**Федеральное государственное образовательное бюджетное учреждение**

**высшего образования**

**«ФИНАНСОВЫЙ УНИВЕРСИТЕТ ПРИ ПРАВИТЕЛЬСТВЕ РФ»**

**Департамент анализа данных и машинного обучения**

**Отчет по практике №5-6**

по дисциплине «системы управления базами данных»

Студента группы ПМ23-1

Факультета информационных технологий и анализа больших данных

Тищенко И.С.

Преподаватель

Карамышев Е.П.

Москва 2024

# Упражнение 1

Для создания базы данных в pgAdmin, которая будет использоваться для систематизации и прослушивания музыки, можно спроектировать следующие таблицы:

## 1. Таблица Users (Пользователи)

Описание: Содержит информацию о пользователях сайта.

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип данных | Описание | Ограничения |
| user\_phone | VARCHAR(11) | Номер телефона | PRIMARY KEY, NOT NULL |
| username | VARCHAR(50) | Имя пользователя | UNIQUE, NOT NULL |
| email | VARCHAR(100) | Электронная почта | UNIQUE, NOT NULL |
| password | VARCHAR(255) | Хэшированный пароль | NOT NULL |
| registration\_date | DATE | Дата регистрации | DEFAULT CURRENT\_DATE |

## 2. Таблица Artists (Исполнители)

Описание: Содержит информацию о музыкальных исполнителях.

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип данных | Описание | Ограничения |
| artist\_email | VARCHAR(50) | Электронная почта | PRIMARY KEY, NOT NULL |
| artist\_name | VARCHAR(100) | Имя исполнителя | UNIQUE, NOT NULL |
| country | VARCHAR(50) | Страна происхождения | NOT NULL |
| birth\_date | DATE | Дата рождения исполнителя |  |
| genre | VARCHAR(50) | Жанр | CHECK (genre IN ('Pop', 'Rock', 'Jazz', 'Classical', 'Electronic', 'Hip-Hop')) |

## 3. Таблица Albums (Альбомы)

Описание: Содержит информацию об альбомах, выпущенных исполнителями.

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип данных | Описание | Ограничения |
| album\_id | SERIAL | Уникальный идентификатор | PRIMARY KEY, NOT NULL |
| album\_name | VARCHAR(100) | Название альбома | NOT NULL |
| release\_date | DATE | Дата выхода альбома | NOT NULL |
| artist\_id | INT | Ссылка на исполнителя | FOREIGN KEY (artist\_id) REFERENCES Artists(artist\_id) ON DELETE CASCADE ON UPDATE CASCADE |
| total\_tracks | INT | Количество треков | CHECK (total\_tracks > 0), NOT NULL |

## 4. Таблица Songs(Песни)

Описание: Содержит информацию о песнях.

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип данных | Описание | Ограничения |
| song\_id | SERIAL | Уникальный идентификатор | PRIMARY KEY, NOT NULL |
| song\_name | VARCHAR(100) | Название песни | NOT NULL |
| album\_id | INT | Ссылка на альбом | FOREIGN KEY (album\_id) REFERENCES Albums(album\_id) ON DELETE CASCADE ON UPDATE CASCADE |
| duration | INTERVAL | Длительность песни | NOT NULL |
| plays\_count | INT | Количество прослушиваний | CHECK (plays\_count >= 0) |

## 5. Таблица Playlists (Плейлисты)

Описание: Содержит информацию о пользовательских плейлистах.

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип данных | Описание | Ограничения |
| playlist\_id | SERIAL | Уникальный идентификатор | PRIMARY KEY, NOT NULL |
| playlist\_name | VARCHAR(100) | Название плейлиста | NOT NULL |
| user\_id | INT | Ссылка на пользователя | FOREIGN KEY (user\_id) REFERENCES Users(user\_id) ON DELETE CASCADE ON UPDATE CASCADE |
| created\_date | DATE | Дата создания плейлиста | DEFAULT CURRENT\_DATE |
| is\_public | BOOLEAN | Статус публичности | DEFAULT FALSE, NOT NULL |

## 6. Таблица для связи плейлистов и песен: Playlist\_Songs

|  |  |  |  |
| --- | --- | --- | --- |
| Атрибут | Тип данных | Описание | Ограничения |
| playlist\_id | INT | Ссылка на плейлист | FOREIGN KEY (playlist\_id) REFERENCES Playlists(playlist\_id) ON DELETE CASCADE ON UPDATE CASCADE |
| song\_id | INT | Ссылка на песню | FOREIGN KEY (song\_id) REFERENCES Songs(song\_id) ON DELETE CASCADE ON UPDATE CASCADE |

## Итоговые ограничения:

1. PRIMARY KEY для каждой таблицы.

2. Внешние ключи для таблиц Albums, Songs, Playlists, Playlist\_Songs.

3. CHECK для жанра, длительности и количества треков.

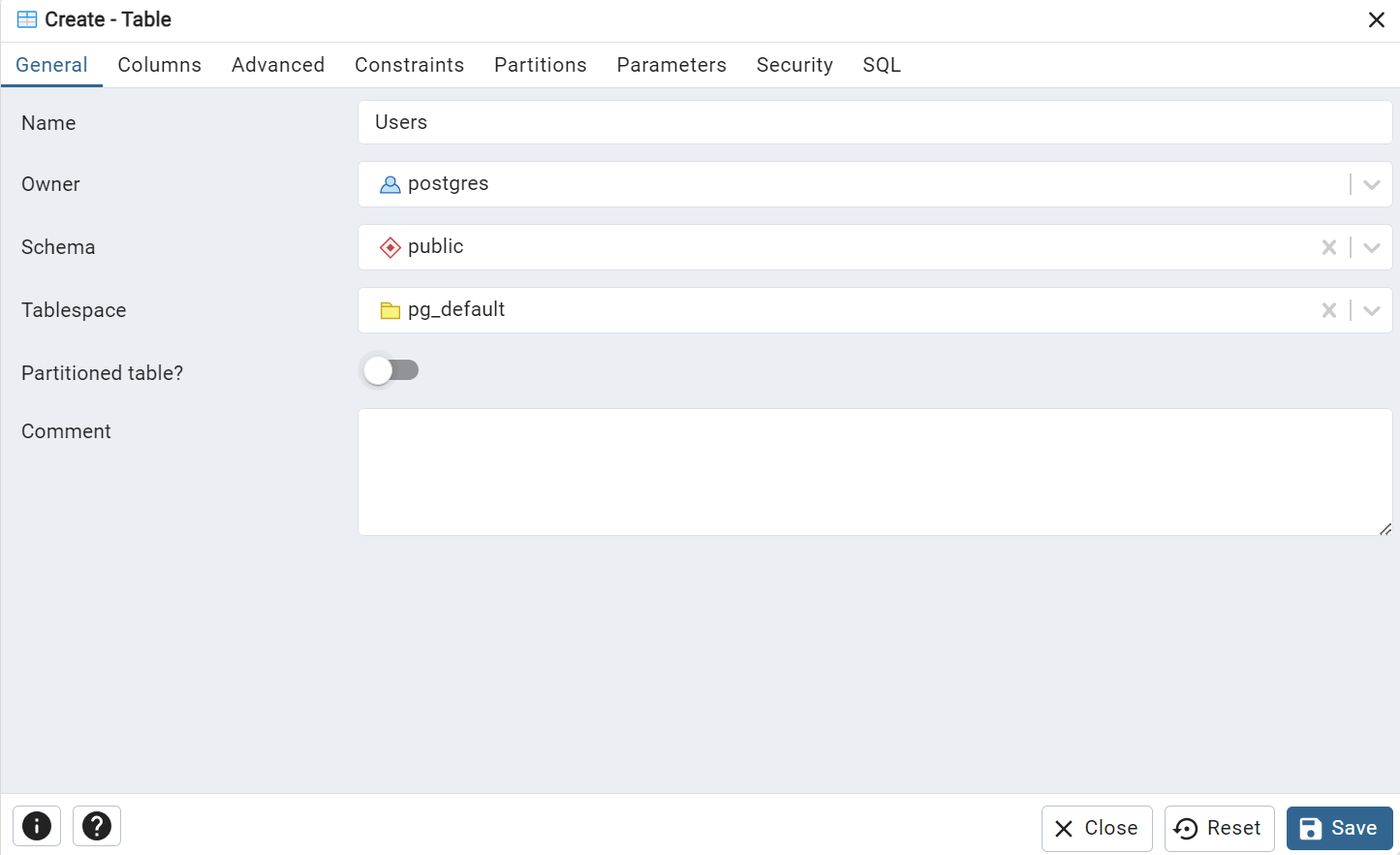
4. Уникальные значения для username, email, artist\_name.

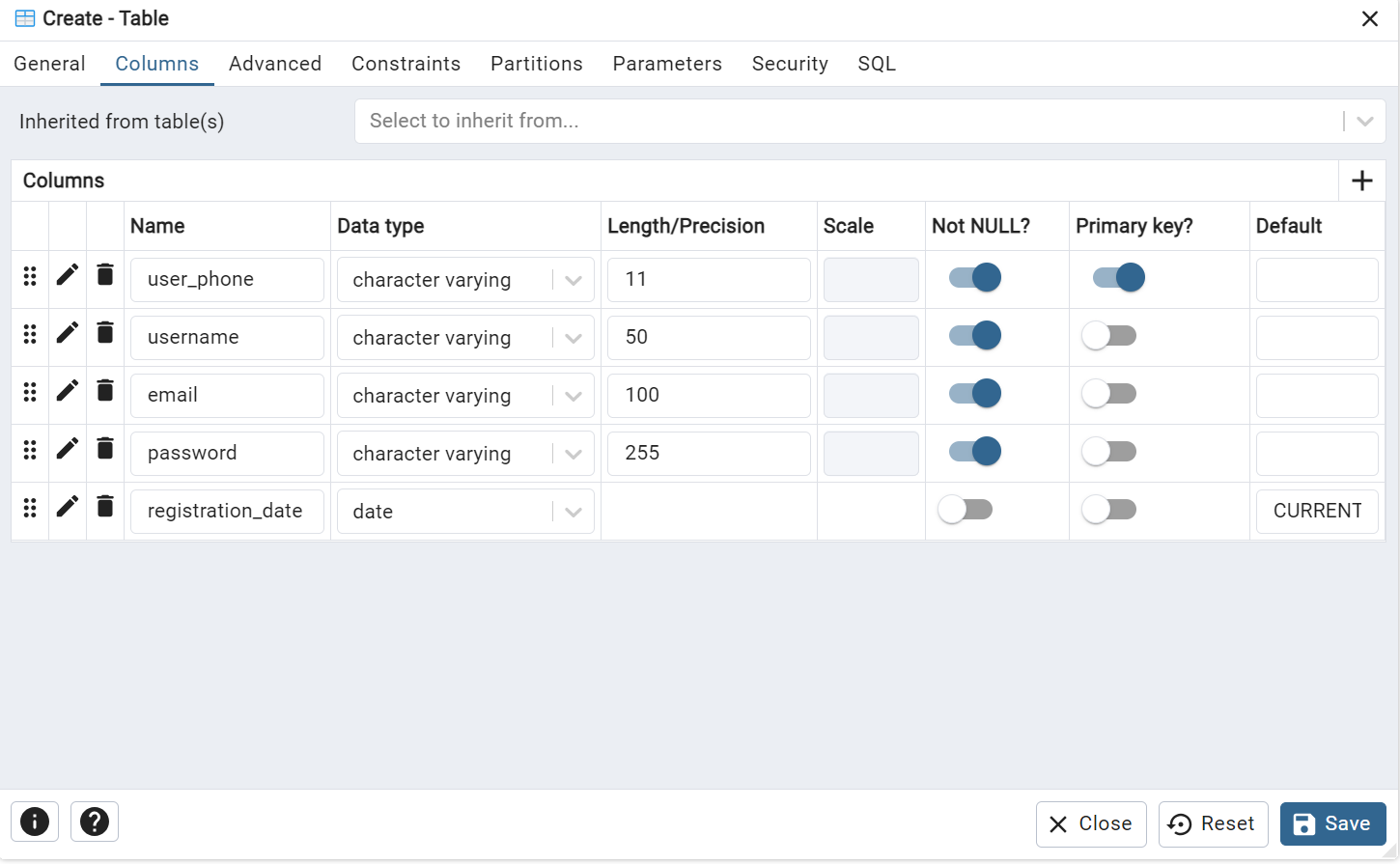
5. DEFAULT для полей registration\_date, created\_date, is\_public.

6. RESTRICT/CASCADE для внешних ключей.

## На базе pgAdmin 4 создадим 6 таблиц.

Покажем, как это происходит для 1-й таблицы.

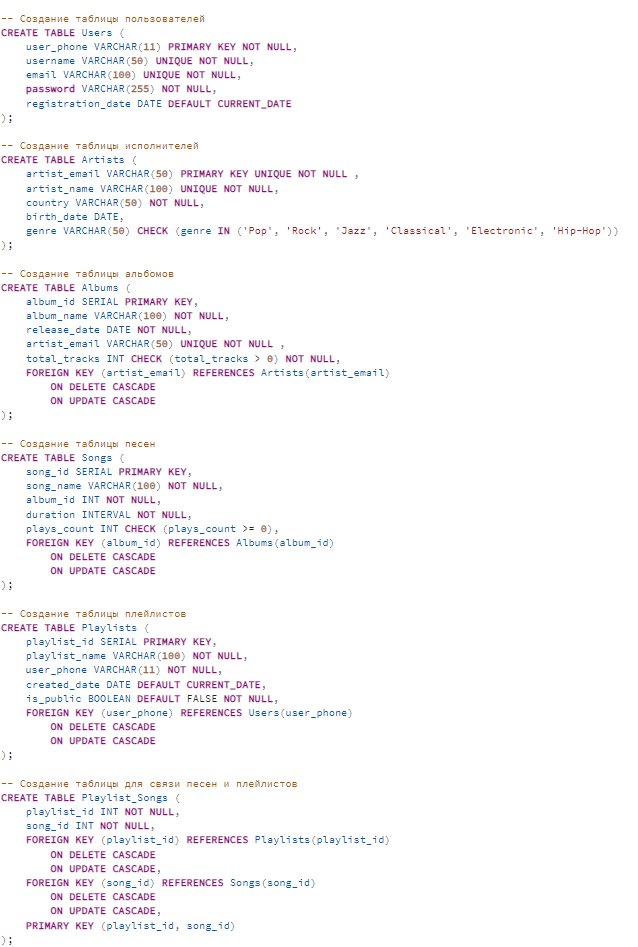




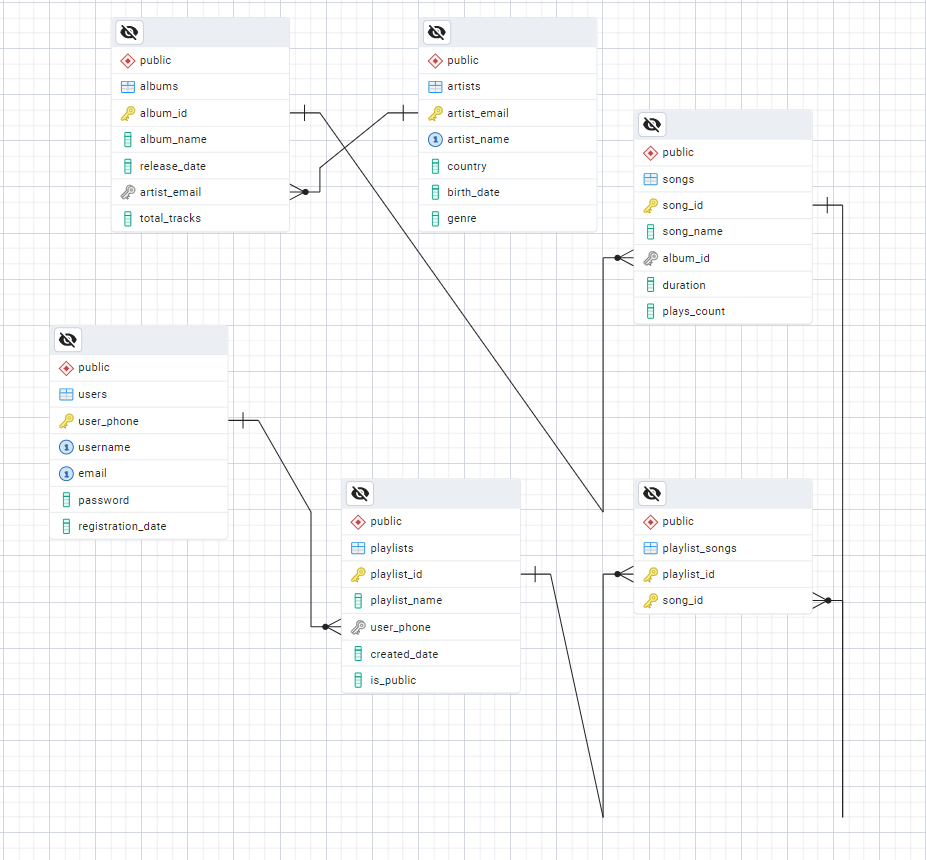
Автоматически созданный код для создания 1-й таблицы.



## Автоматический код для создания 6 таблиц.



## Построим схему для этой БД.

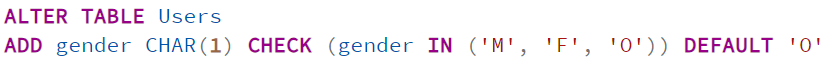


# Упражнение 2

Разработать запросы для изменения структуры БД, предварительно сохранив копию исходной БД:

## а. добавление/удаление/модификация столбцов и ограничений в существующие таблицы (модифицировать не менее 3 таблиц).

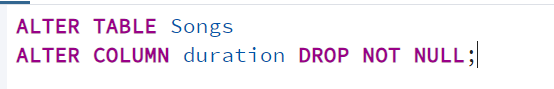
Добавление нового столбца "gender" в таблицу Users



Модификация столбца "artist\_name" в таблице Artists, изменяем его длину до 150 символов

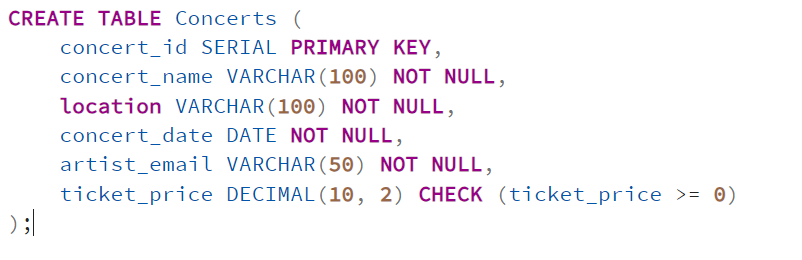


Удаление ограничения NOT NULL у столбца "duration" в таблице Songs



## б. создание дополнительной таблицы;

Создадим новую таблицу "Concerts" для хранения информации о концертах



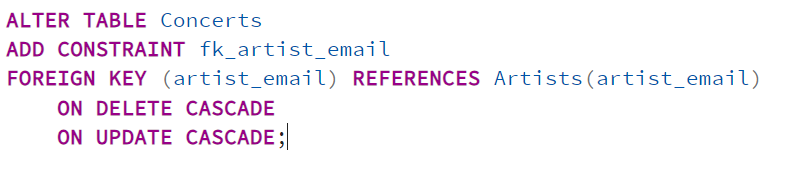
## в. добавление столбца в дополнительную таблицу;

Добавление столбца "available\_tickets" в таблицу Concerts



## г. создание ограничения внешнего ключа для связи новой таблицы с существующими;

Связываем "Concerts" с таблицей "Artists" через "artist\_email"



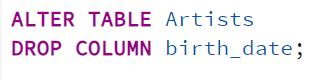
## д. модификация столбца в таблице;

Изменение типа данных столбца "plays\_count" в таблице Songs на BIGINT для увеличения диапазона



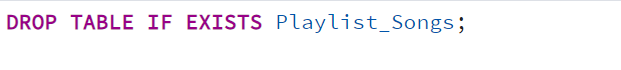
## е. удаление столбца из таблицы;

Удаление столбца "birth\_date" из таблицы Artists



## ж. удаления таблицы.

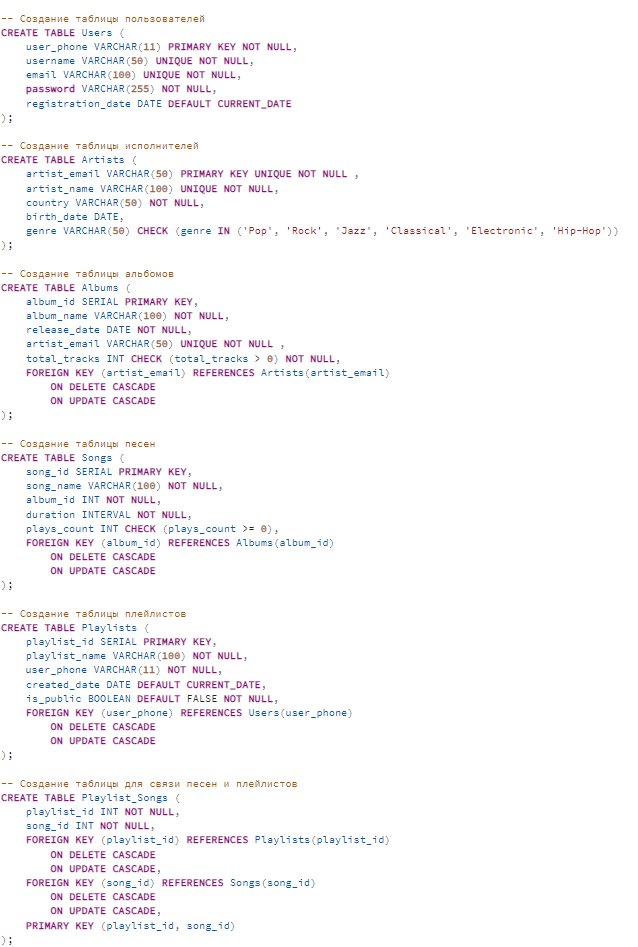
Удаление таблицы Playlist\_Songs



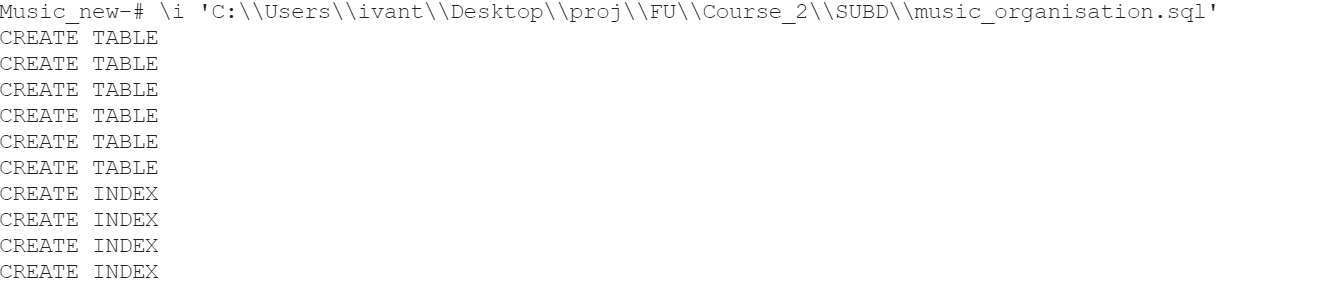
# Упражнение 3

## 2. Напишите общий скрипт для создания первоначальной структуры базы данных, разработанной на предыдущем практическом занятии.

Вспомним код для создания первоначальной БД.

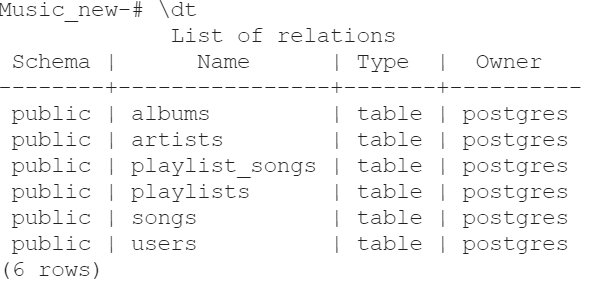


## 3. Запустите скрипт с помощью psql (psql \i <путь к файлу.sql>), предварительно переименовав созданную ранее БД, убедитесь в его работоспособности.



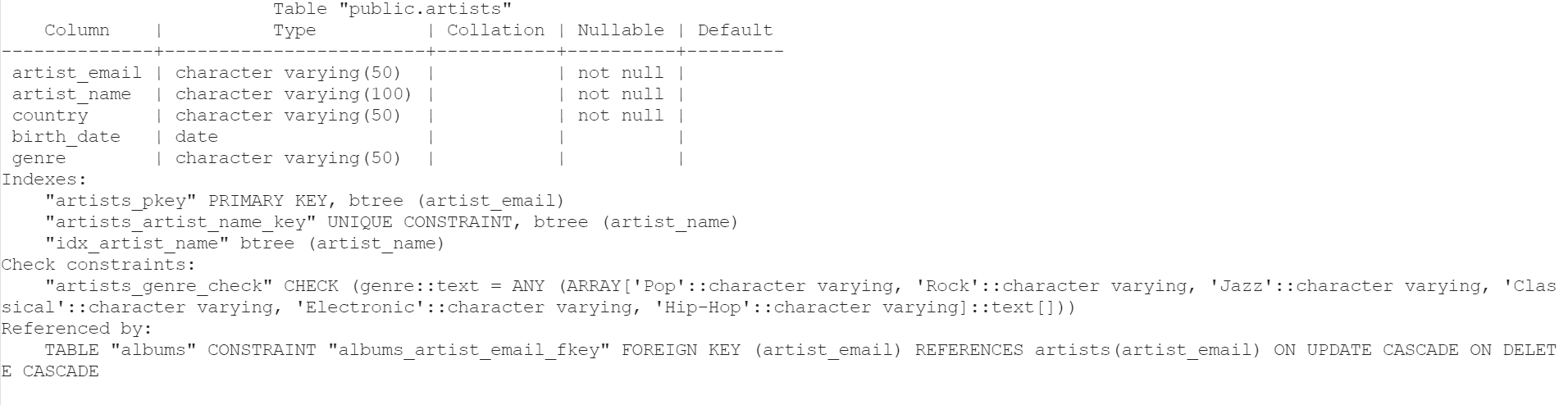
## 4. Создайте описание базы данных, включающее перечисление ее таблиц, для каждой таблицы – характеристики столбцов, используя команды psql: \dt, \d.

### \dt

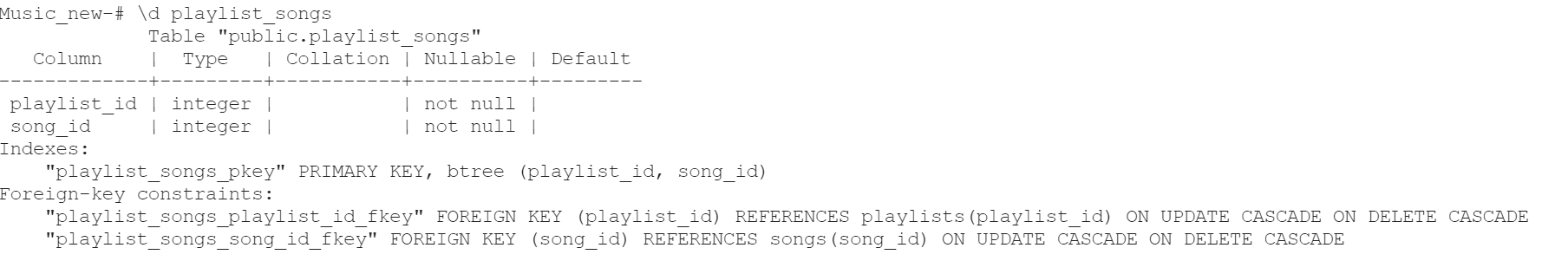


### \d albums

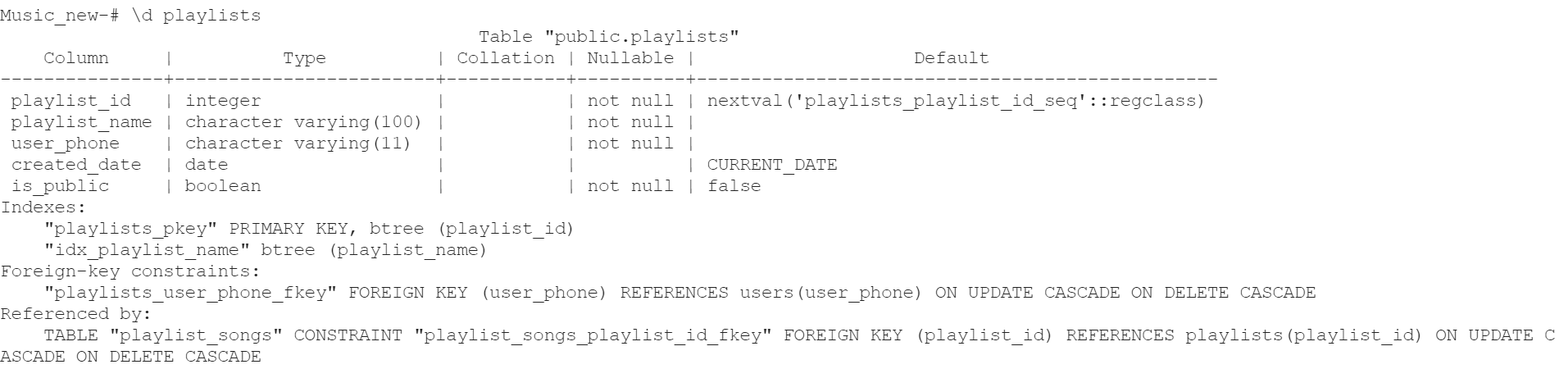
### \d artists



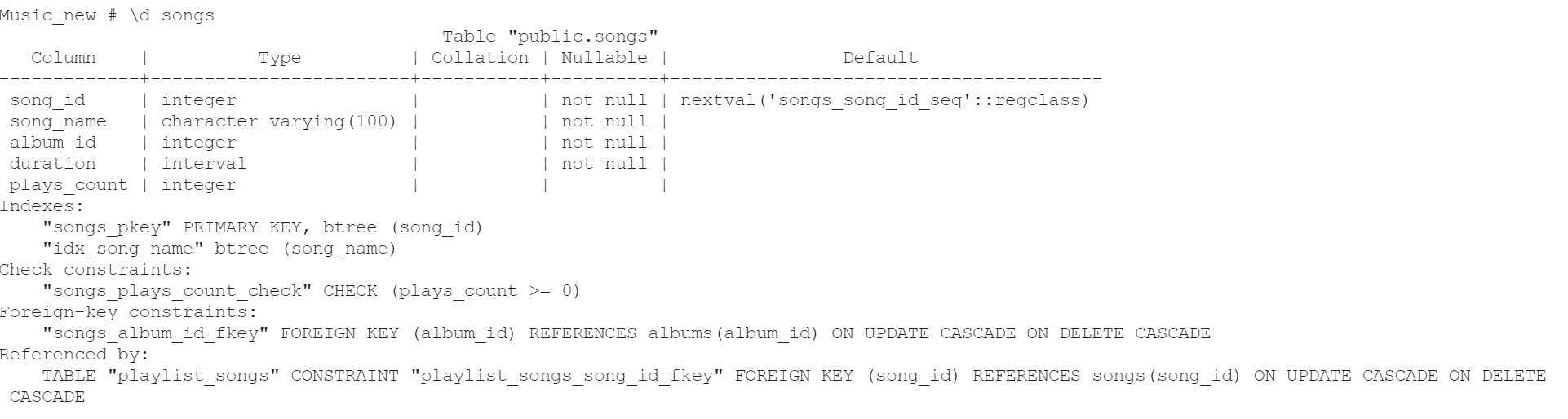
### \d playlist\_songs



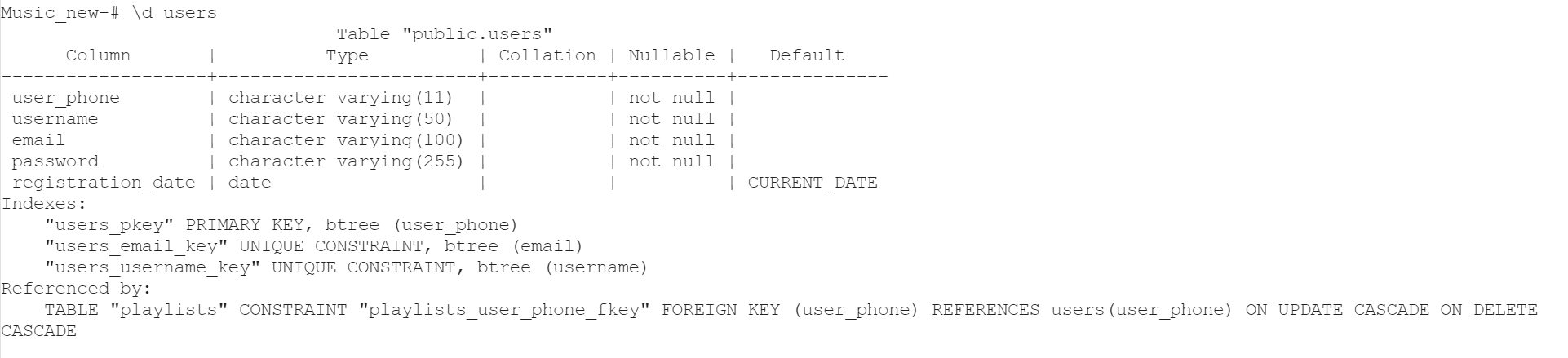
### \d playlists



### \d songs

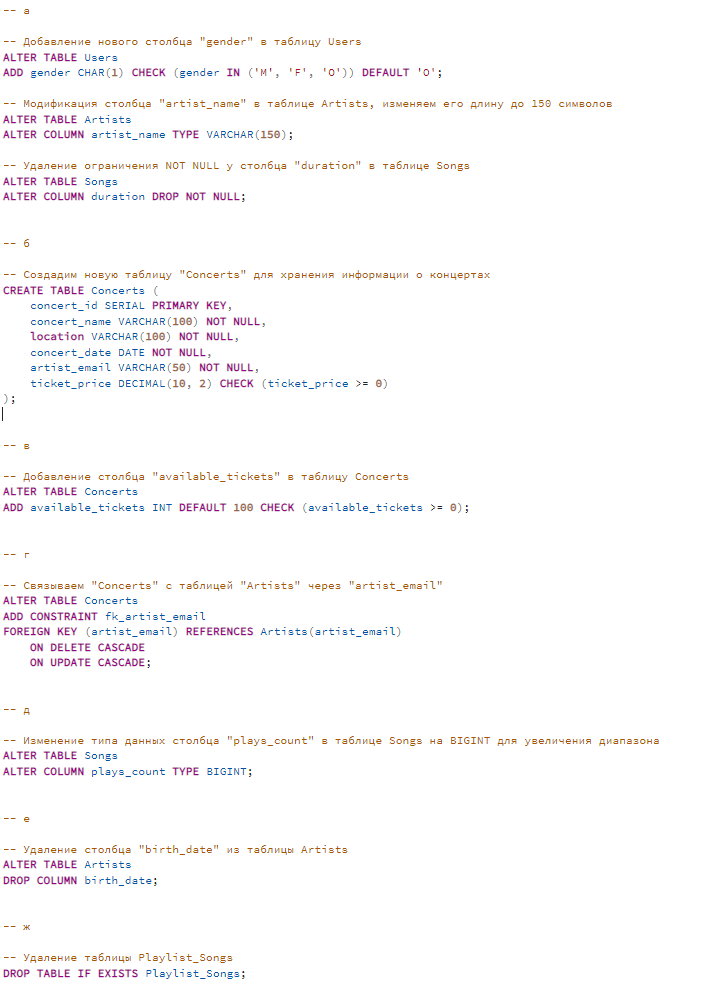


### \d users

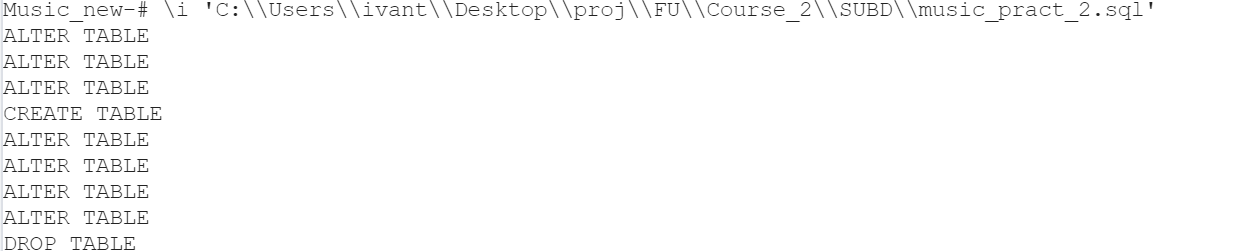


## 5. Выполните все скрипты для модификации БД, написанные на предыдущем практическом занятии, в psql.

Вспомним код из прошлого практического задания:

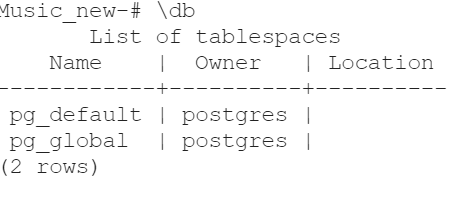


Запишем код в файл music\_pract\_2.sql и запустим из psql

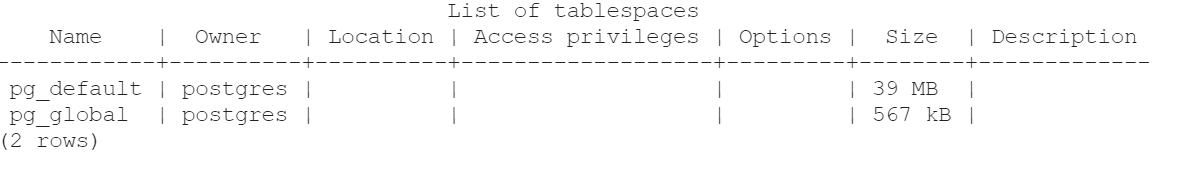


## 6. Посмотрите список табличных пространств с помощью psql кратко и подробно.

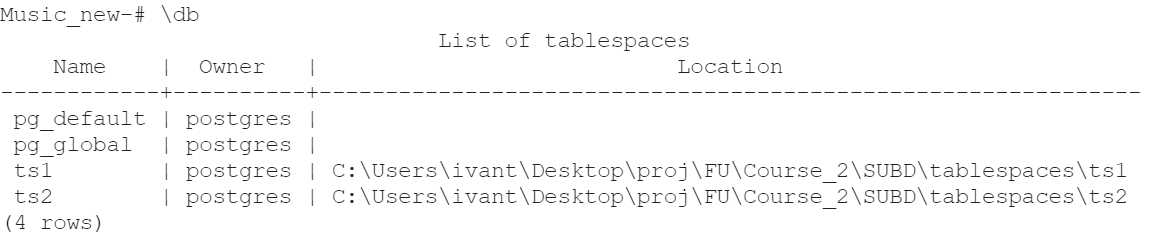
### \db

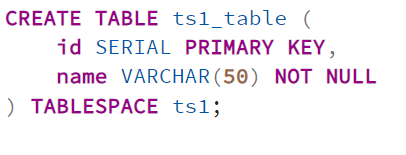


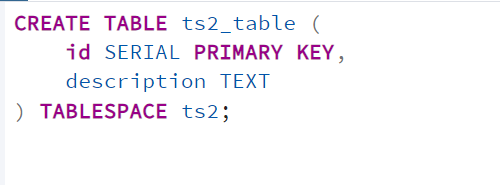
### \db+



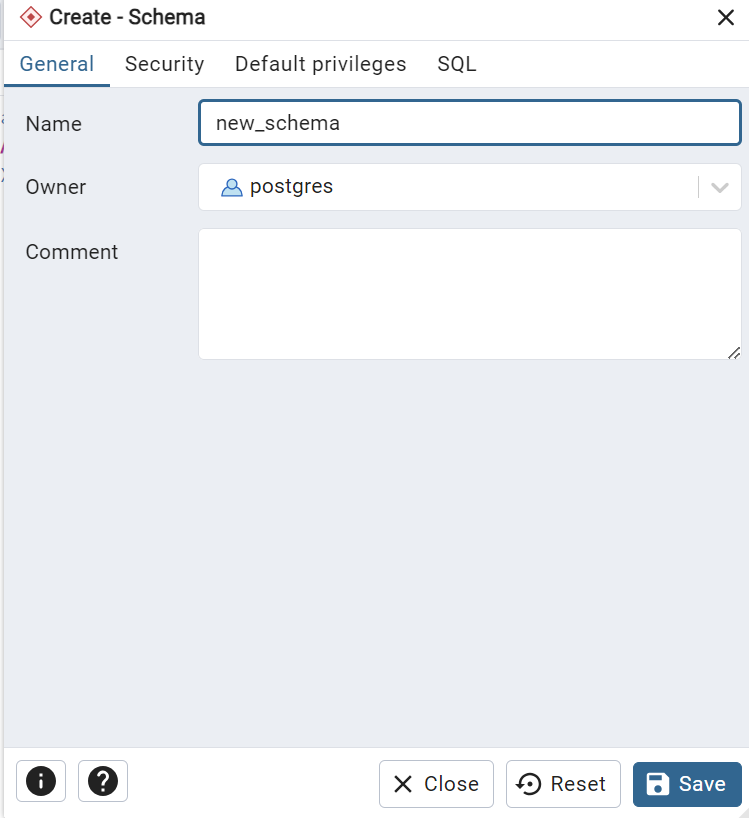
## 7. Добавьте с помощью pgAdmin 2 новых табличных пространства, создайте дополнительно несколько таблиц в новых табличных пространствах.





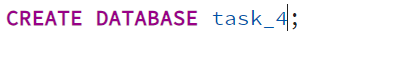


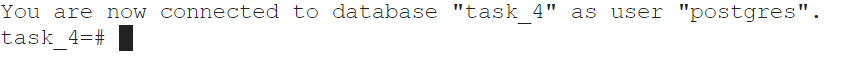
## 8. Добавьте с помощью pgAdmin новую схему в БД.



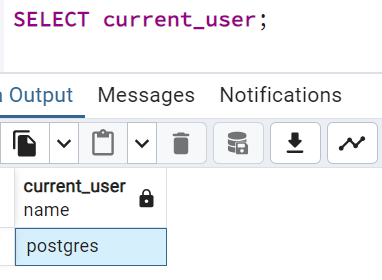
# Упражнение 4

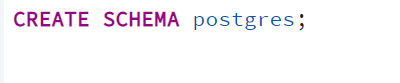
## Создайте новую базу данных и подключитесь к ней.





## Создайте схему, названную так же, как и пользователь.

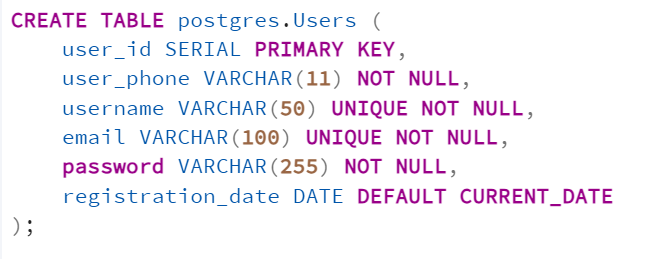


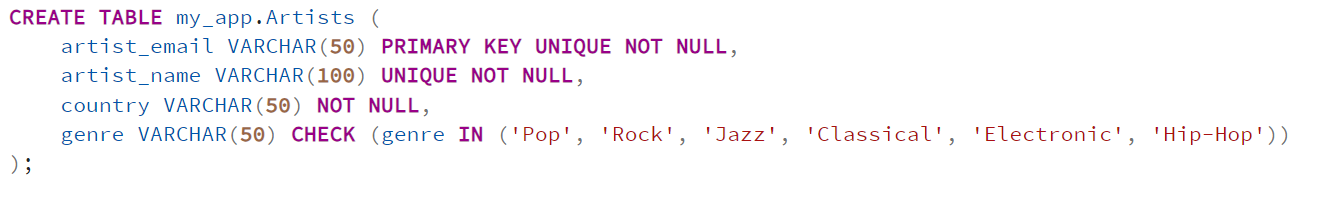


## Создайте схему my\_app.

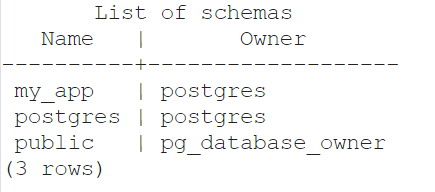


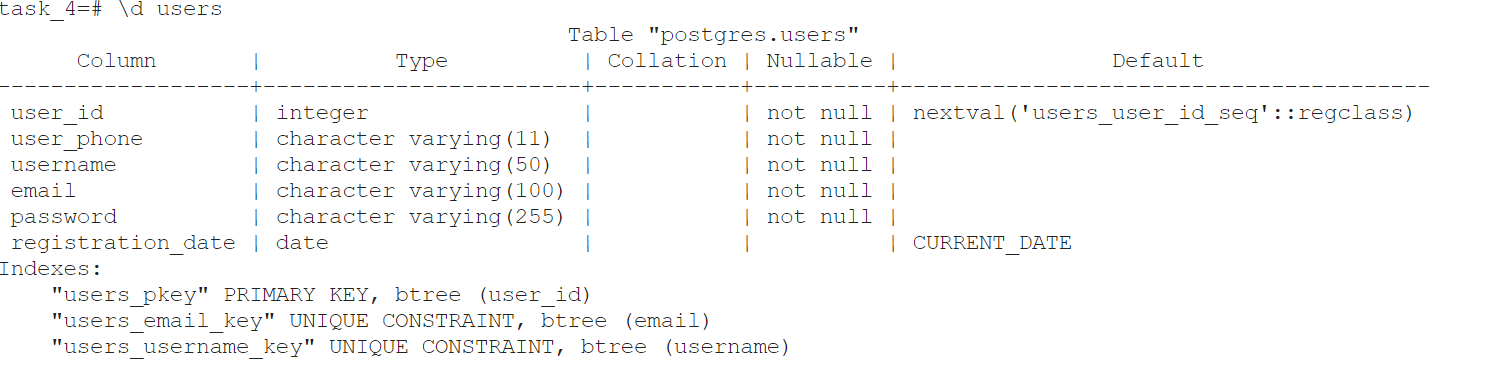
## Создайте несколько таблиц в обоих схемах, соответствующих БД, созданной в предыдущей практической работе.

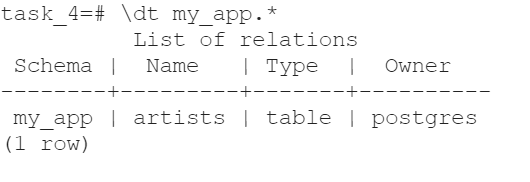


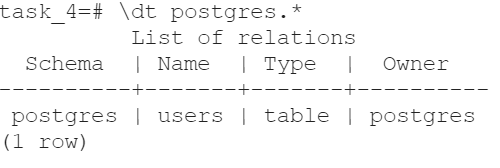


## Получите в psql описание созданных схем и список всех таблиц в них.









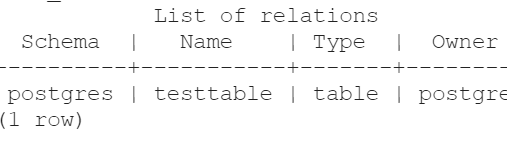
## Установите путь поиска так, чтобы при подключении к БД таблицы из обоих схем были доступны по неквалифицированному имени; приоритет должна иметь «пользовательская» схема.



## Проверьте правильность настройки.

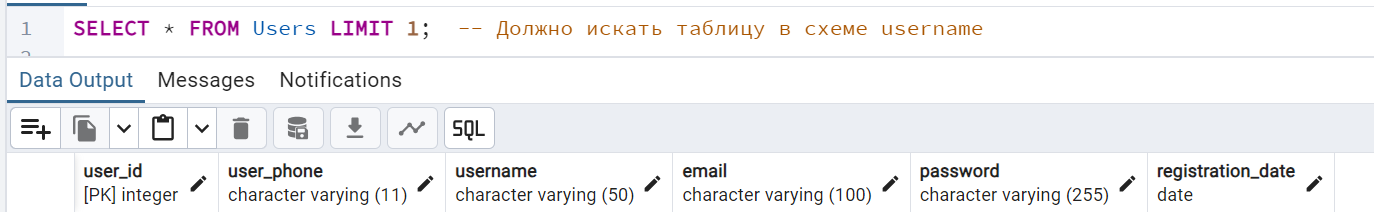
Создадим временную таблицу, чтобы удостовериться в том, что psql использует пользовательскую схему.





Проверка пройдена успешно.

Попробуем сделать пару запросов.





Все работает правильно