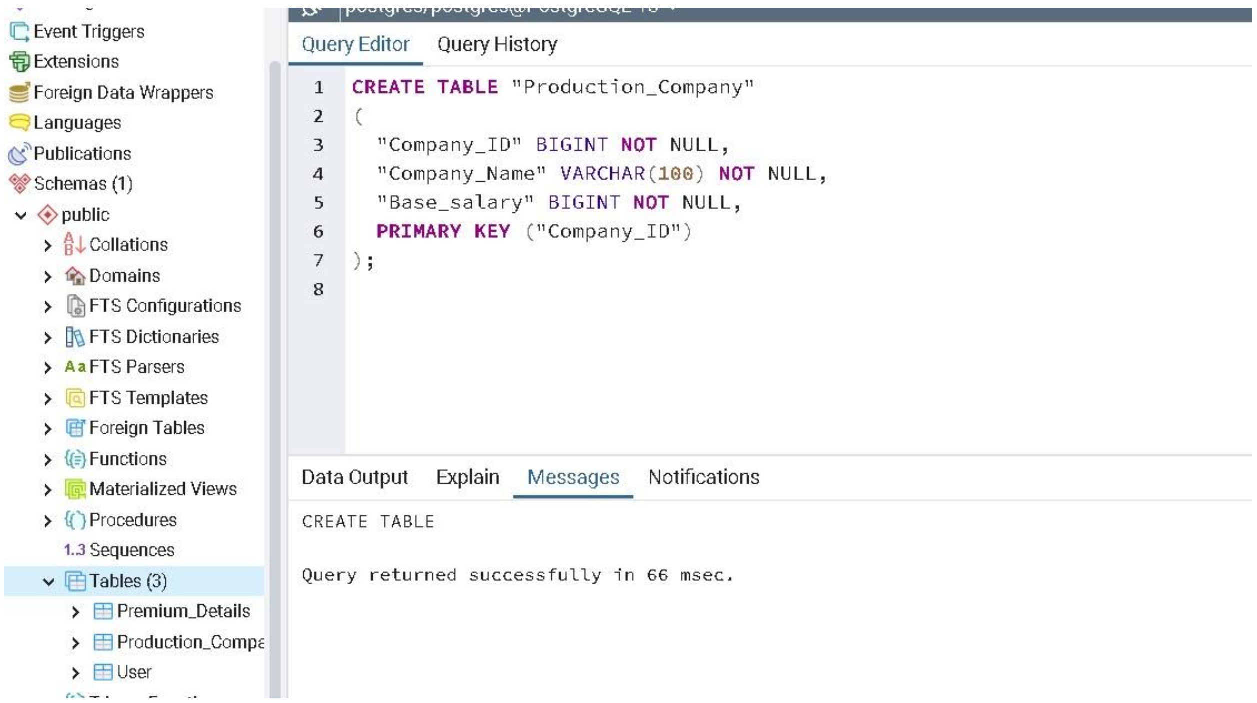
1. **User**



```
CREATE TABLE "User" (
  "User_ID" BIGINT NOT NULL,
  "User_Name" VARCHAR(100) NOT NULL,
  "Password" VARCHAR(128) NOT NULL,
  "Email_ID" VARCHAR(100) NOT NULL,
  "Mobile_Number" CHAR(10) NOT NULL,
  "Plan_ID" INT NOT NULL,
  "User_Type" VARCHAR(60) NOT NULL,
  PRIMARY KEY ("User_ID"),
  FOREIGN KEY ("Plan_ID") REFERENCES "Premium_Details"("Plan_ID")
);
```

```
CREATE TABLE

Query returned successfully in 60 msec.
```

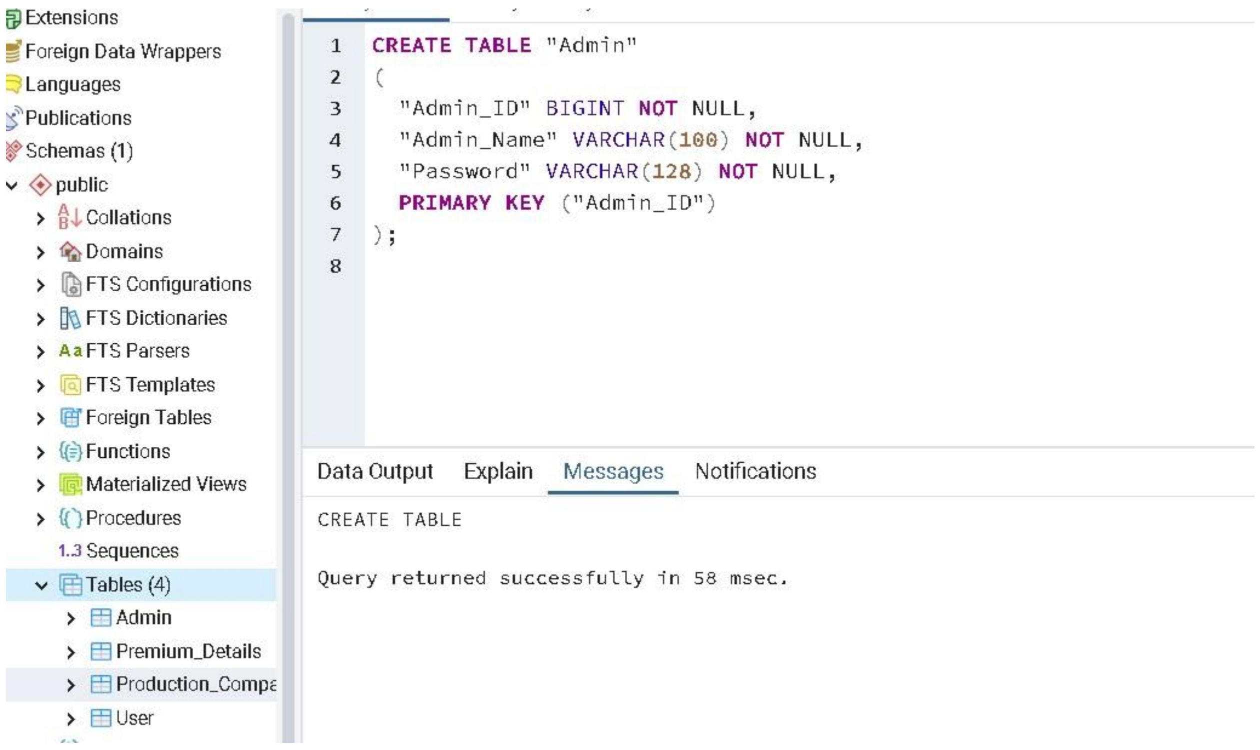
2. Production_Company



```
CREATE TABLE "Production_Company"
(
  "Company_ID" BIGINT NOT NULL,
  "Company_Name" VARCHAR(100) NOT NULL,
  "Base_salary" BIGINT NOT NULL,
  PRIMARY KEY ("Company_ID")
);
```

```
CREATE TABLE

Query returned successfully in 66 msec.
```

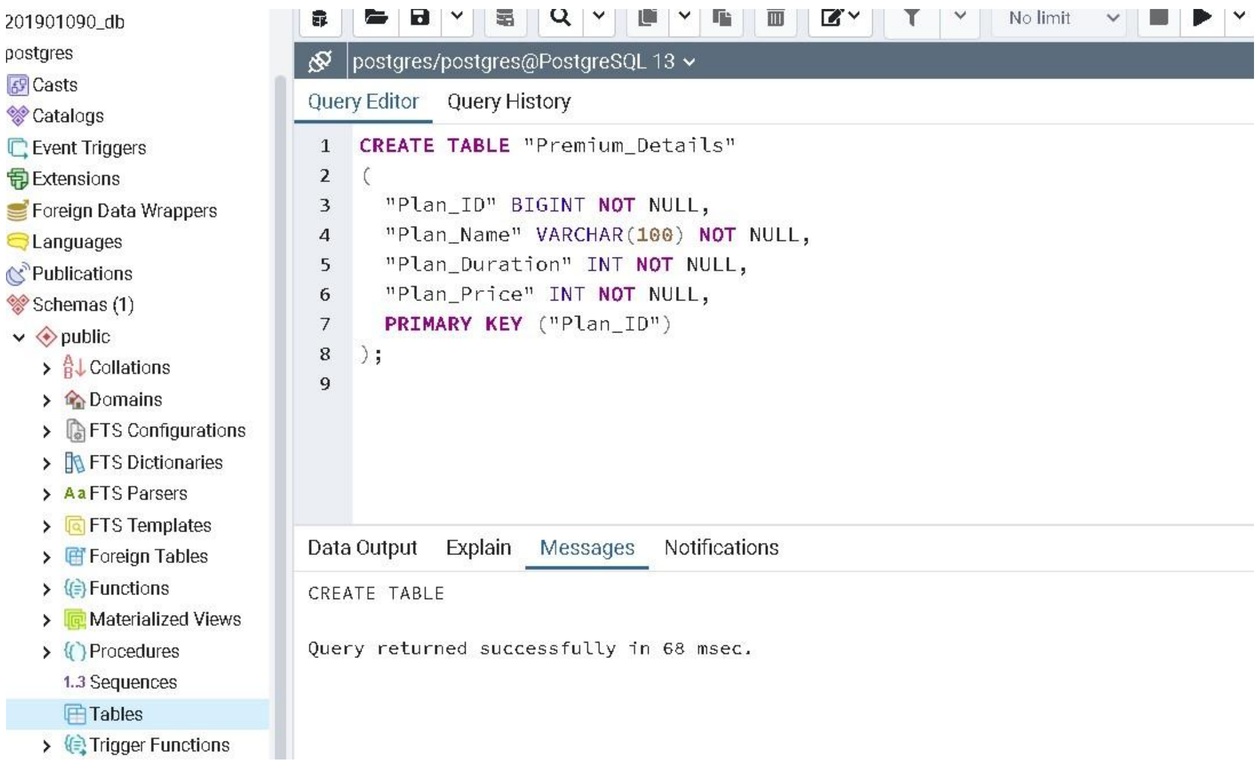
3. Admin



```
CREATE TABLE "Admin"
(
  "Admin_ID" BIGINT NOT NULL,
  "Admin_Name" VARCHAR(100) NOT NULL,
  "Password" VARCHAR(128) NOT NULL,
  PRIMARY KEY ("Admin_ID")
);
```

```
CREATE TABLE

Query returned successfully in 58 msec.
```

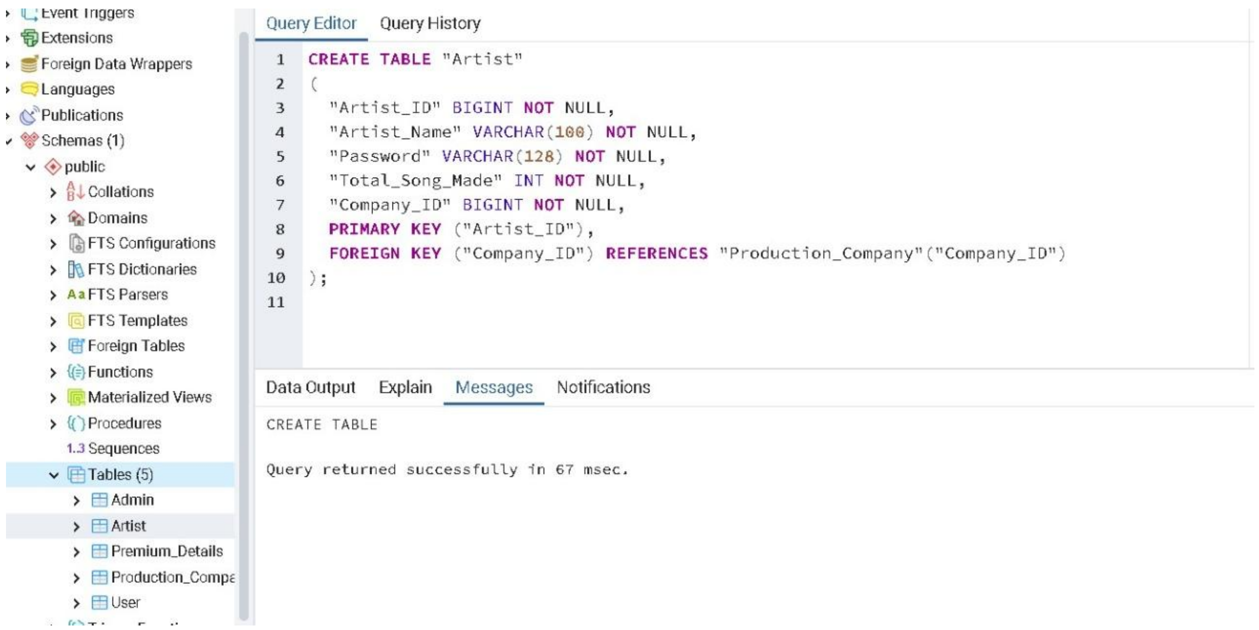
4. Premium_Details



```
CREATE TABLE "Premium_Details"
(
  "Plan_ID" BIGINT NOT NULL,
  "Plan_Name" VARCHAR(100) NOT NULL,
  "Plan_Duration" INT NOT NULL,
  "Plan_Price" INT NOT NULL,
  PRIMARY KEY ("Plan_ID")
);
```

```
CREATE TABLE

Query returned successfully in 68 msec.
```

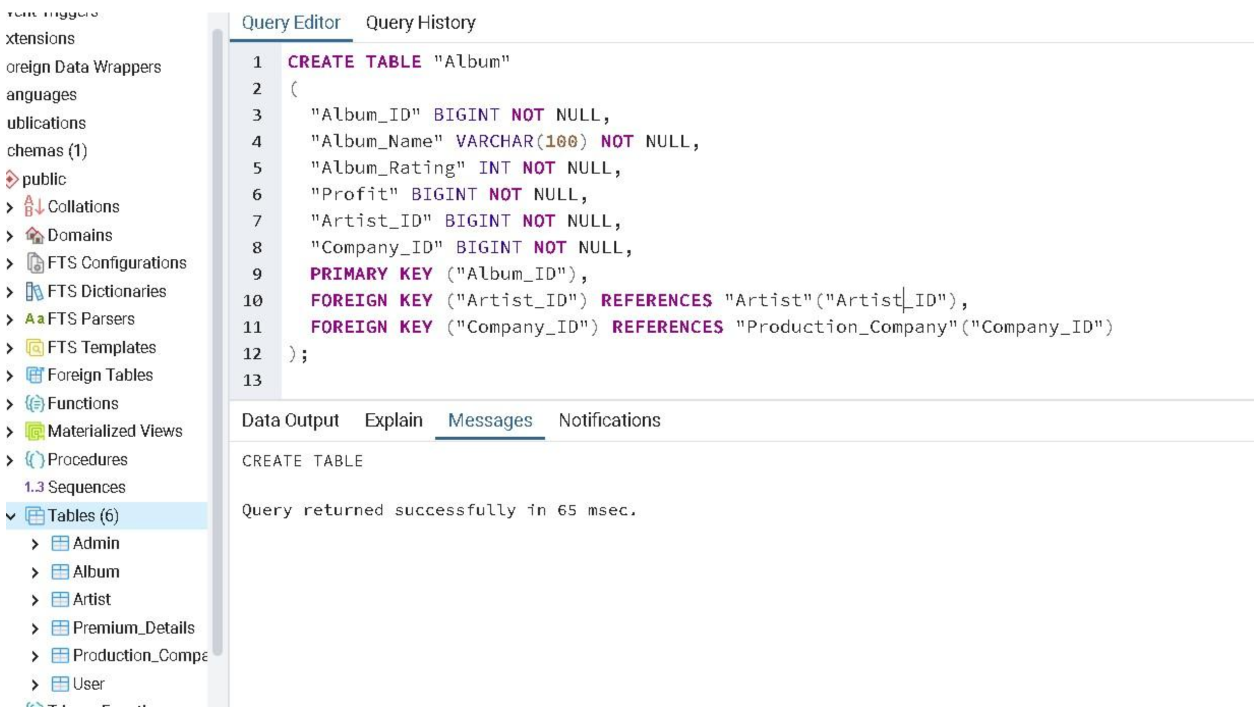
5. Artist



```
CREATE TABLE "Artist"
(
  "Artist_ID" BIGINT NOT NULL,
  "Artist_Name" VARCHAR(100) NOT NULL,
  "Password" VARCHAR(128) NOT NULL,
  "Total_Song_Made" INT NOT NULL,
  "Company_ID" BIGINT NOT NULL,
  PRIMARY KEY ("Artist_ID"),
  FOREIGN KEY ("Company_ID") REFERENCES "Production_Company"("Company_ID")
);
```

```
CREATE TABLE

Query returned successfully in 67 msec.
```

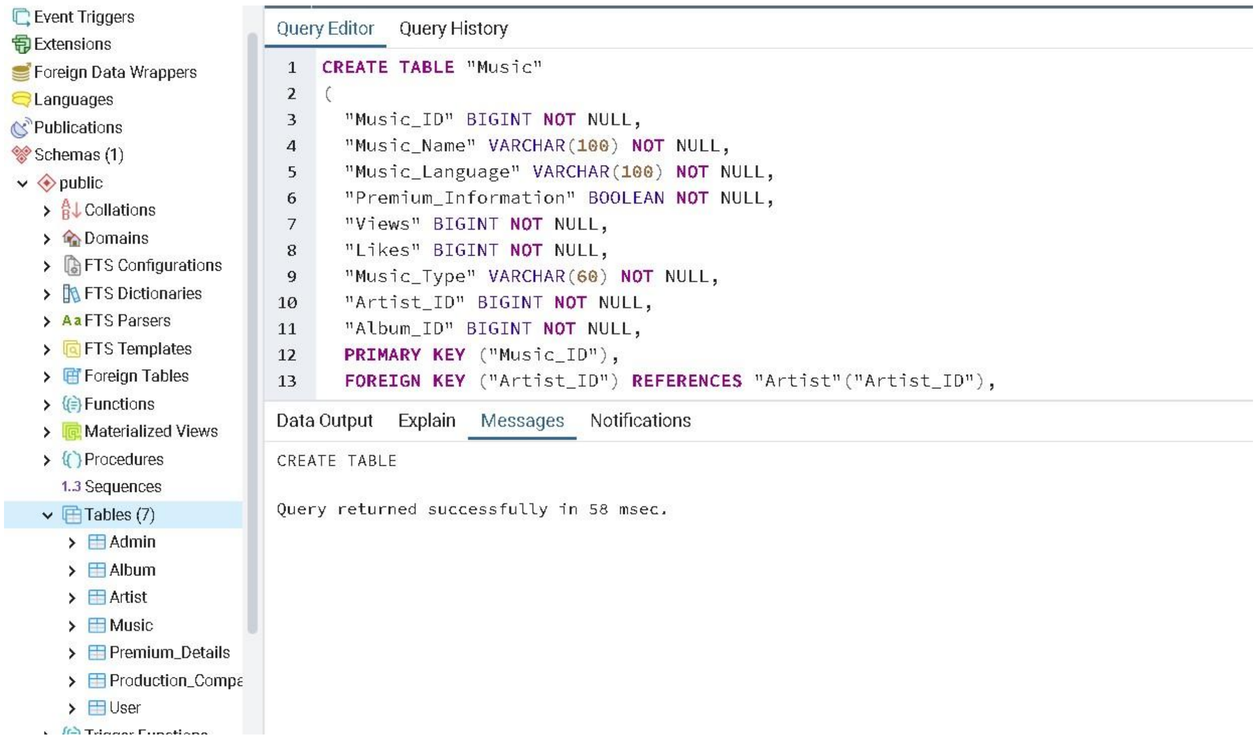
6. Album



```
CREATE TABLE "Album"
(
  "Album_ID" BIGINT NOT NULL,
  "Album_Name" VARCHAR(100) NOT NULL,
  "Album_Rating" INT NOT NULL,
  "Profit" BIGINT NOT NULL,
  "Artist_ID" BIGINT NOT NULL,
  "Company_ID" BIGINT NOT NULL,
  PRIMARY KEY ("Album_ID"),
  FOREIGN KEY ("Artist_ID") REFERENCES "Artist"("Artist_ID"),
  FOREIGN KEY ("Company_ID") REFERENCES "Production_Company"("Company_ID")
);
```

```
CREATE TABLE

Query returned successfully in 65 msec.
```

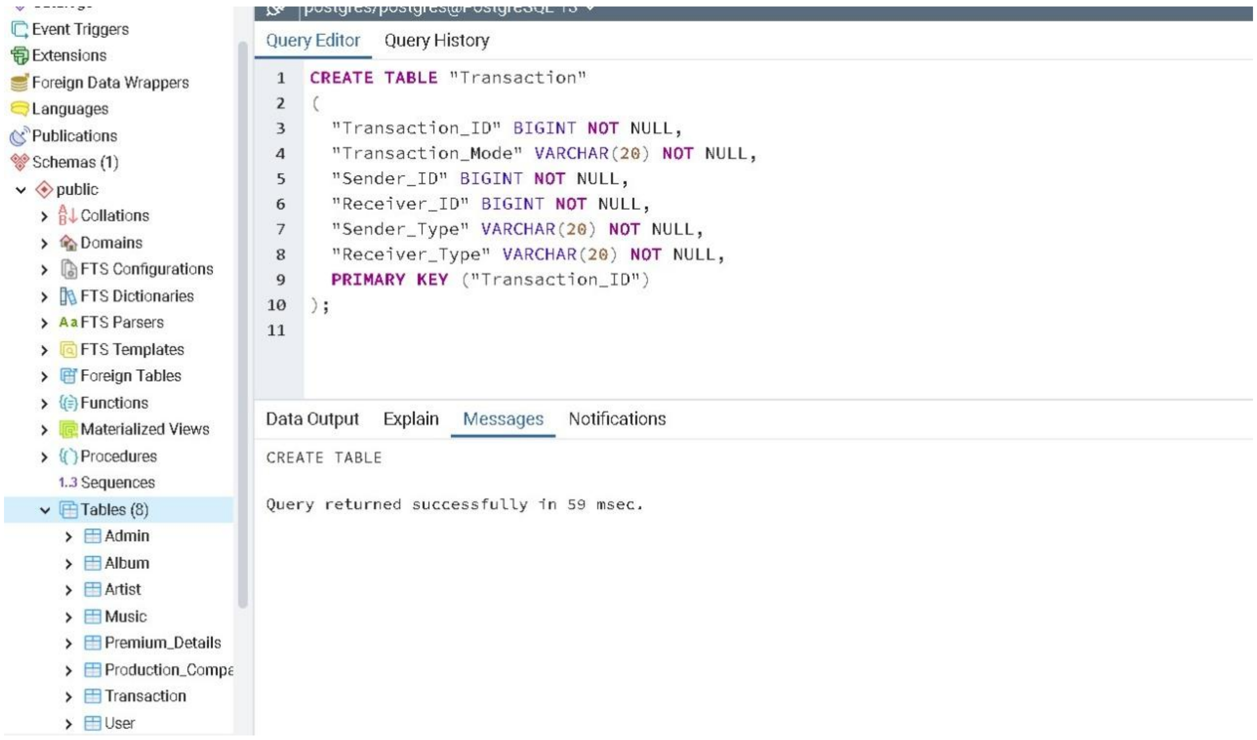
7. Music



```
CREATE TABLE "Music"
(
  "Music_ID" BIGINT NOT NULL,
  "Music_Name" VARCHAR(100) NOT NULL,
  "Music_Language" VARCHAR(100) NOT NULL,
  "Premium_Information" BOOLEAN NOT NULL,
  "Views" BIGINT NOT NULL,
  "Likes" BIGINT NOT NULL,
  "Music_Type" VARCHAR(60) NOT NULL,
  "Artist_ID" BIGINT NOT NULL,
  "Album_ID" BIGINT NOT NULL,
  PRIMARY KEY ("Music_ID"),
  FOREIGN KEY ("Artist_ID") REFERENCES "Artist"("Artist_ID"),
```

```
CREATE TABLE

Query returned successfully in 58 msec.
```

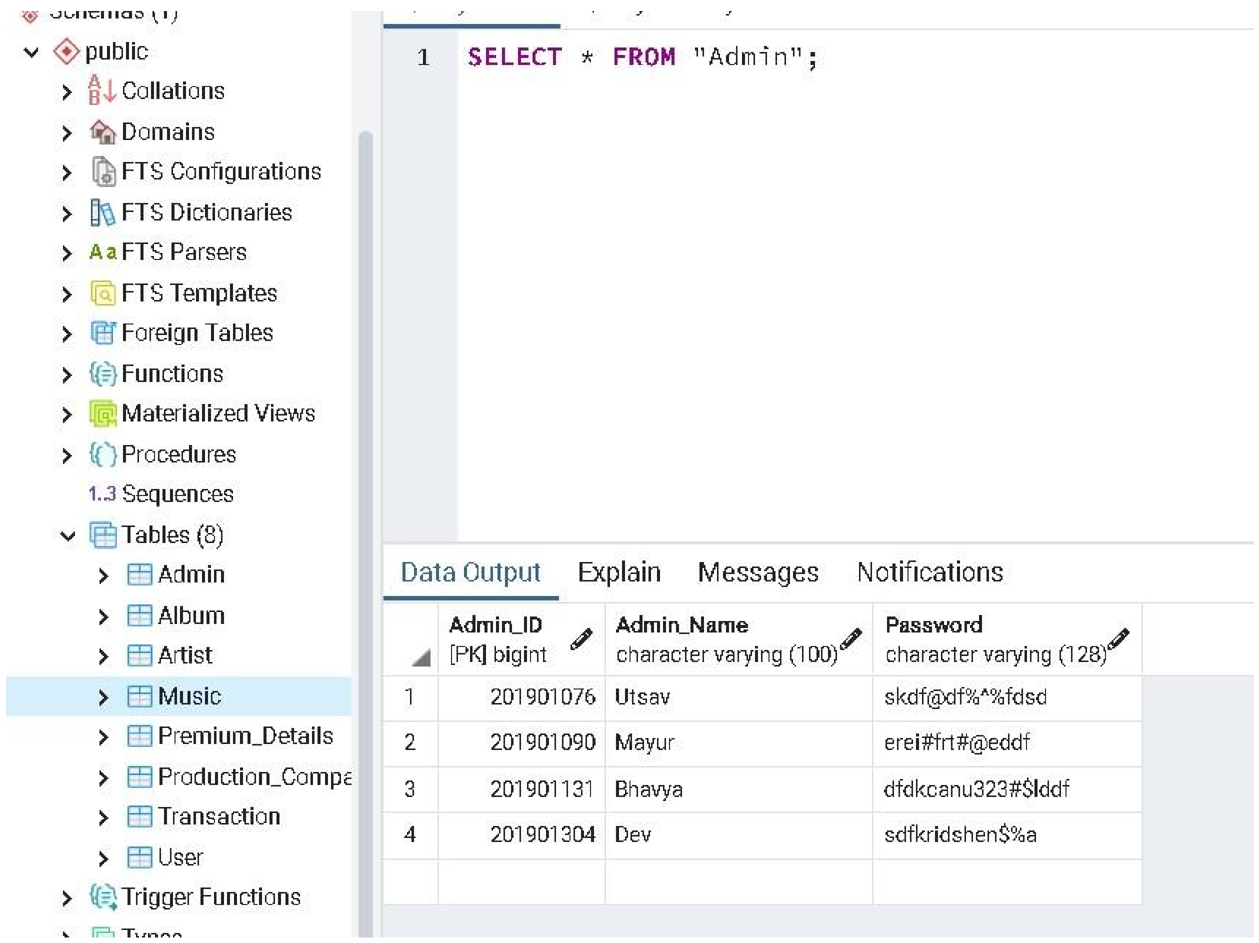
8. Transaction



```
CREATE TABLE "Transaction"
(
  "Transaction_ID" BIGINT NOT NULL,
  "Transaction_Mode" VARCHAR(20) NOT NULL,
  "Sender_ID" BIGINT NOT NULL,
  "Receiver_ID" BIGINT NOT NULL,
  "Sender_Type" VARCHAR(20) NOT NULL,
  "Receiver_Type" VARCHAR(20) NOT NULL,
  PRIMARY KEY ("Transaction_ID")
);
```

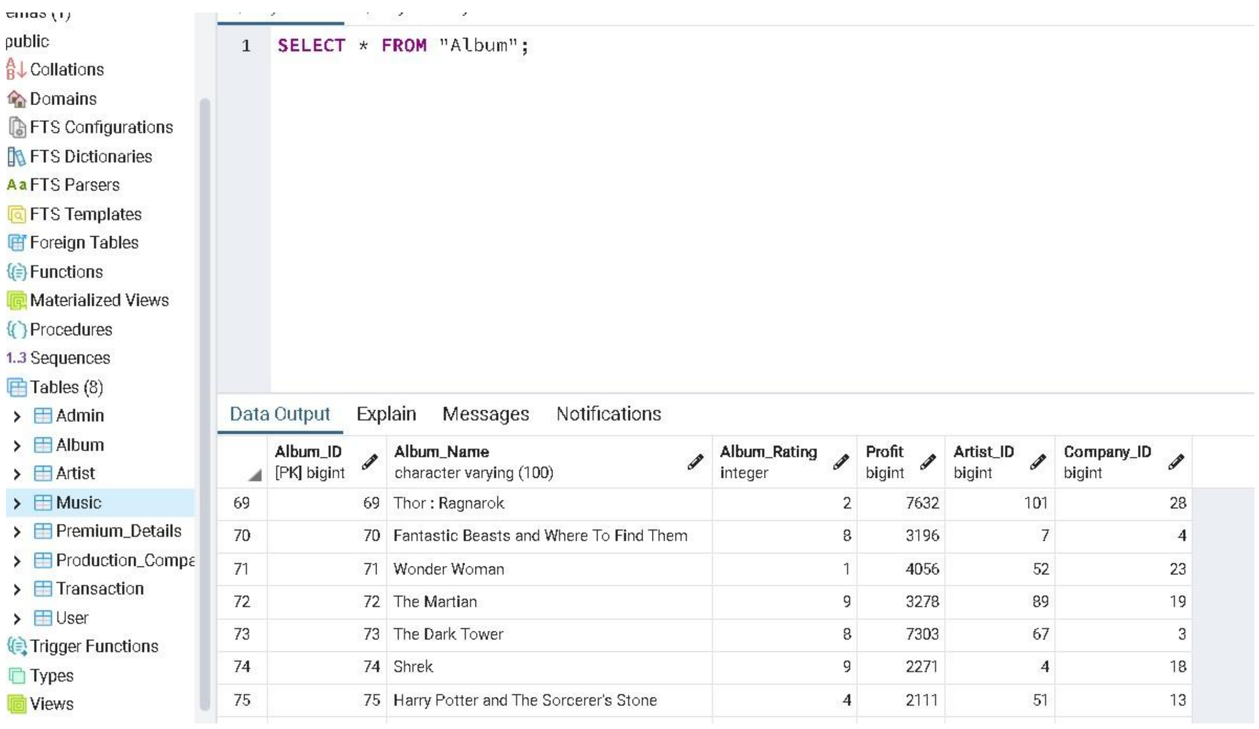
```
CREATE TABLE

Query returned successfully in 59 msec.
```

Data snapshots



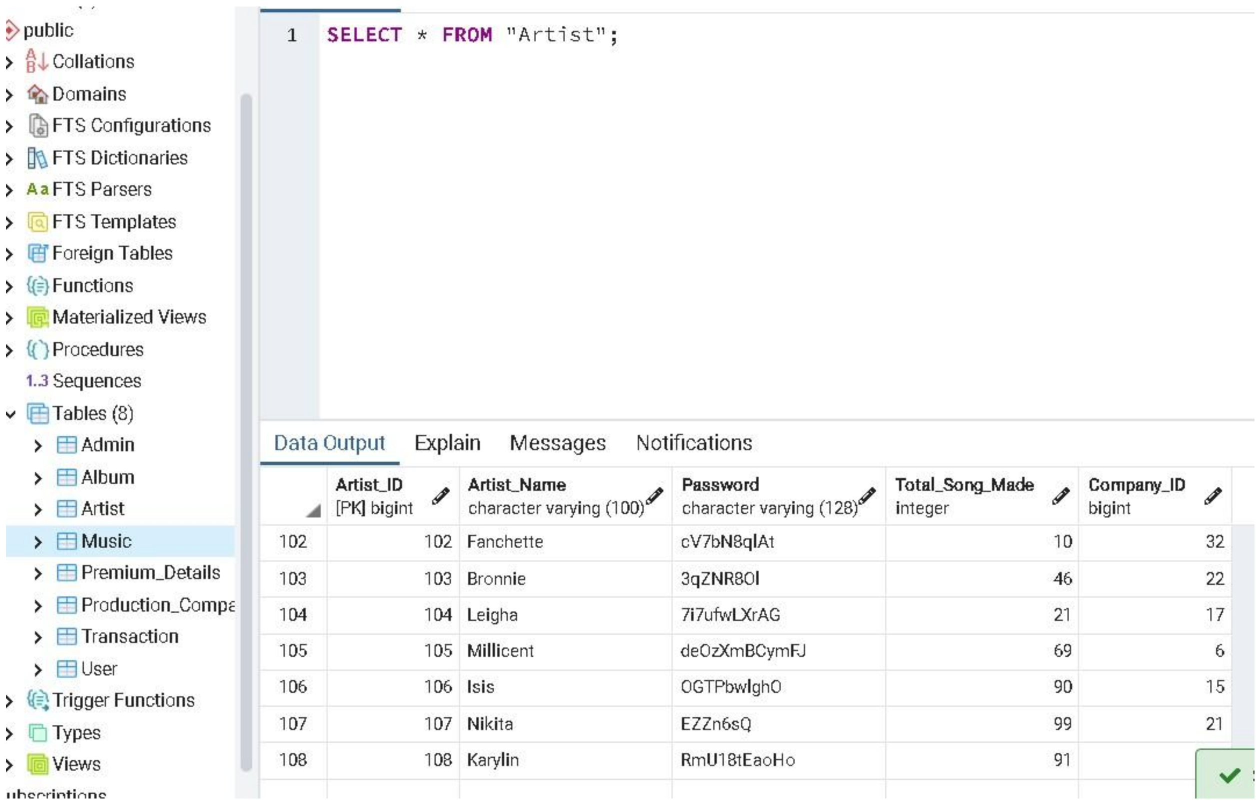
```sql
SELECT * FROM "Admin";
```

	Admin_ID [PK] bigint	Admin_Name character varying (100)	Password character varying (128)
1	201901076	Utsav	skdf@df%^%fdsd
2	201901090	Mayur	erei#frt#@eddf
3	201901131	Bhavya	dfdkcanu323#$lddf
4	201901304	Dev	sdfkridshen$%a



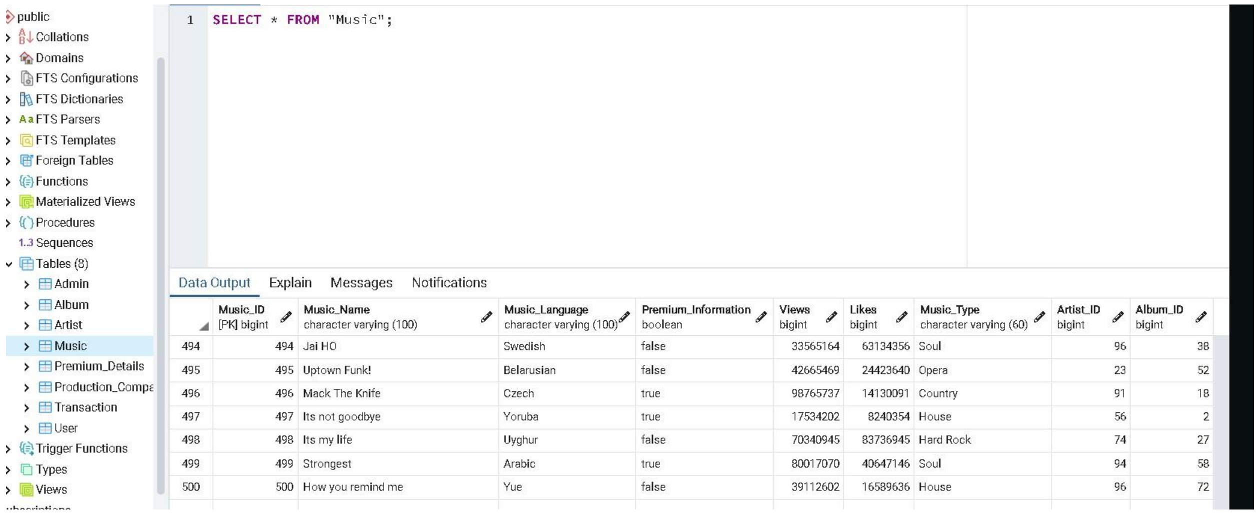
```sql
SELECT * FROM "Album";
```

	Album_ID [PK] bigint	Album_Name character varying (100)	Album_Rating integer	Profit bigint	Artist_ID bigint	Company_ID bigint
69	69	Thor : Ragnarok	2	7632	101	28
70	70	Fantastic Beasts and Where To Find Them	8	3196	7	4
71	71	Wonder Woman	1	4056	52	23
72	72	The Martian	9	3278	89	19
73	73	The Dark Tower	8	7303	67	3
74	74	Shrek	9	2271	4	18
75	75	Harry Potter and The Sorcerer's Stone	4	2111	51	13



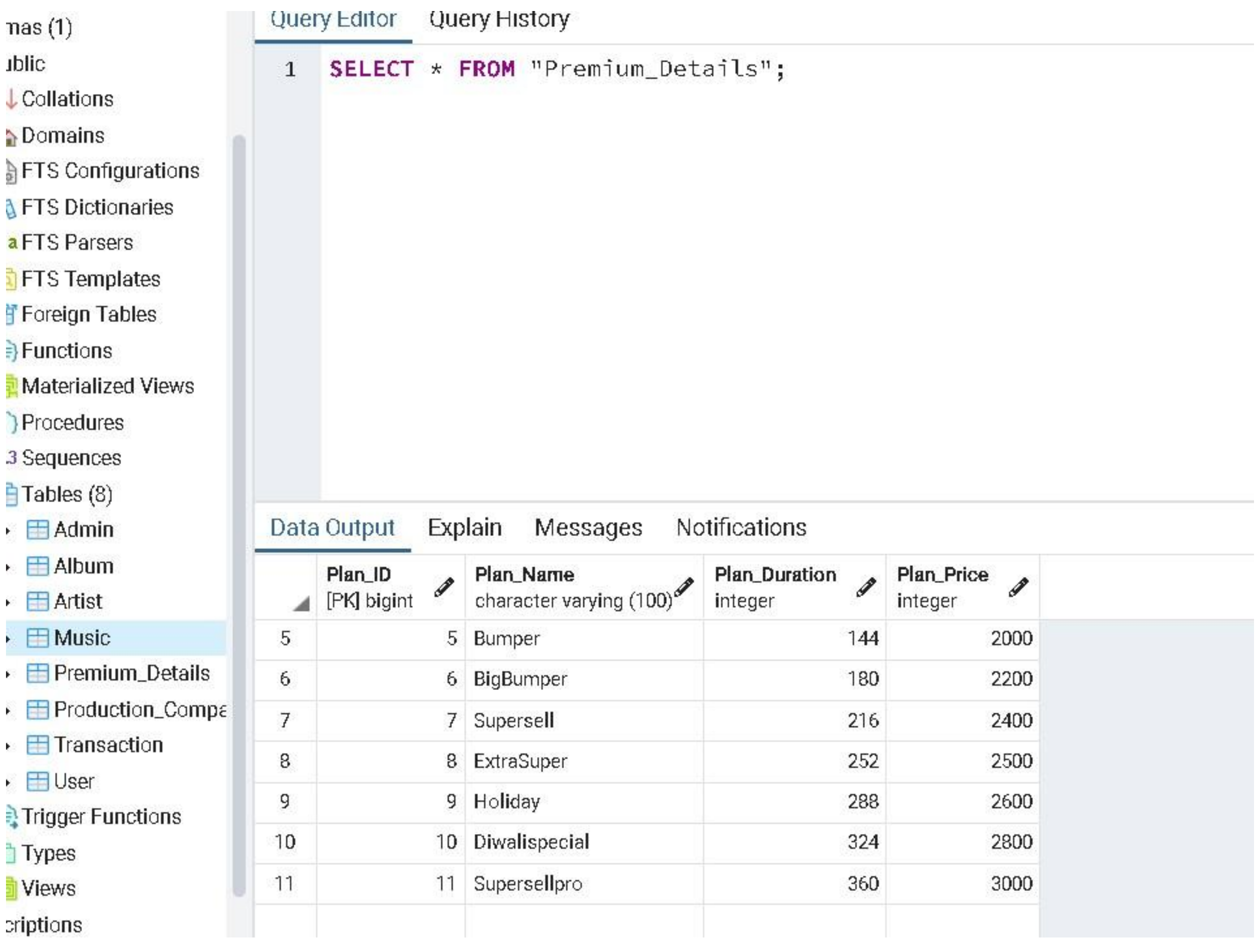
```
SELECT * FROM "Artist";
```

	Artist_ID [PK] bigint	Artist_Name character varying (100)	Password character varying (128)	Total_Song_Made integer	Company_ID bigint
102	102	Fanchette	cV7bN8qlAt	10	32
103	103	Bronnie	3qZNR8Ol	46	22
104	104	Leigha	7i7ufwLXrAG	21	17
105	105	Millicent	deOzXmBCymFJ	69	6
106	106	Isis	OGTPbwlghO	90	15
107	107	Nikita	EZZn6sQ	99	21
108	108	Karylin	RmU18tEaoHo	91	



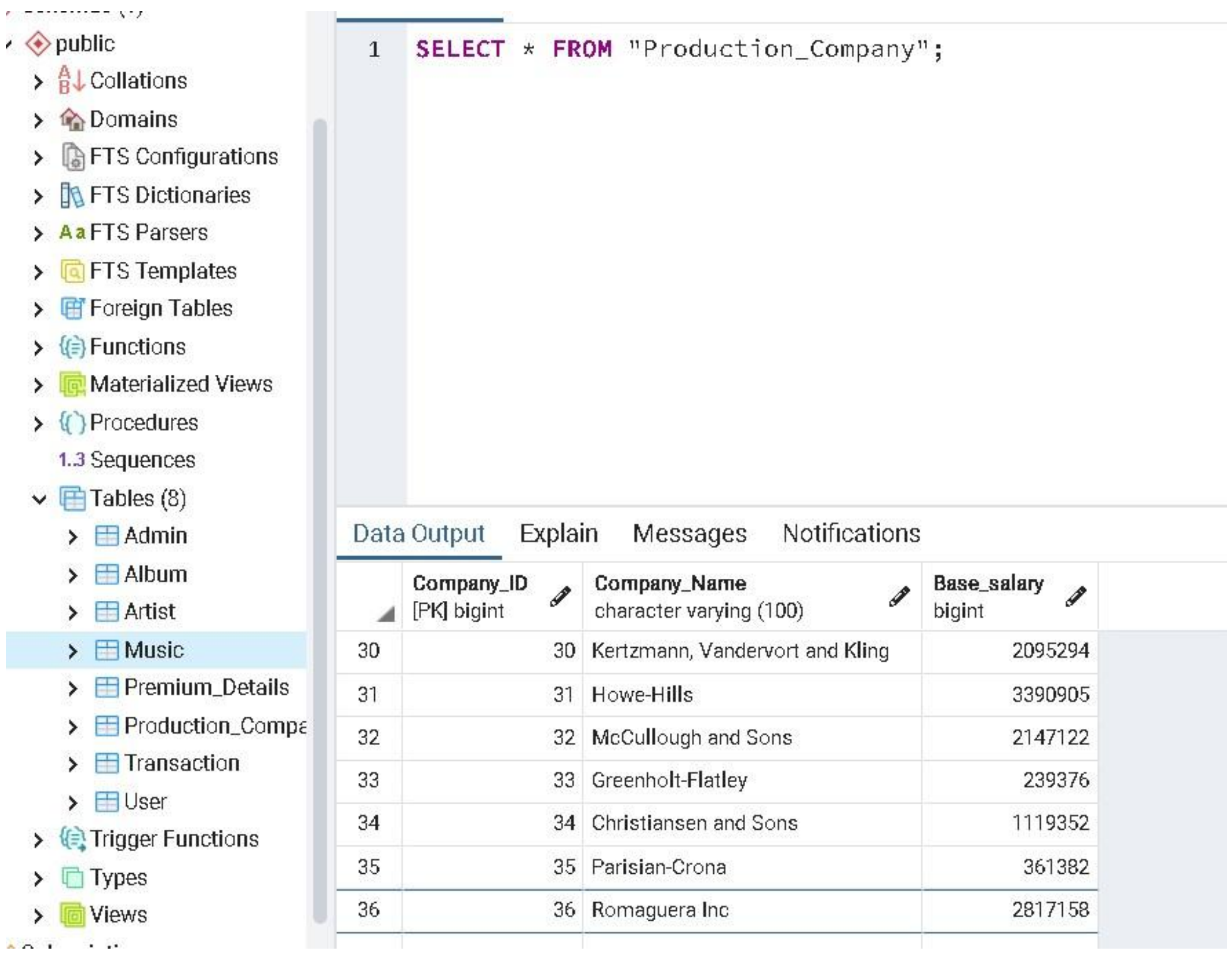
```
SELECT * FROM "Music";
```

	Music_ID [PK] bigint	Music_Name character varying (100)	Music_Language character varying (100)	Premium_Information boolean	Views bigint	Likes bigint	Music_Type character varying (60)	Artist_ID bigint	Album_ID bigint
494	494	Jai HO	Swedish	false	33565164	63134356	Soul	96	38
495	495	Uptown Funk!	Belarusian	false	42665469	24423640	Opera	23	52
496	496	Mack The Knife	Czech	true	98765737	14130091	Country	91	18
497	497	Its not goodbye	Yoruba	true	17534202	8240354	House	56	2
498	498	Its my life	Uyghur	false	70340945	83736945	Hard Rock	74	27
499	499	Strongest	Arabic	true	80017070	40647146	Soul	94	58
500	500	How you remind me	Yue	false	39112602	16589636	House	96	72



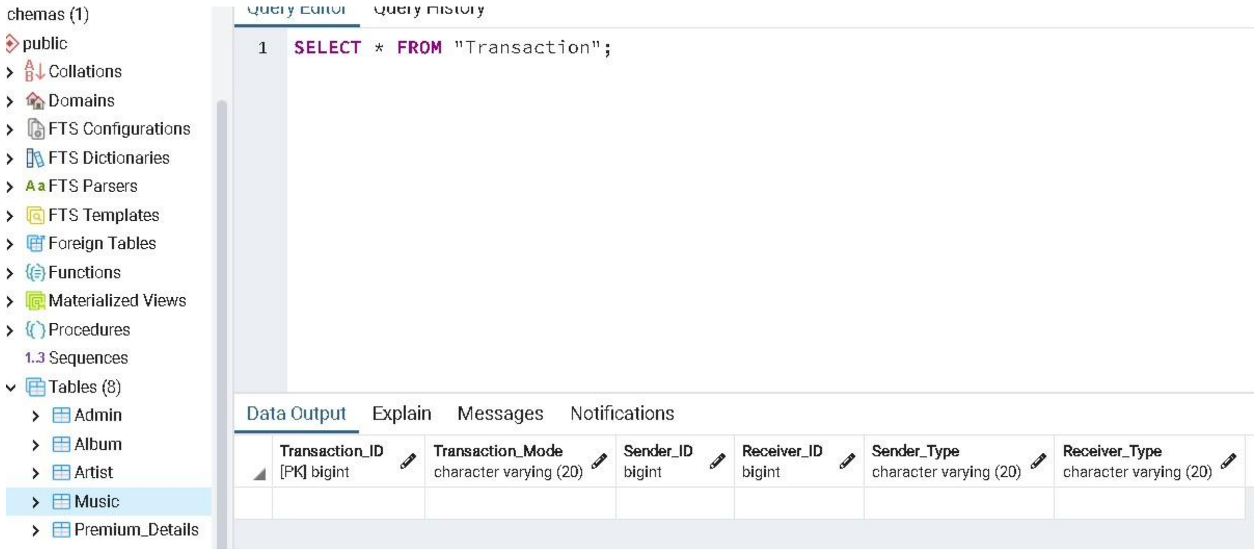
```
SELECT * FROM "Premium_Details";
```

	Plan_ID [PK] bigint	Plan_Name character varying (100)	Plan_Duration integer	Plan_Price integer
5	5	Bumper	144	2000
6	6	BigBumper	180	2200
7	7	Supersell	216	2400
8	8	ExtraSuper	252	2500
9	9	Holiday	288	2600
10	10	Diwalispecial	324	2800
11	11	Supersellpro	360	3000



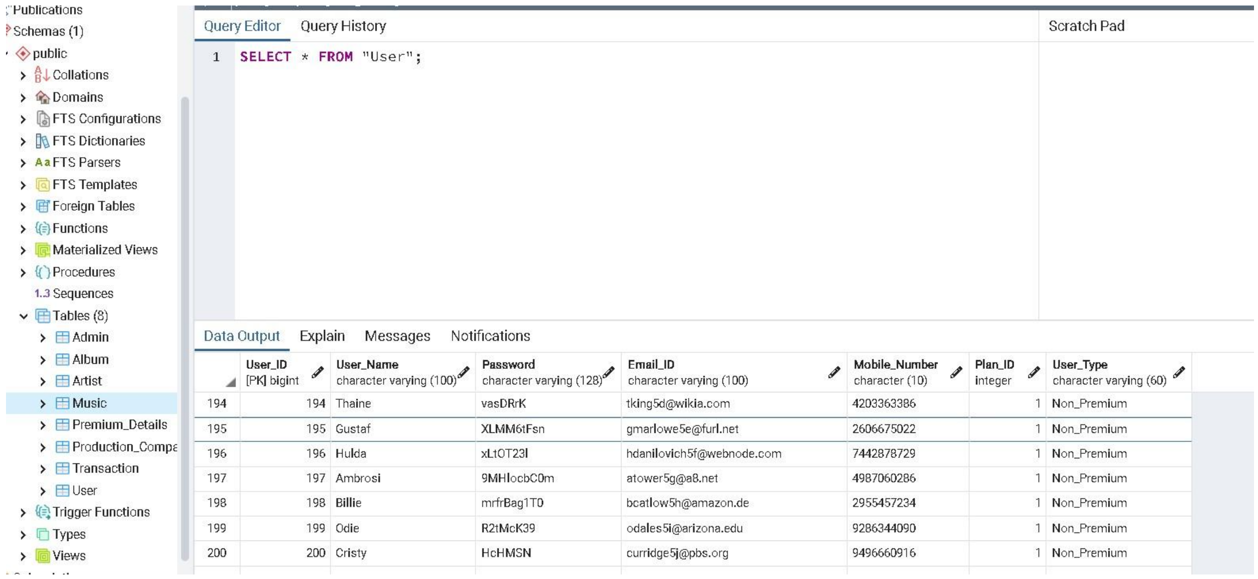
```sql
SELECT * FROM "Production_Company";
```

	Company_ID [PK] bigint	Company_Name character varying (100)	Base_salary bigint
30	30	Kertzmann, Vandervort and Kling	2095294
31	31	Howe-Hills	3390905
32	32	McCullough and Sons	2147122
33	33	Greenholt-Flatley	239376
34	34	Christiansen and Sons	1119352
35	35	Parisian-Crona	361382
36	36	Romaguera Inc	2817158



```sql
SELECT * FROM "Transaction";
```

	Transaction_ID [PK] bigint	Transaction_Mode character varying (20)	Sender_ID bigint	Receiver_ID bigint	Sender_Type character varying (20)	Receiver_Type character varying (20)



```sql
SELECT * FROM "User";
```

	User_ID [PK] bigint	User_Name character varying (100)	Password character varying (128)	Email_ID character varying (100)	Mobile_Number character (10)	Plan_ID integer	User_Type character varying (60)
194	194	Thaine	vasDRrK	tking5d@wikia.com	4203363386	1	Non_Premium
195	195	Gustaf	XLMM6tFsn	gmarlowe5e@furl.net	2606675022	1	Non_Premium
196	196	Hulda	xLtOT23l	hdanilovich5f@webnode.com	7442878729	1	Non_Premium
197	197	Ambrosi	9MHlocbC0m	atower5g@a8.net	4987060286	1	Non_Premium
198	198	Billie	mrfrBag1T0	bcatlow5h@amazon.de	2955457234	1	Non_Premium
199	199	Odie	R2tMcK39	odales5i@arizona.edu	9286344090	1	Non_Premium
200	200	Cristy	HcHMSN	curridge5j@pbs.org	9496660916	1	Non_Premium

In the Transaction table we have no tuples because we have set it as a transaction management system when the user buys premium it gets inserted in that table.