

# ОПИСАНИЕ СРЕДЫ РАЗРАБОТКИ И СХЕМЫ БАЗЫ ДАННЫХ

## Визуальная среда разработки DBeaver

DBeaver - это бесплатный и многоплатформенный инструмент для работы с базами данных, который предоставляет разработчикам баз данных удобный способ выполнения следующих задач:

- создание подключения к базе данных;
- просмотр и управление объектами базы данных;
- ввод, отладка и выполнение SQL-операторов;
- экспорт и импорт данных в нужных форматах;
- осуществлять резервное копирования и восстановление баз данных

DBeaver можно использовать для управления любой базой данных, имеющей драйверы ODBC или JDBC. Этот инструмент полезен для администраторов баз данных, разработчиков, программистов SQL и аналитиков. После запуска DBeaver на экране появляется главное окно, представленное на рис.1.1.

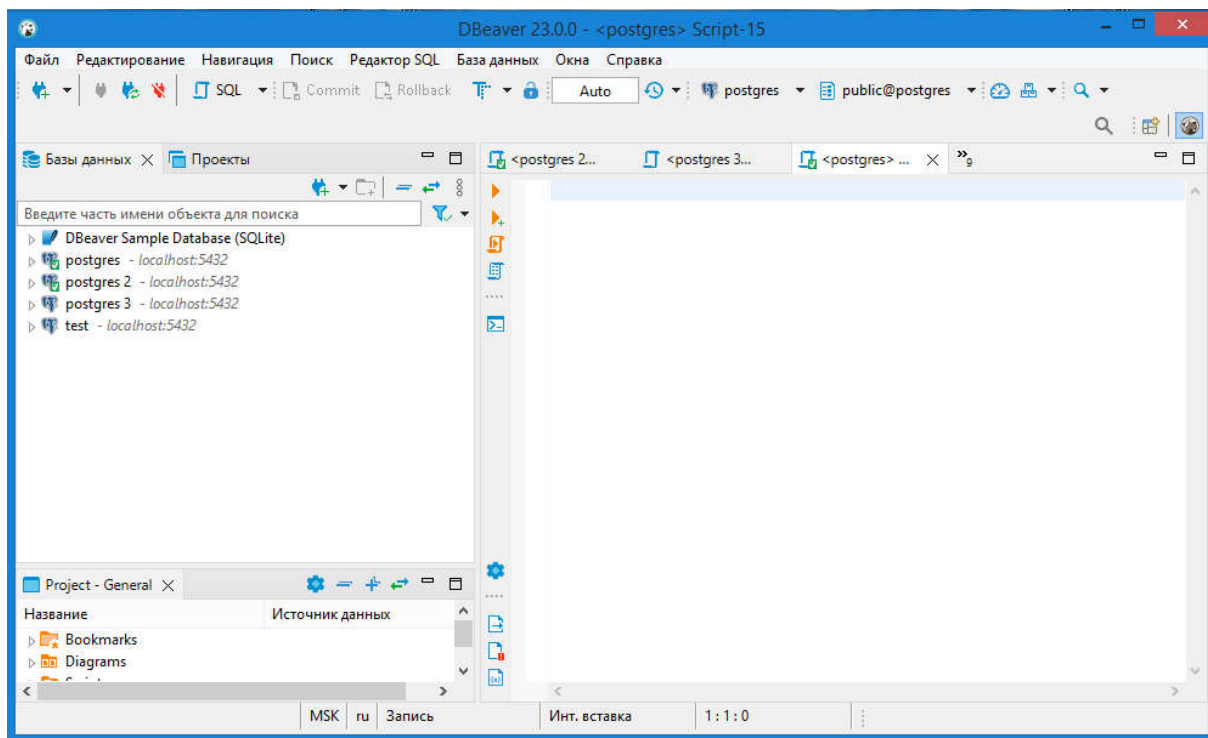


Рис. 1.1. Главное окно DBeaver

## 1.1. Главное окно DBeaver

Главное окно DBeaver содержит строку меню, панель инструментов, окно навигатора, окно проектов и рабочую область.

### 1.1.1. Меню

Строка меню содержит следующие элементы:

- **Файл** – содержит пункты меню для создания файлов, папок, проектов, подключений к базе данных, проектов баз данных и диаграмм ER, а также элементов импорта и экспорта.
- **Редактирование** – содержит основные команды, предназначенные для активного элемента.
- **Навигация** – позволяет перемещаться по скриптам и объектам базы данных.
- **Поиск** – предоставляет опции для поиска среди файлов, объектов базы данных и по данным.
- **Редактор SQL** – используется для открытия редактора запросов SQL и скриптов.
- **База данных** – позволяет управлять драйверами базы данных, подключениями и транзакциями, подключаться к базе данных и отключаться от нее.
- **Окна** – включает в себя пункты для управления внешним видом окна DBeaver.
- **Справка** – содержит ссылки на информационные и справочные ресурсы.

### 1.1.2. Панель инструментов

Панель инструментов (рис. 1.2.) содержит кнопки для исполнения основных и часто используемых команд.

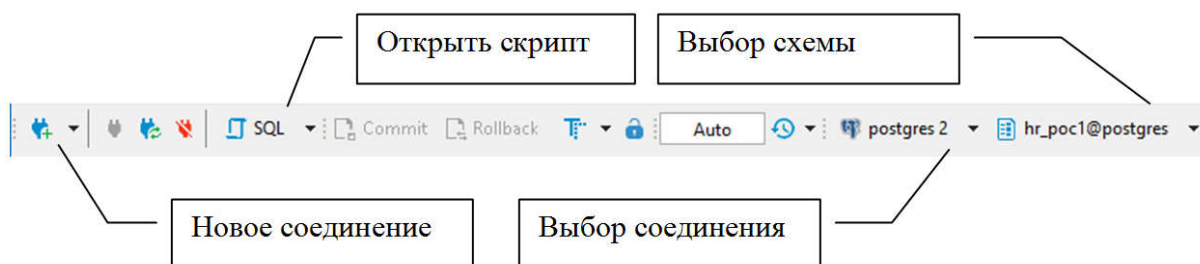


Рис. 1.2. Панель инструментов DBeaver

Некоторые кнопки включены (выделены цветом), другие отключены (серые). Наборы включенных и отключенных кнопок меняются в зависимости от того, какой редактор в данный момент активен в рабочей области. Только включенные кнопки применимы к активному виду или редактору.

Рассмотрим назначение наиболее важных и часто используемых кнопок:

- **Новое соединение** – позволяет создать новое соединение с базой данных.
- **Открыть скрипт** – предназначена для открытия ранее созданного скрипта или создания нового окна в рабочей области для ввода скрипта.
- **Выбор соединения** – предназначена для выбора соединения и базы данных, с которыми мы будем работать.
- **Выбор схемы** – позволяет выбрать схему, для которой будут выполнены команды, содержащиеся в скрипте.

### ***1.1.3. Рабочая область***

В рабочей области может быть одновременно открыто несколько представлений и редакторов, но активным может быть только один из них.

**Представления** – это окна в рабочей области, которые используются для отображения объектов базы данных.

**Редакторы** – это окна, в которых можно редактировать объекты базы данных, создавать, редактировать и выполнять SQL запросы, процедуры, функции, триггеры.

### ***1.1.4. Окно проектов***

В окне проектов отображаются все проекты, созданные в системе, и предоставляются инструменты для управления ими. Обычно это файлы, хранящиеся в файловой системе.

## **1.2. Подключение к базе данных**

Чтобы иметь возможность управлять базой данных, используя DBeaver, необходимо создать подключение к этой базе данных. Подключение включает в

себя драйвер и ряд параметров конфигурации, включая расположение базы данных и учетные данные для доступа к ней.

Для создания подключения нужно щелкнуть по кнопке **Новое соединение** на панели инструментов DBeaver (рис.1.2), и в появившемся окне рис.1.3, выбрать СУБД, которую вы собираетесь использовать. Мы будем использовать СУБД PostgreSQL, поэтому нужно выбрать эту СУБД и нажать кнопку **Далее**. На экране появится окно для ввода параметров соединения рис.1.4.

При установке СУБД PostgreSQL автоматически создается база данных postgres и пользователь postgres, который является владельцем этой базы данных и обладает правами администратора, также, при установке, требуется ввести пароль, который будет являться паролем пользователя postgres. Эти данные нужно ввести в окне для ввода параметров соединения рис.1.4.

Для проверки правильности ввода этих данных рекомендуется нажать кнопку **Тест соединения**. Если все в порядке то на экране должно появиться окно, представленное на рис. 1.5. В этом случае, для завершения создания соединения, следует нажать кнопку **Готово** рис.1.4.

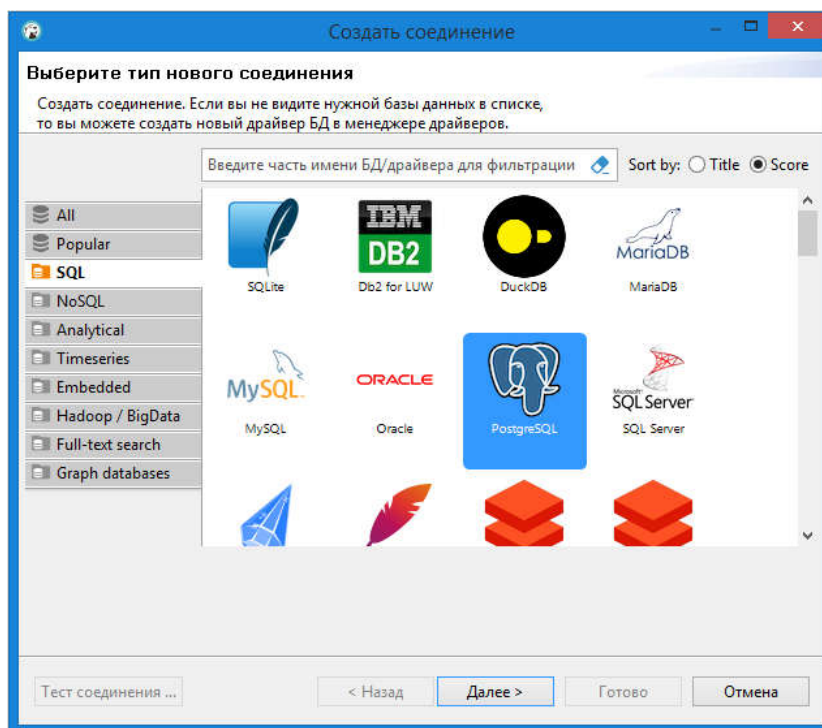


Рис.1.3. Выбор СУБД

Рис.1.4. Окно для ввода параметров соединения

Рис. 1.5. Тест соединения

### 1.3. Загрузка схемы базы данных из резервной копии

Для того чтобы загрузить в базу данных `postges` схему `hr_roc`, с которой мы будем работать, следует в навигаторе выбрать базу данных `postges`, щелкнуть

правой кнопкой и в появившемся контекстном меню рис.1.6, выбрать команду **Инструменты** и в следующем меню выбрать команду **Восстановить**. На экране появится окно **Настройка восстановления** рис.1.7. В этом окне следует выбрать файл, который содержит резервную копию базы данных и нажать кнопку **Готово**.

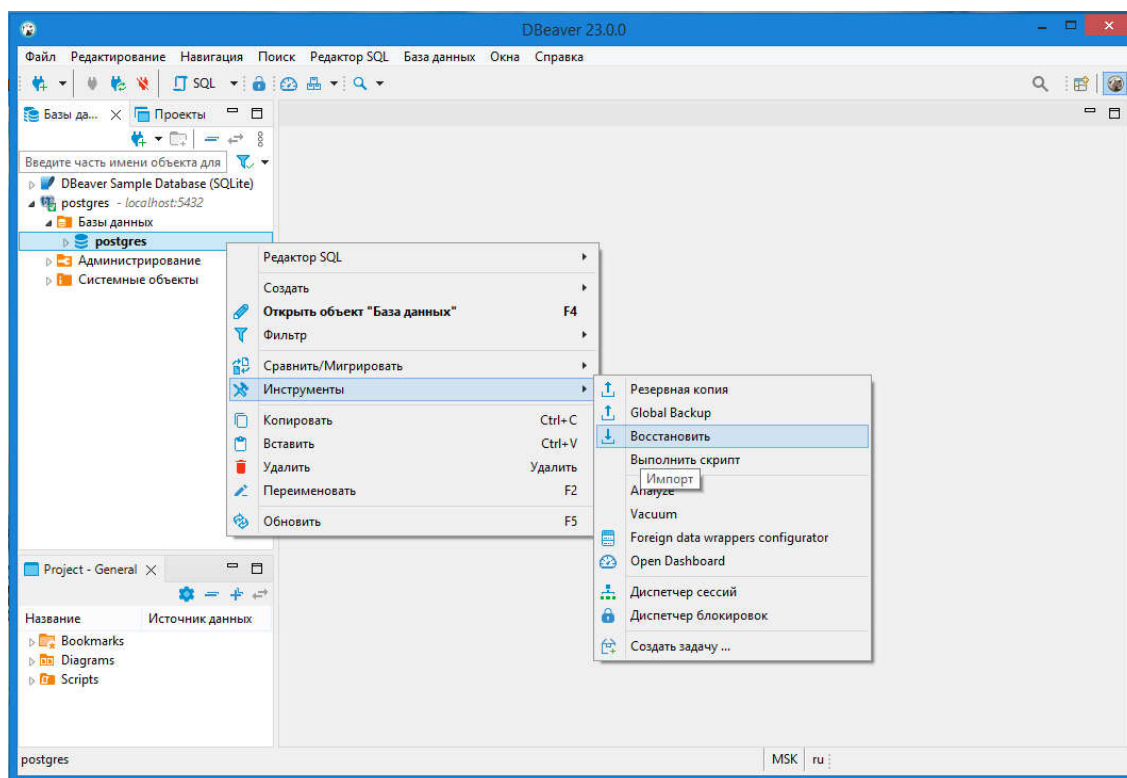


Рис.1.6. Загрузка данных из резервной копии.

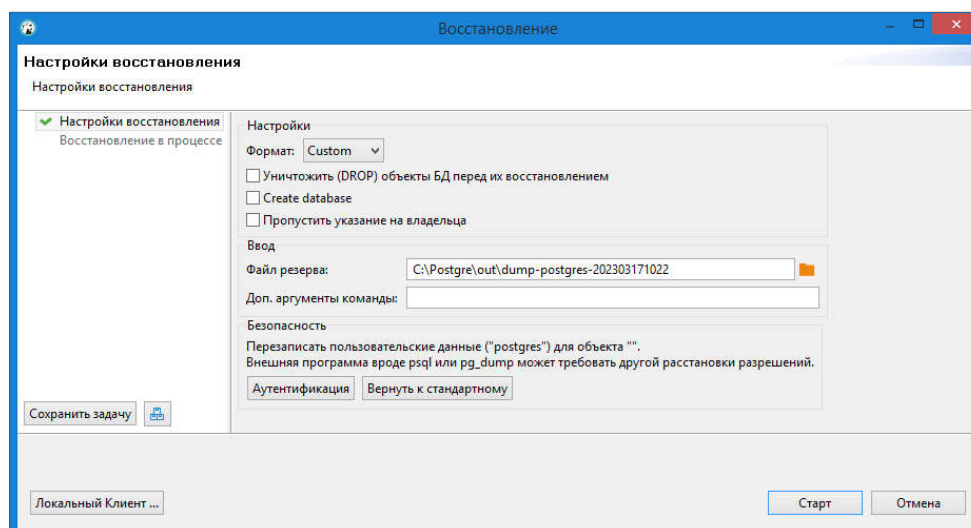


Рис. 1.7. Выбор файла резервной копии.

В результате этих действий все таблицы схемы hr\_roc будут извлечены из резервной копии и загружены в базу данных postgres. Схему hr\_roc нужно  
 МАИ, Ткачев О.А. Методические указания к лабораторным работам по дисциплине «Базы данных».

переименовать. Для этого следует щелкнуть правой кнопкой на имени схемы, в контекстном меню рис.1.8 выбрать команду **Переименовать**.

Новое имя: **bXXX\_YY**, где XXX – номер группы, YY – порядковый номер студента в журнале.

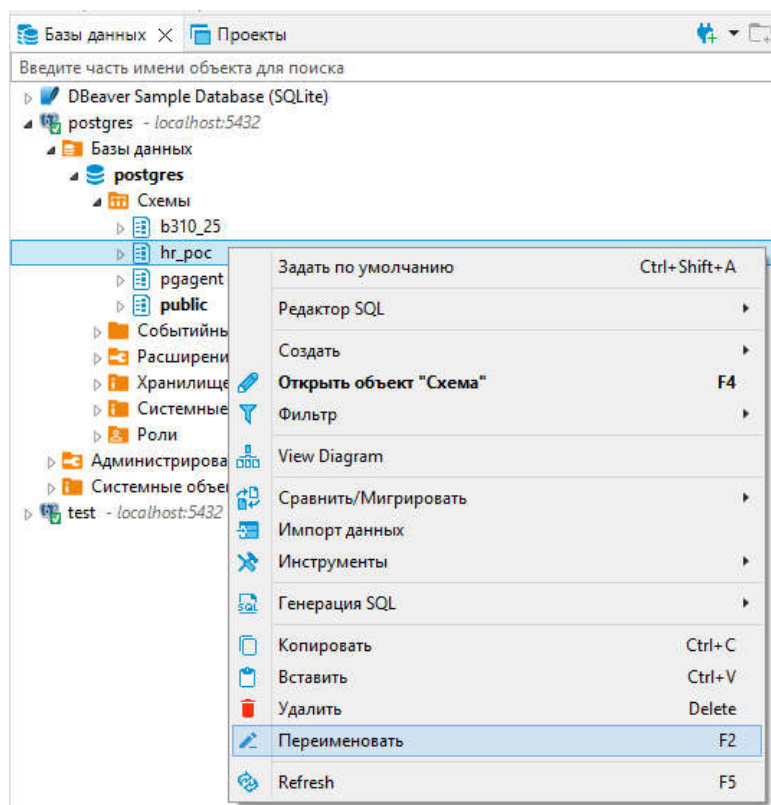


Рис.1.8. Переименование схемы hr\_pos1

#### 1.4. Навигатор базы данных

Навигатор базы данных предназначен для работы со структурой и содержимым баз данных. Внешний вид окна навигатора показан на рис.1.9.



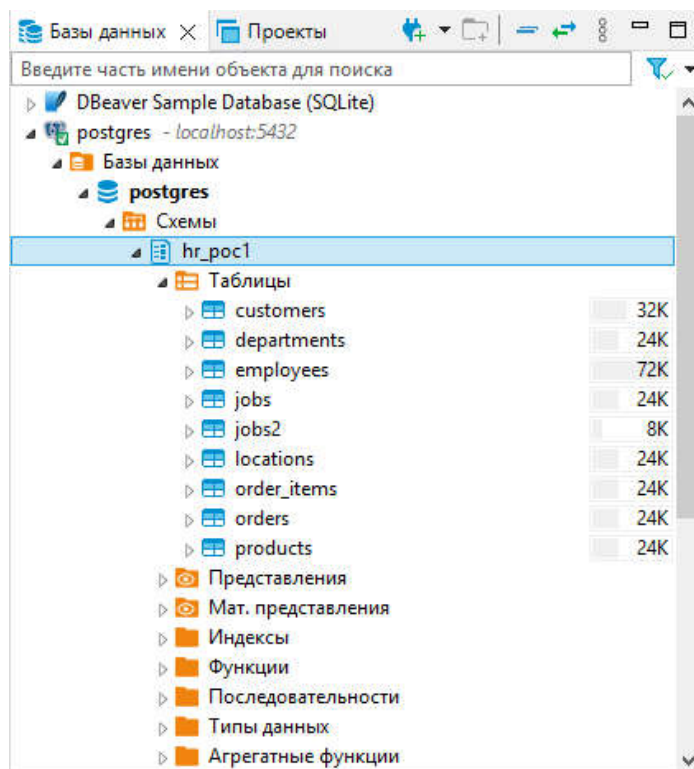




Рис. 1.9. Окно навигатора базы данных

Навигатор базы данных содержит дерево объектов, панель инструментов и меню просмотра. Каждый объект в дереве имеет свое собственное контекстное меню. Дерево содержит следующие объекты:

- Папки - .
- Подключения к базе данных - .
- Объекты базы данных - таблицы, представления, столбцы и т.д.

Если сделать двойной щелчок на имени таблицы, то откроется окно редактора для работы с этой таблицей. В этом окне можно посмотреть и отредактировать свойства имеющихся столбцов, удалить и добавить столбцы. На рис.1.10 показано окно редактора для таблицы customers.



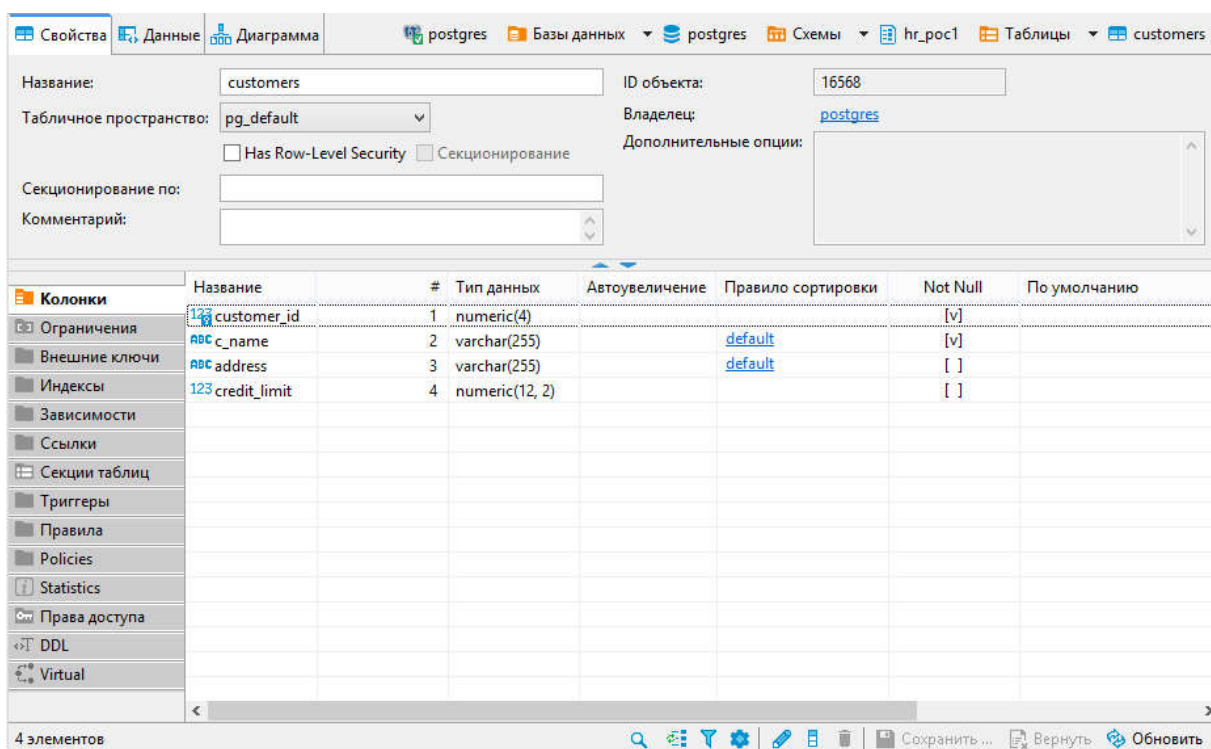


Рис.1.10. Свойства таблицы customers

Если выбрать вкладку **Данные**, в верхней части этого окна, то на экране отобразятся данные, содержащиеся в выбранной таблице рис. 1.11, их можно просматривать и редактировать.

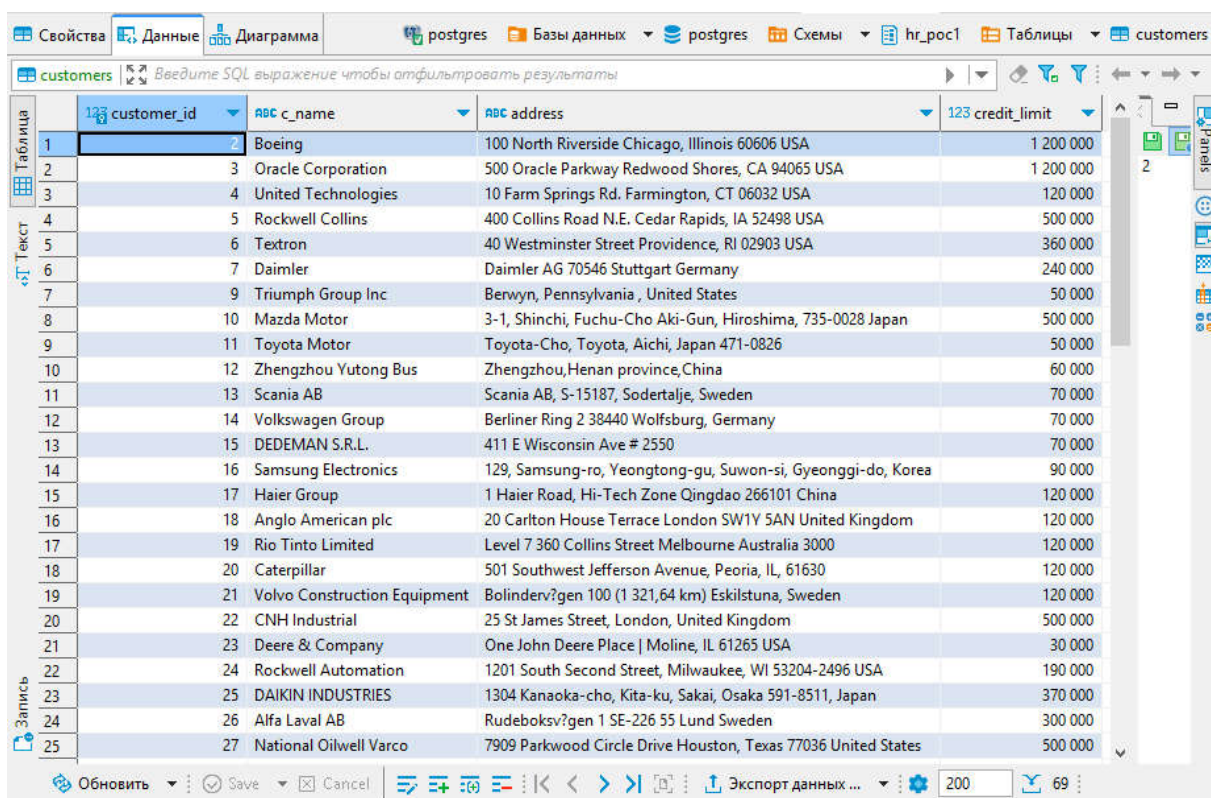


Рис. 1.11. Данные содержащиеся в таблице customers

## 1.5. Редактор SQL

Редактор SQL используется для создания, редактирования и выполнения запросов SQL и скриптов. Скрипт это последовательность запросов или команд языка программирования.

### 1.5.1. Окно редактора SQL

Для одного соединения можно открыть несколько окон редактора SQL. Открыть окно этого редактора можно различными способами. Мы будем использовать раскрывающийся список **Открыть скрипт SQL**, в котором следует выбрать команду **Новый редактор SQL** рис.1.12, или комбинацию клавиш **Ctrl + J**.

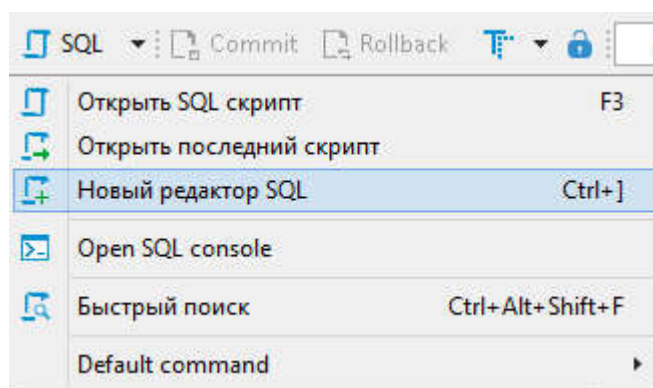


Рис.1.12. Выбор команды Новый редактор SQL

В результате на экране появится окно редактора SQL рис.1.13. Для того чтобы выполнить новый запрос нужно:

- выбрать соединение и схему,
- ввести текст запроса,
- нажать кнопку Выполнить SQL запрос.

В нижней части окна редактора появится панель результатов, которая будет содержать результаты выполнения запроса.

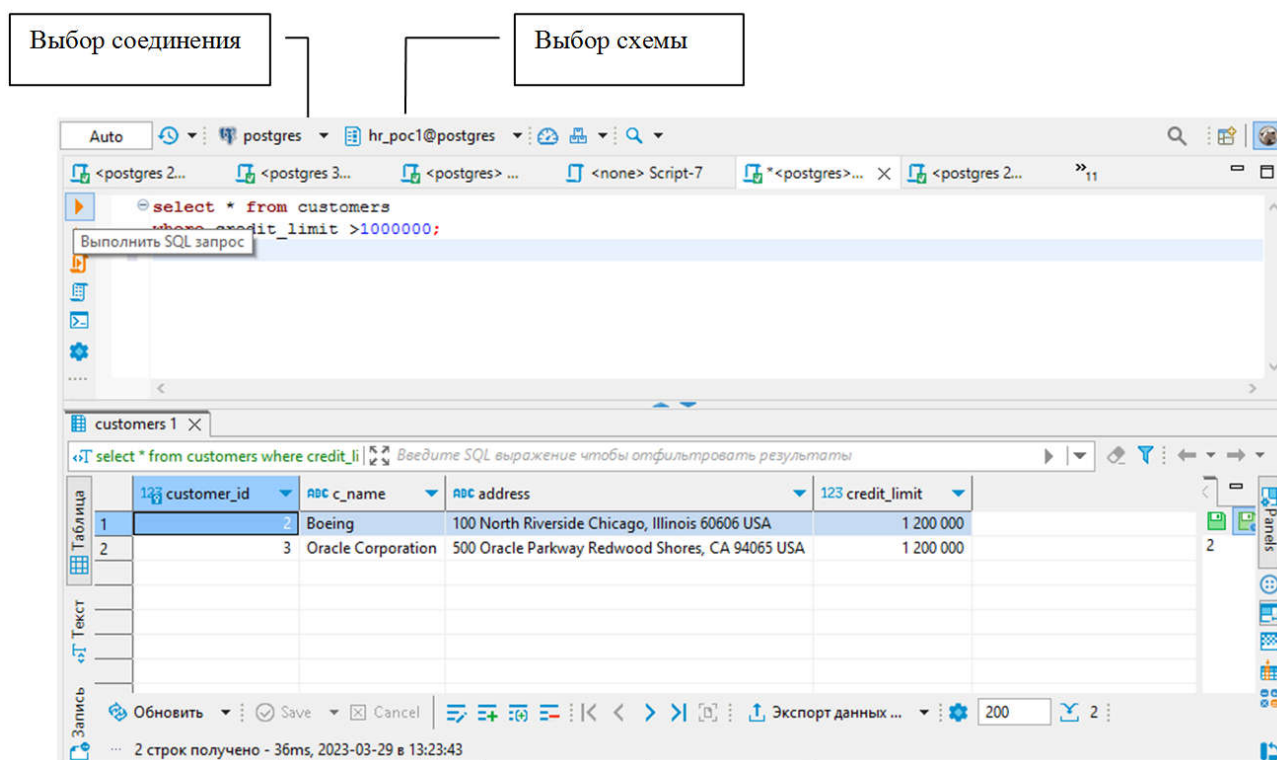
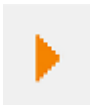






Рис.1.13. Окно редактора SQL

Редактор SQL имеет настраиваемую панель инструментов. В таблице 1.1. содержится описание основных кнопок этой панели.

Таблица 1.1. Описание кнопок панели инструментов редактора SQL.

Кнопка	Описание
	Выполнить инструкцию SQL ( Ctrl + Enter). Происходит выполнение запроса и полученные результаты отображаются на панели результатов или в окне SQL терминала
	Выполнить инструкцию SQL в новой вкладке (Ctrl +\). Эта кнопка аналогична предыдущей кнопки, но результаты запроса отображаются на новой вкладке панели результатов. Эту кнопку удобно использовать для сравнения результатов 2х запросов.
	Выполнить скрипт (Alt+X). Происходит выполнение скрипта, содержащегося в окне редактора SQL, и его результаты отображаются в окне вывода (консоли) и/или окне SQL терминала.

	Показать план выполнения запроса (Shift + Ctrl +E)
	SQL терминал. Результаты выполнения SQL запроса или скрипта будут выведены на специальной вкладке панели результатов, в текстовом формате..

### 1.5.2. SQL терминал

SQL Terminal – это вкладка результатов редактора SQL, где отображаются результаты выполненных запросов в текстовом формате. Это очень удобный способ вывода результатов. Для того чтобы открыть SQL Terminal, нужно нажать соответствующую кнопку на левой панели инструментов редактора SQL. На рис.1.14 показано окно редактора SQL, которое содержит запрос и результаты его выполнения.

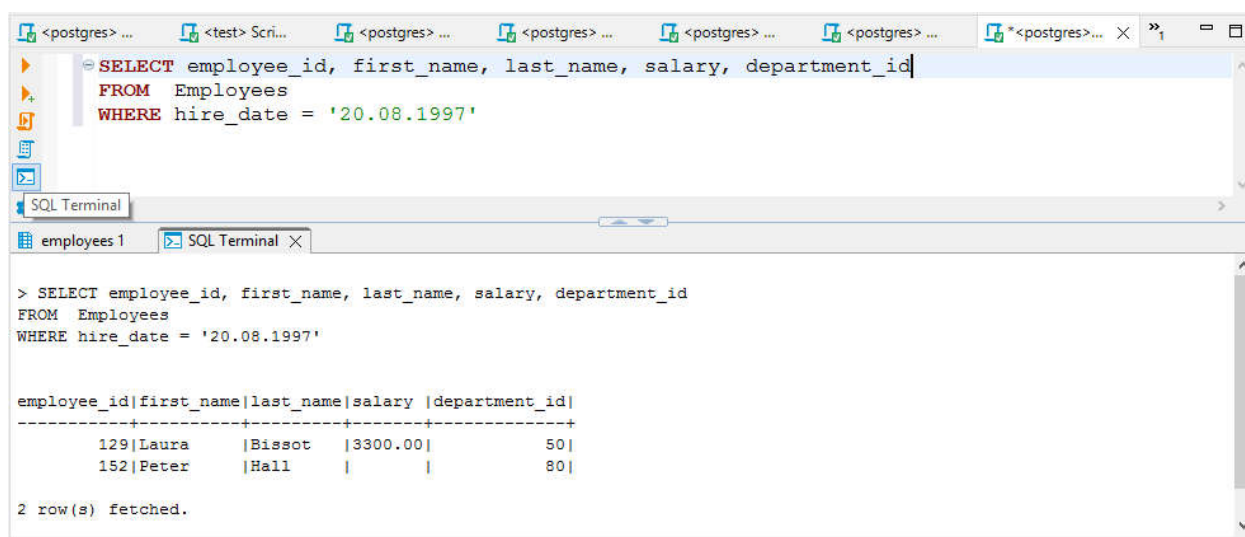


Рис.1.14. Отображение результатов на вкладке SQL Terminal

По умолчанию SQL Terminal не используется. Для того чтобы его включить и настроить нужно нажать кнопку **Настройки**, на панели инструментов редактора SQL, и в появившемся окне рис. 1.15 выбрать команду **Редакторы → SQL Terminal**.

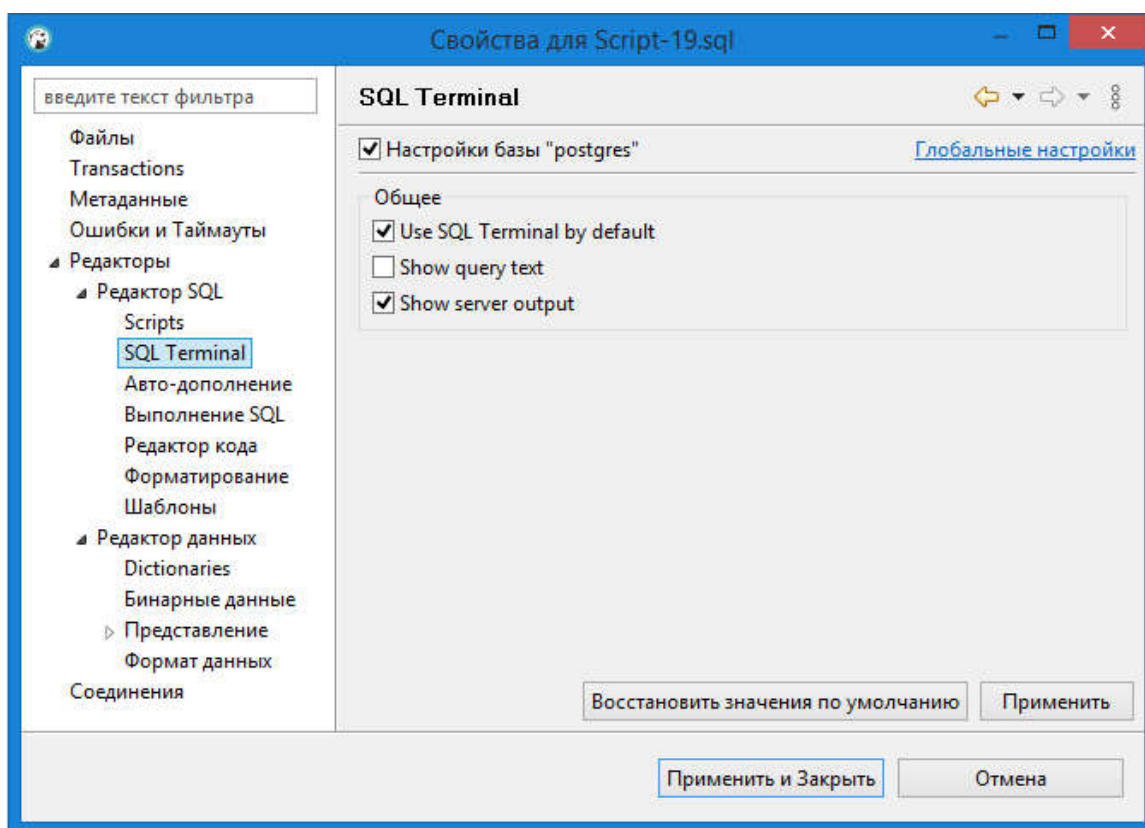


Рис. 1.15. Настройка SQL Terminal

### 1.5.3. Сохранение запросов

Можно сохранять запросы в текущем проекте или где-нибудь в файловой системе. Чтобы сохранить запрос в текущем проекте, нажмите **Ctrl+S** или выберите команду **Сохранить** в контекстном меню панели сценариев. Запрос, сохраненный таким образом, можно найти в папке **Scripts** проекта..

Чтобы сохранить скрипт в файловой системе, выберите команду **Файл -> Сохранить SQL-скрипт**, в контекстном меню рис.1.16, а затем выберите папку в файловой системе и введите имя файла рис.1.17.



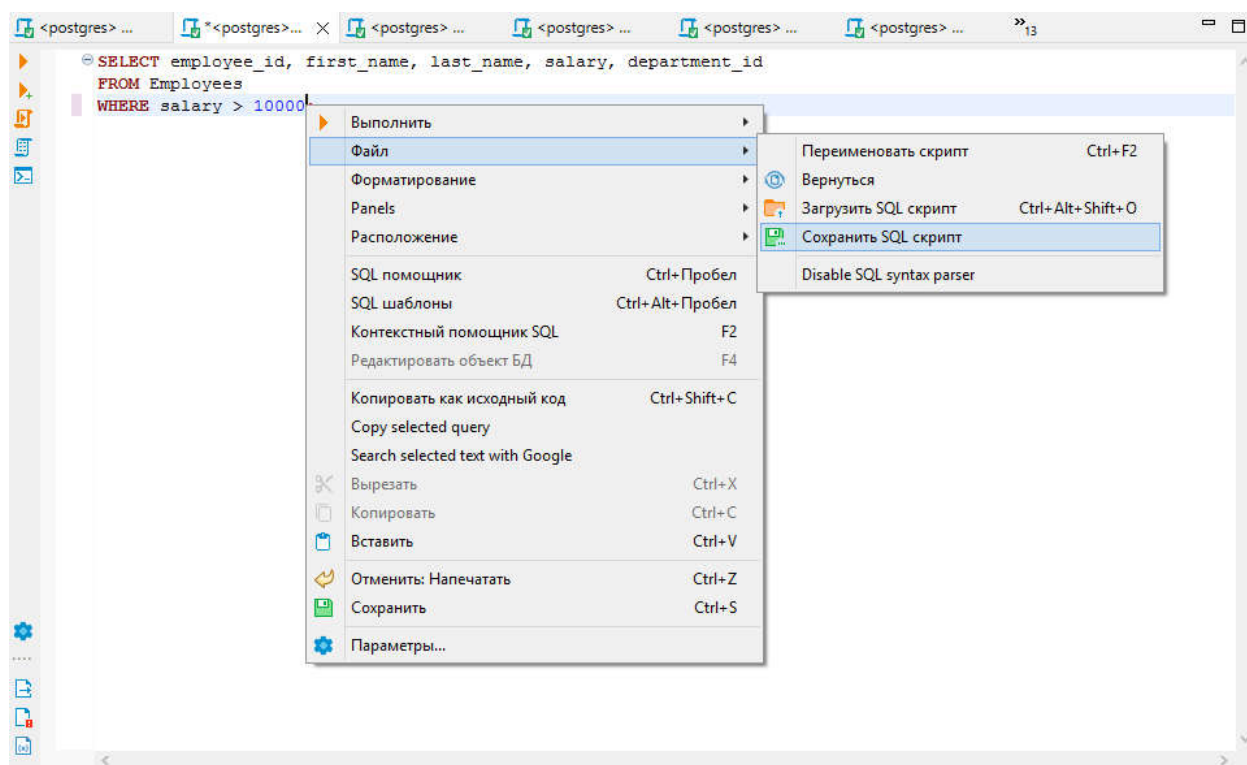


Рис. 1.16. Сохранение запроса в файловой системе

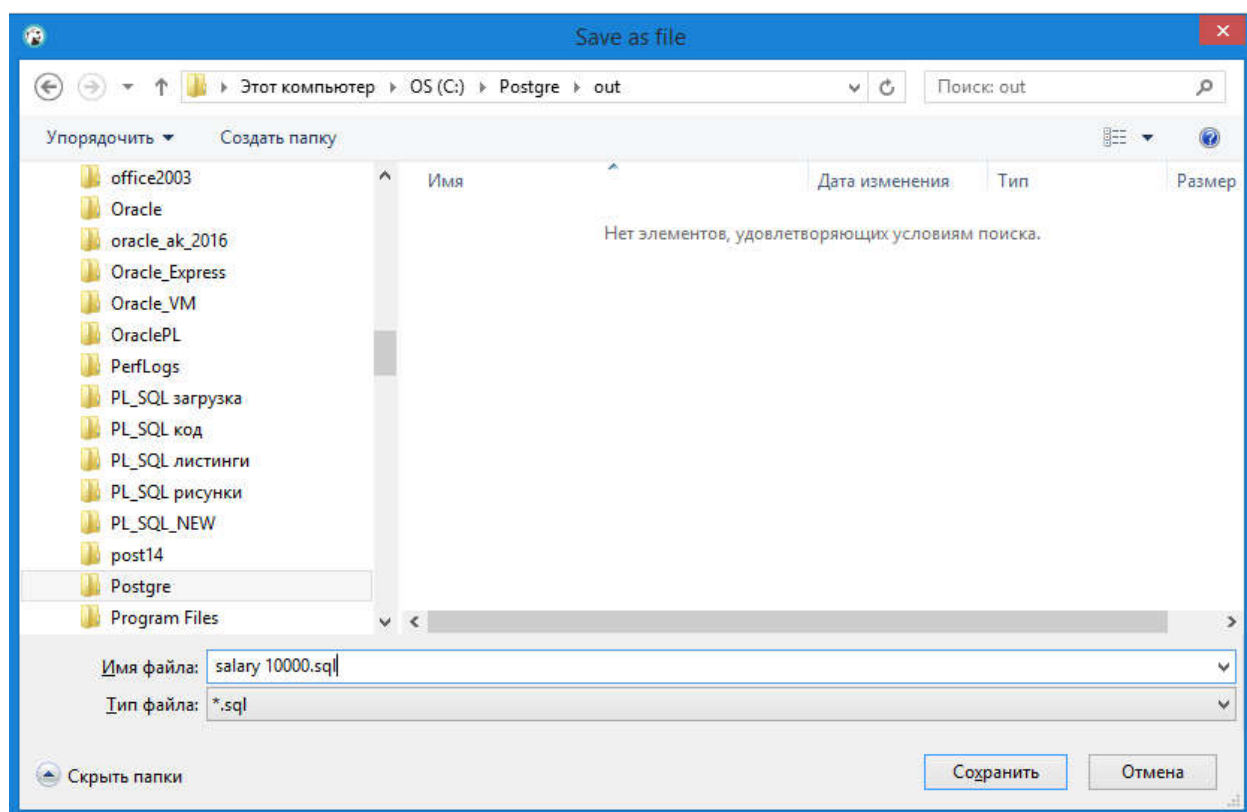


Рис. 1.17. Выбор папки и ввод имени файла

Чтобы загрузить скрипт, хранящийся в файловой системе, выберите команду **Файл -> Загрузить SQL-скрипт** в контекстном меню рис.1.16, а затем

## 1.6. Диаграммы сущность – связь

Диаграммы сущность – связь (ERD) это графические представления таблиц (сущностей) базы данных и связей между ними. DBeaver позволяет просматривать диаграммы существующих таблиц, схемы всей базы данных и создавать пользовательские диаграммы.

### 1.6.1. Диаграммы таблиц и схем

Если выбрать вкладку **Диаграмма**, в верхней части окна **Свойства таблицы** рис.1.10, то на экране будет отображено графическое представление таблицы и таблиц, с которыми эта таблица связана рис.1.18.

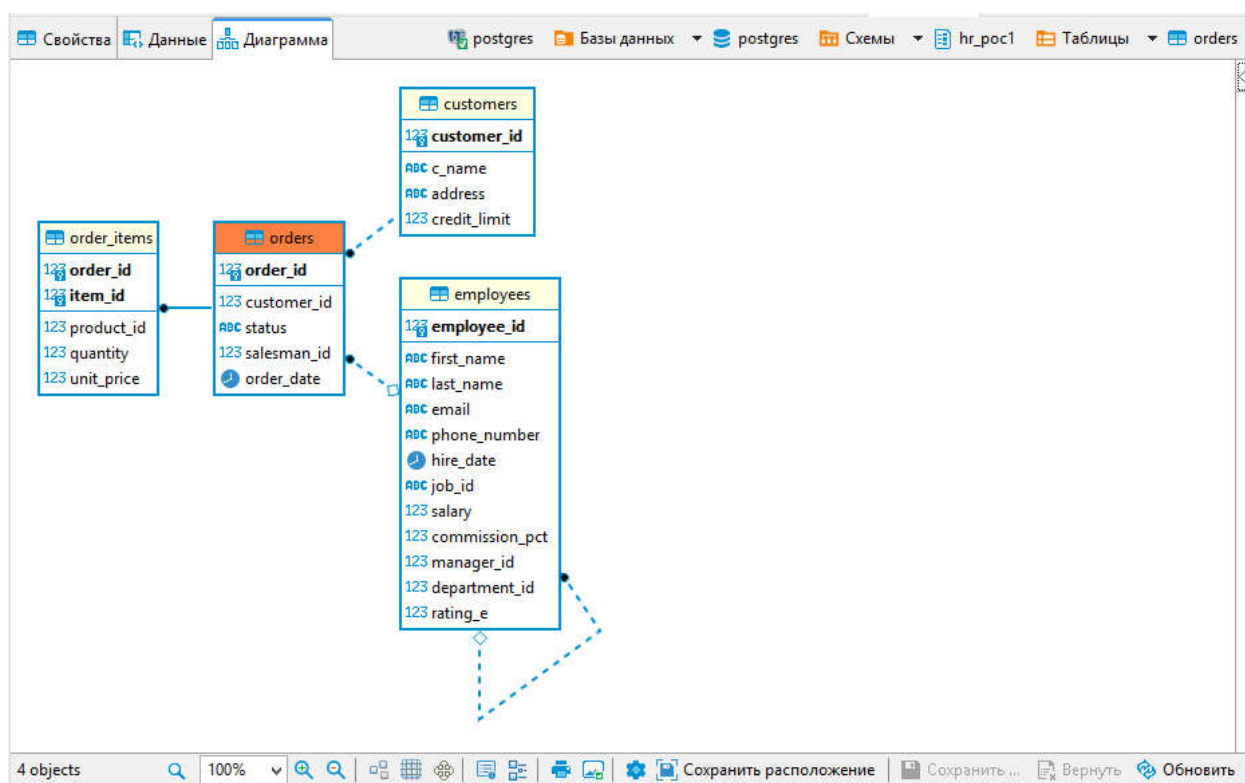


Рис.1.18. Диаграмма сущность – связь таблицы orders

Если выбрать вкладку **Диаграмма**, в верхней части окна **Свойства схемы**, то на экране будет отображена диаграмма рис.1.19, которая представляет собой



графическое представление всех таблиц (сущностей) этой схемы и связей между ними.

В нижней части окна на рис.1.19 имеется панель инструментов, кнопки которой позволяют изменить внешний вид диаграммы, распечатать ее или сохранить во внешнем файле.

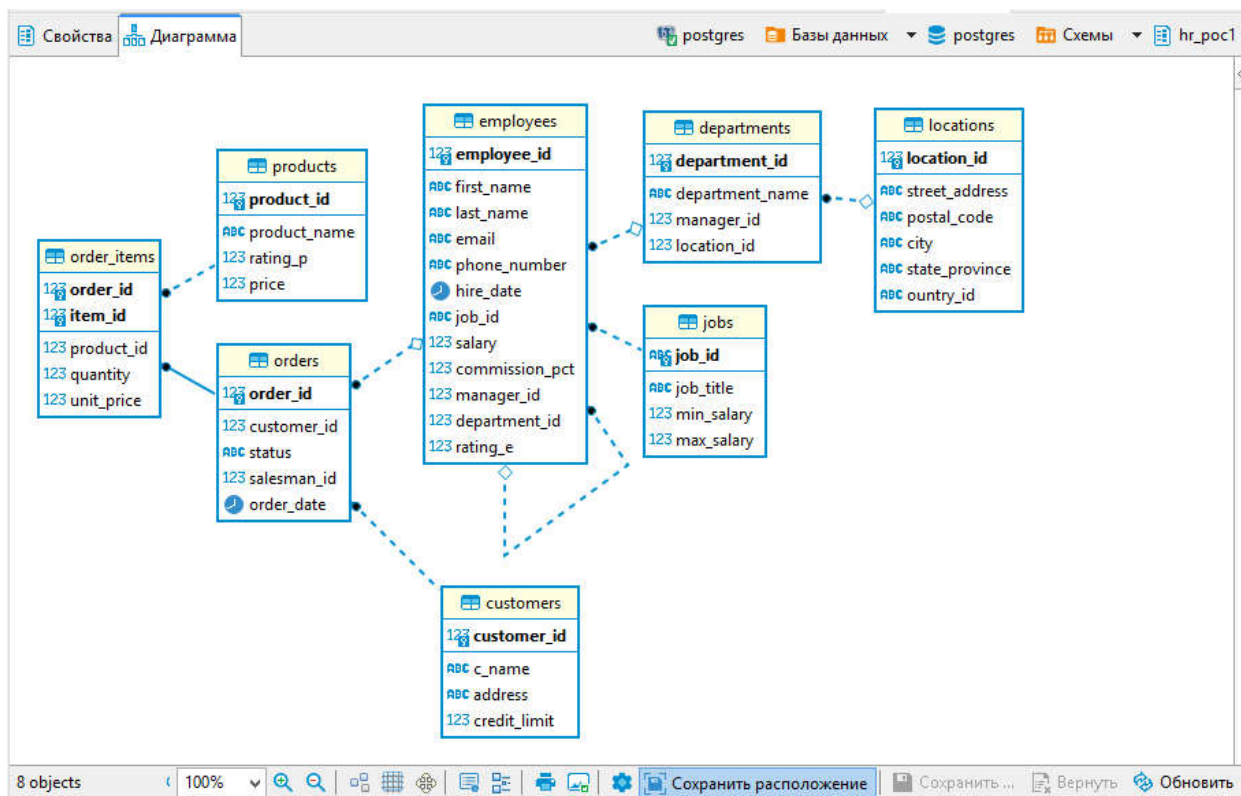






Рис.1.19. Диаграмма сущность – связь схемы hr\_рос

Линии, представляющие связи между таблицами, могут выглядеть по-разному, в зависимости от типа связи. Описание типов связей содержится в таблице 1.2.

Таблица 1.2. Типы связей

Обозначения	Описание
	Идентифицирующая связь, это означает, что столбец внешнего ключа одновременно является частью первичного ключа.
	Обычная связь

Обозначения	Описание
	Черная точка указывает таблицу, которая содержит внешний ключ
	Ромб используется в конце линии когда столбец, являющийся внешним ключом, может иметь значения NULL.

### 1.6.2. Описание используемой схемы базы данных

При создании SQL-запросов и программ PL/pgSQL нужно иметь четкое представление о схеме базы данных, с которой вы работаете, и знать бизнес-правила и ограничения, которые существуют в предметной области.

На рис. 1.19. представлена E-R диаграмма схемы **HR\_POC**, которую мы будем использовать. Рассмотрим назначение таблиц этой схемы и сформулируем бизнес-правила, которые необходимо соблюдать.

В таблице **Employees** содержатся данные о сотрудниках. Каждый сотрудник компании имеет уникальный идентификационный номер (**employee\_id**), номер должности (**job\_id**), ставку заработной платы (**salary**) и код менеджера (**manager\_id**). Некоторые сотрудники в дополнение к зарплате получают комиссионные (**commission\_pct**). Размер комиссионных определяется как часть от заработной платы. Столбец **job\_id** используется для установления связи с таблицей **Jobs**, и для него определено ограничение внешнего ключа. Следствием этого является то, что значение данного столбца должно совпадать с одним из значений столбца **job\_id** в таблице **Jobs** или иметь неопределенное значение **NULL**. Это ограничение обеспечивается средствами СУБД. Аналогичными свойствами обладает столбец **department\_id**, который используется для установления связи с таблицей **Departments**.

В таблице **Jobs** содержится информация обо всех возможных должностях организации. Каждая должность имеет уникальный идентификационный номер

(**job\_id**), наименование (**job\_title**), минимальную (**min\_salary**) и максимальную ставку заработной платы (**max\_salary**). При изменении заработной платы сотрудника, занимающего определенную должность, нужно следить, чтобы новая зарплата не выходила за заданные границы. Средствами СУБД это ограничение не обеспечивается.

Данные об отделах содержатся в таблице **Departments**. Каждый отдел имеет уникальный код (**department\_id**), код руководителя (**manager\_id**), наименование (**department\_name**), а также место расположения (**location\_id**).

Компания имеет распределенную структуру, поэтому в таблице **Locations** хранятся данные о местонахождении отделов, которые состоят из адреса (**street\_address**), почтового индекса (**postal\_code**), названия города (**city**), названия штата (**state\_province**) и кода страны (**country\_id**). В таблице **Locations** содержатся данные о населенных пунктах, в которых пока нет отделов.

В таблице **Customers** хранятся данные о клиентах. Столбец **customer\_id** содержит код клиента и является ключом этой таблицы. В этой таблице есть столбцы: **c\_name** – имя клиента, **address** – адрес клиента, **credit\_limit** – кредитный лимит. Используя столбец **credit\_limit** можно сформулировать следующее правило: запретить оформление заказа, если общая сумма заказов клиента, находящихся в состоянии ожидания, превышает его кредитный лимит.

Таблица **Order** содержит данные о заказах и имеет следующие столбцы: **customer\_id** – код клиента для которого оформлен заказ, **salesman\_id** – код сотрудника, который оформил заказ, **order\_date** – дата оформления заказа. Столбец **status**, таблицы **Order**, определяет состояние заказа и может принимать следующие значения: **Pending** – «в ожидании», **Shipped** – «отправлен», **Canceled** – «отменен». Используя этот столбец, сформулируем следующее бизнес-правило: можно изменить содержимое заказа, который находится в состоянии **Pending**, но нельзя изменить содержимое заказа, который находится в состоянии **Shipped**.

Заказ может содержать несколько товаров. Для хранения данных о товарах в каждом заказе служит таблица **Order\_Items**. Разберем назначение столбцов: **order\_id** – номер заказа; **item\_id** – строка (пункт) заказа; **product\_id** – код товара; **quantity** – количество товара в заказе; **unit\_price** – продажи.

В таблице **Products** содержатся следующие данные о товарах: **product\_id** – код товара; **product\_name** – наименование товара; **rating\_p** – рейтинг товара, **price** – цена товара.

Столбец **price** в таблице **Products** содержит текущую цену товара, а столбец **unit\_price** в таблице **Order\_Items** – цену, по которой он был продан. Разница между значениями этих столбцов может возникать из-за того, что клиенту была предоставлена скидка, также со временем значение **price** может измениться, а значение **unit\_price** – нет.

В таблице **Employees** имеется столбец **rating\_e**. Значение элементов этого столбца целочисленные и должны лежать в диапазоне от 1 до 5. Будем считать, что значение столбца **rating\_e** отражает квалификацию сотрудника.

В таблице **Products** содержится столбец **rating\_p**. Значения элементов этого столбца также должны лежать в диапазоне от 1 до 5 и отражают сложность товара.

Используя столбцы **rating\_e** и **rating\_p**, можно сформулировать следующее бизнес-правило: сотрудник имеет право продавать только те товары, рейтинг которых не превышает его рейтинга. Это бизнес-правило мы будем неоднократно использовать при решении задач.