

**Министерство науки и высшего образования Российской Федерации**  
федеральное государственное автономное образовательное учреждение высшего образования  
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе №3 «Создание таблиц базы данных PostgreSQL. Заполнение таблиц  
рабочими данными»

по дисциплине «Проектирование и реализация баз данных»

Автор: Федорин К.В.

Факультет: ИКТ

Группа: К3240

Преподаватель: Говорова М.М.



Санкт-Петербург 2023

## ЛАБОРАТОРНАЯ РАБОТА №3.2

### СОЗДАНИЕ ТАБЛИЦ БАЗЫ ДАННЫХ PostgreSQL. ЗАПОЛНЕНИЕ ТАБЛИЦ РАБОЧИМИ ДАННЫМИ

**Цель работы:** овладеть практическими навыками создания таблиц базы данных PostgreSQL 1X, заполнения их рабочими данными, резервного копирования и восстановления БД.

**Оборудование:** компьютерный класс.

**Программное обеспечение:** СУБД PostgreSQL 1X, pgAdmin 4.

#### Практическое задание:

1. Создать базу данных с использованием pgAdmin 4 (согласно индивидуальному заданию).
2. Создать схему в составе базы данных.
3. Создать таблицы базы данных.
4. Установить ограничения на данные: *Primary Key, Unique, Check, Foreign Key*.
5. Заполнить таблицы БД рабочими данными.
6. Создать резервную копию БД.

Указание:

Создать две резервные копии:

- с расширением *CUSTOM* для восстановления БД;
  - с расширением *PLAIN* для листинга (в отчете);
  - при создании резервных копий БД настроить параметры *Dump options* для *Type of objects* и *Queries*.
7. Восстановить БД.

#### Технология выполнения работы:

##### Создание базы данных

Технология создания базы данных с использованием pgadmin 4 рассмотрена в лабораторной работе 3.1.

##### Создание схемы

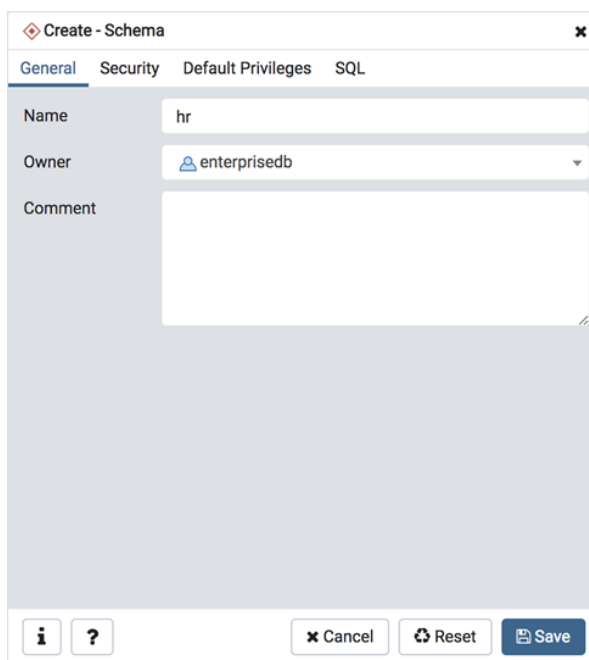
Для определения схемы базы данных используется диалоговое окно *Schema* (Схема). Схема – это организационная рабочая область базы данных, похожая на каталоги или пространства имен. Чтобы создать схему, вы должны быть суперпользователем базы данных или иметь привилегию CREATE.

*Schema*-диалог организует разработку схемы с помощью следующих диалоговых вкладок: *General* и *Security*. На вкладке *SQL* отображается код SQL, созданный в диалоговых окнах (рис. 1).

Для идентификации схемы используются поля на вкладке *Общие*:

- поле «Имя» используется, чтобы добавить описательное имя для схемы. Имя будет отображаться в древовидном элементе управления *pgAdmin*;
- выбрать владельца схемы из раскрывающегося списка в поле «Владелец»;
- хранить заметки о схеме в поле *Комментарий*.

Полное описание диалога схемы:  
[https://www.pgadmin.org/docs/pgadmin4/4.30/schema\\_dialog.html](https://www.pgadmin.org/docs/pgadmin4/4.30/schema_dialog.html)



*Рисунок 1 – Диалог создания схемы*

## Создание или изменение таблицы

pgAdmin 4 предоставляет диалоги, которые позволяют изменять все свойства и атрибуты таблицы.

Используйте диалог *Таблица* для создания или изменения таблицы.

Диалоговое окно «Таблица» организует разработку таблицы с помощью следующих диалоговых вкладок: «Общие», «Столбцы», «Ограничения», «Дополнительно», «Параметр» и «Безопасность». На вкладке *SQL* отображается код SQL, созданный в диалоговых окнах (рис. 2).

The image shows a 'Create - Table' dialog box with the following fields and options:

- Name:** pem.test
- Owner:** enterprisedb
- Schema:** public
- Tablespace:** Select from the list
- Partitioned Table?:** No
- Comment:** (empty text area)

Buttons at the bottom: Cancel, Reset, Save.

Рисунок 2 – Диалоговое окно создания таблицы

Чтобы определить таблицу, используются поля на вкладке *Общие*:

- поле *Имя* используется, чтобы добавить описательное имя для таблицы. Таблица не может иметь того же имени, что и любая существующая таблица, последовательность, индекс, представление, сторонняя таблица или тип данных в той же схеме. Указанное имя будет отображаться в древовидном элементе управления *pgAdmin*. Это поле обязательно к заполнению.
- Выбирать владельца таблицы в раскрывающемся списке в поле «*Владелец*». По умолчанию владельцем таблицы является роль, которая создает таблицу.
- Выбирать имя схемы, в которой будет находиться таблица, из раскрывающегося списка в поле *Схема*.
- Использовать раскрывающийся список в поле *Табличное пространство*, чтобы указать табличное пространство, в котором будет храниться таблица.
- Хранить заметки о таблице в поле «*Комментарий*».

Для определения столбцов таблицы используется вкладка *Столбцы* (рис. 3).

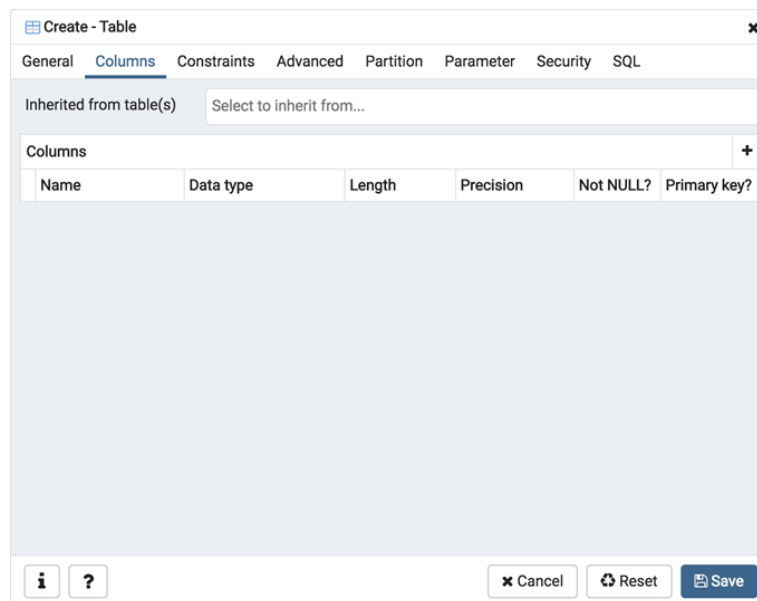


Рисунок 3 – Определение столбцов таблицы

Чтобы указать любые родительские таблицы, используется раскрывающийся список рядом с *Наследуется от таблицы (таблиц)*; таблица будет наследовать столбцы от выбранных родительских таблиц. Щелкнуть внутри поля *Inherited from table (s)*, чтобы выбрать имя таблицы в раскрывающемся списке. Повторитт, чтобы добавить любые другие родительские таблицы. Удалить выбранную таблицу, нажав на  $\times$  слева от имени родителя. Унаследованные имена столбцов и типы данных не могут редактироваться в текущем диалоге; они должны быть изменены на родительском уровне.

Чтобы указать имена столбцов и их типы данных в таблице «*Столбцы*», щелкнуть значок «*Добавить*» (+):

- использовать поле *Имя*, чтобы добавить описательное имя для столбца;
- Использовать раскрывающийся список в поле *Тип данных*, чтобы выбрать тип данных для столбца. Это может включать в себя спецификаторы массива. Для получения дополнительной информации о типах данных, поддерживаемых PostgreSQL, обратиться к основной документации (URL: <https://www.postgresql.org/docs/10/datatype.html>).
- если включено, использовать поля *Длина* и *Точность*, чтобы указать максимальное количество значащих цифр в числовом значении или максимальное количество символов в текстовом значении.
- для *NOT NULL?* Переключить No в положение Yes, чтобы запросить значение в поле столбца;
- для задания первичного ключа *Primary key?* переключить в положение «*Yes*», чтобы указать столбец с ограничением первичного ключа.

Чтобы добавить дополнительные столбцы, нажмите значок «*Добавить*» (+). Чтобы удалить столбец, щелкнуть значок корзины слева от строки и подтвердите удаление во всплывающем окне «*Удалить строку*».

Полное описание диалога создания таблицы:  
[https://www.pgadmin.org/docs/pgadmin4/4.30/table\\_dialog.html](https://www.pgadmin.org/docs/pgadmin4/4.30/table_dialog.html)

## Задание ограничений для таблицы

Для создания ограничений таблицы можно использовать диалог *Constraint* таблицы. Для создания ограничения необходимо в контекстном меню диалога *Constraint* выбрать *тип ограничения*.

### Ограничения Check

Используйте диалоговое окно «*Check*» для определения или изменения проверочного ограничения. Ограничение проверки определяет выражение, которое выдает логический результат, которому должны удовлетворять новые или обновленные строки, чтобы операция вставки или обновления выполнялась успешно.

Диалоговое окно «*Check*» организует разработку проверочного ограничения с помощью вкладок «*General*» и «*Definition tabs*». На вкладке *SQL* отображается код SQL, созданный в диалоговых окнах (рис. 4).

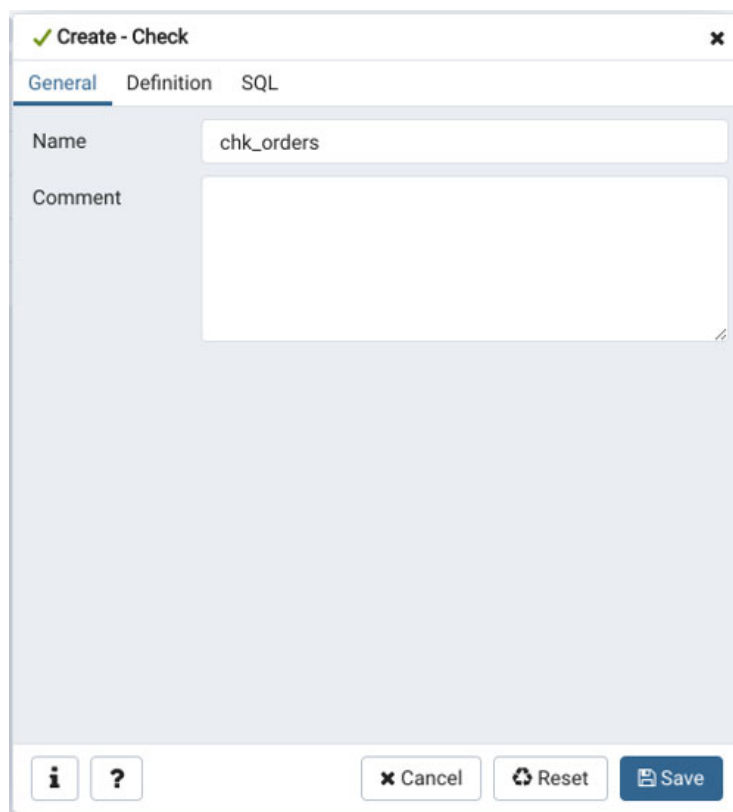


Рисунок 4 – Диалог создания *Check*-ограничения

Использовать поля на вкладке *Общие*, чтобы определить проверочное ограничение:

- использовать поле *Name*, чтобы предоставить описательное имя для проверочного ограничения, которое будет отображаться в элементе управления дерева *pgAdmin*. В PostgreSQL 9.5 и далее, когда таблица имеет несколько проверочных ограничений, они будут проверяться для каждой строки в алфавитном порядке по имени и после ограничений NOT NULL.

- хранить заметки о проверочном ограничении в поле *Comment*.

Нажать вкладку «*Definition*», чтобы продолжить.

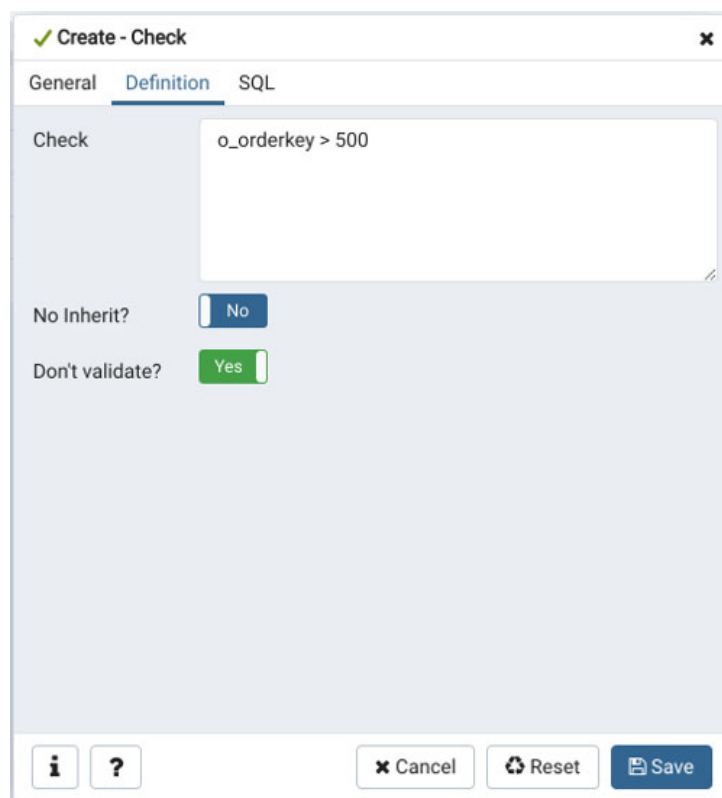


Рисунок 5 – Определение ограничения

Полное описание диалога для определения *Check-ограничения*: [https://www.pgadmin.org/docs/pgadmin4/latest/check\\_dialog.html](https://www.pgadmin.org/docs/pgadmin4/latest/check_dialog.html).

Чтобы просмотреть SQL-скрипт добавления ограничения, нужно выбрать в контекстном меню ограничения *Script Create*.

### Ограничения Foreign key

Используйте диалоговое окно «*Foreign key*», чтобы указать поведение ограничения внешнего ключа. Ограничение внешнего ключа поддерживает ссылочную целостность между двумя таблицами. Ограничение внешнего ключа не может быть определено между временной таблицей и постоянной таблицей.

Диалоговое окно «*Foreign key*» организует разработку ограничения внешнего ключа с помощью следующих вкладок: *General*, *Definition*, *Columns* и *Action*. На вкладке *SQL* отображается код SQL, созданный в диалоговых окнах (рис. 6).

The screenshot shows the 'Create - Foreign key' dialog box with the 'General' tab selected. The 'Name' field contains 'fk\_orders'. The 'Comment' field is empty. At the bottom, there are buttons for 'Cancel', 'Reset', and 'Save', along with information and help icons.

Tab	Name	Comment
General	fk_orders	

Рисунок 6 – Определение ограничения Foreign key

Нажать вкладку «*Definition*», чтобы продолжить (рис. 7).

The screenshot shows the 'Create - Foreign key' dialog box with the 'Definition' tab selected. The 'Deferrable?' field is set to 'No'. The 'Deferred?' field is set to 'No'. The 'Match type' field is set to 'SIMPLE'. The 'Validated?' field is set to 'No'. The 'Auto FK index?' field is set to 'No'. The 'Covering index' field contains 'orders\_pkey'. At the bottom, there are buttons for 'Cancel', 'Reset', and 'Save', along with information and help icons.

Field	Value
Deferrable?	No
Deferred?	No
Match type	SIMPLE
Validated?	No
Auto FK index?	No
Covering index	orders_pkey

Рисунок 7 – Диалог определения внешнего ключа



Использовать поля на вкладке «*Definition*» для определения ограничения внешнего ключа:

- переместить *Deferrable?* в положение «*Да*», чтобы указать, что время ограничения является отложенным и может быть отложено до конца оператора. По умолчанию это *Нет*.
  - переместить *Deferred?* в положение «*Да*», чтобы указать время, в течение которого ограничение будет отложено до конца оператора. По умолчанию это *Нет*.
  - переключатель *match* указывает тип соответствия, который подкреплено ограничением:
    - выбрать *Full*, чтобы указать, что все столбцы многоколоночного внешнего ключа должны быть нулевыми, если любой столбец пуст; если все столбцы являются нулевыми, строка не обязательно должна иметь совпадение в ссылочной таблице.
    - выбрать «*Simple*», чтобы указать, что один столбец внешнего ключа может быть пустым; если какой-либо столбец имеет значение null, строка не обязательно должна иметь совпадение в ссылочной таблице.
  - переместить переключатель *Validated* в положение *Yes*, чтобы указать серверу проверять существующее содержимое таблицы (по внешнему ключу или проверять ограничение) при сохранении изменений в этом диалоговом окне.
  - перевести переключатель *Auto FK Index* в положение *No*, чтобы отключить функцию автоматического индексирования.
  - поле рядом с *Covering Index* генерирует имя индекса, если переключатель *Auto FK Index* находится в положении *Yes* ; или это поле отключено.
- Нажать вкладку *Столбцы*, чтобы продолжить (рис. 8).

Local	Referenced
o_orderkey	n_regionkey

Рисунок 8 – Определение внешнего ключа

Использовать поля на вкладке «*Columns*», чтобы указать один или несколько столбцов. Ограничение внешнего ключа требует, чтобы один или несколько столбцов таблицы содержали только те значения, которые соответствуют значениям в столбце (столбцах), на который имеется ссылка строки таблицы, на которую имеется ссылка:

- использовать раскрывающийся список рядом с *Local column*, чтобы указать столбец в текущей таблице, который будет сравниваться с внешней таблицей.
- использовать раскрывающийся список рядом со списком «*References*», чтобы указать имя таблицы, в которой находятся столбцы сравнения.
- использовать раскрывающийся список рядом с *Referencing*, чтобы указать столбец во внешней таблице.

Нажать значок «*Add*» (+), чтобы добавить столбец в список; повторить описанные выше шаги и щелкнуть значок «*Add*» (+), чтобы добавить дополнительные столбцы. Чтобы отменить запись, щелкнуть значок корзины слева от записи и подтвердить удаление во всплывающем окне «*Delete Row*» .

Нажать вкладку *Action*, чтобы продолжить.

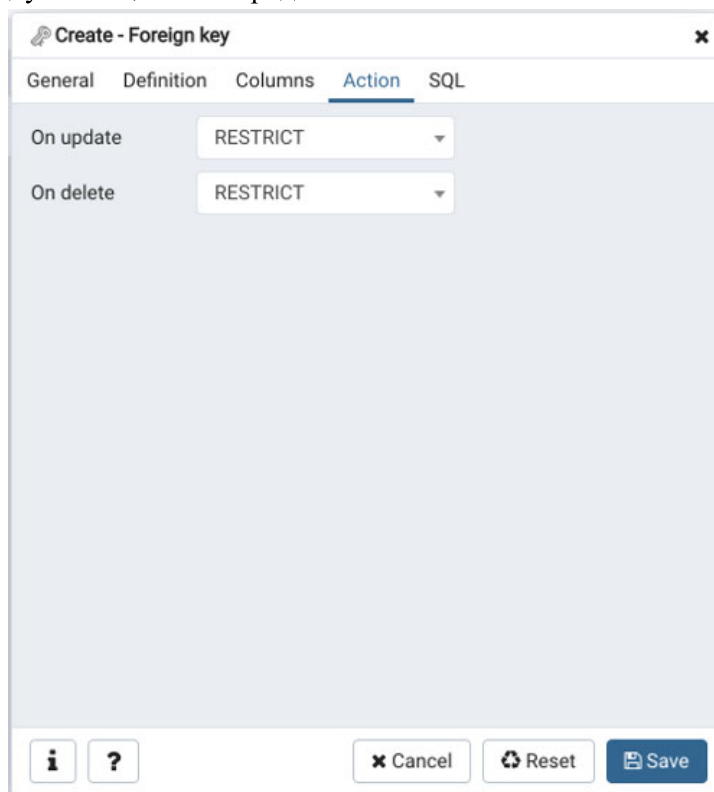


Рисунок 9 – Задание параметров ограничения

Использовать раскрывающиеся списки на вкладке *Action*, чтобы указать поведение, связанное с ограничением внешнего ключа, которое будет выполняться при обновлении или удалении данных в таблице:

- используйте раскрывающийся список рядом с *On update*, чтобы выбрать действие, которое будет выполняться при обновлении данных в таблице.
- используйте раскрывающийся список рядом с *On delete*, чтобы выбрать действие, которое будет выполняться при удалении данных в таблице.

Поддерживаемые действия: см. презентацию Введение в SQL.

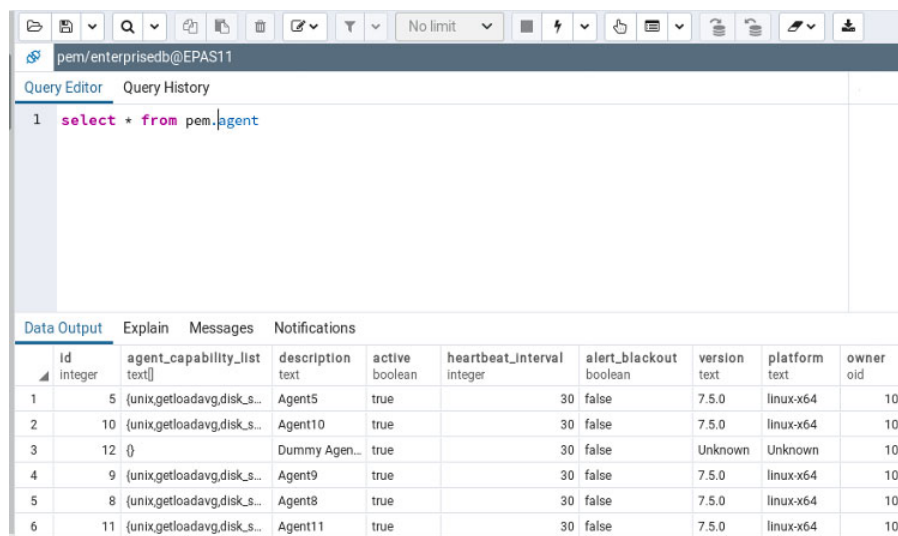
В диалоге *Constraint* можно также создать ограничения первичного ключа и уникальности значения столбца.

Полное описание диалога для определения внешнего ключа:  
[https://www.pgadmin.org/docs/pgadmin4/6.20/foreign\\_key\\_dialog.html](https://www.pgadmin.org/docs/pgadmin4/6.20/foreign_key_dialog.html)

## Заполнение таблиц рабочими данными

Инструмент запросов (рис. 10) – это мощная, многофункциональная среда, которая позволяет выполнять произвольные команды SQL и просматривать набор результатов. Вы можете получить доступ к Query Tool через пункт меню *Query Tool* в меню *Tools* или через контекстное меню отдельных узлов элемента управления дерева Browser. Инструмент запросов позволяет вам:

- выпуск специальных SQL-запросов.
- выполнение произвольные команды SQL.
- отображение текущее состояние соединения и транзакции в соответствии с настройками пользователя.
- сохранение данные, отображаемых на панели вывода, в файл CSV.
- просмотр плана выполнения оператора SQL в текстовом или графическом формате.
- просмотр аналитической информации об операторе SQL.



The screenshot shows the Query Tool interface with a SQL query editor and a results table. The query is `select * from pem.agent`. The results table has 10 rows and 10 columns.

	Id integer	agent_capability_list text[]	description text	active boolean	heartbeat_interval integer	alert_blackout boolean	version text	platform text	owner oid
1	5	{unix.getloadavg,disk_s...	Agent5	true	30	false	7.5.0	linux-x64	10
2	10	{unix.getloadavg,disk_s...	Agent10	true	30	false	7.5.0	linux-x64	10
3	12	{}	Dummy Agen...	true	30	false	Unknown	Unknown	10
4	9	{unix.getloadavg,disk_s...	Agent9	true	30	false	7.5.0	linux-x64	10
5	8	{unix.getloadavg,disk_s...	Agent8	true	30	false	7.5.0	linux-x64	10
6	11	{unix.getloadavg,disk_s...	Agent11	true	30	false	7.5.0	linux-x64	10

Рисунок 10 – Инструмент запросов Query Tool

Можно одновременно открыть несколько копий инструмента Query на отдельных вкладках. Чтобы закрыть копию инструмента .

«Запрос», щелкнуть X в верхнем правом углу панели вкладок.

Query Tool имеет две панели:

- на верхней панели отображается *SQL Editor*. Вы можете использовать панель для ввода, редактирования или выполнения запроса. Он также показывает *историю* вкладку, которую можно использовать для просмотра запросов, которые были выполнены в сессии, и *блокнот*, который может быть использован для хранения фрагментов текста во время редактирования. Если Scratch Pad закрыт, его можно повторно открыть (или открыть дополнительные), щелкнув правой кнопкой мыши в редакторе SQL и других панелях и добавив новую панель.

- на нижней панели отображается панель «*Data Output*». Панель с вкладками отображает набор результатов, возвращаемый запросом, информацию о плане выполнения запроса, серверные сообщения, связанные с выполнением запроса, и любые асинхронные уведомления, полученные от сервера.

Полное описание *Query Tool*:

[https://www.pgadmin.org/docs/pgadmin4/latest/query\\_tool.html](https://www.pgadmin.org/docs/pgadmin4/latest/query_tool.html)

Используя команду *insert* помощью *Query Tool*, необходимо заполнить таблицы БД рабочими данными и проверить их содержимое, используя команду *select \* from <имя-таблицы>*.

## Создание резервной копии базы данных

Утилита *pg\_dump pgAdmin* предоставляет простой способ создания резервной копии в текстовом или архивном формате. Затем можно использовать клиентское приложение (например, *psql* или *Query Tool*), чтобы восстановить текстовый файл резервной копии, или использовать утилиту Postgres *pg\_restore*, чтобы восстановить архивную резервную копию. *Pg\_dump* утилита должна иметь доступ на чтение всех объектов базы данных, для которых нужно резервную копию.

Можно сделать резервную копию одной таблицы, схемы или полной базы данных. Выбрать имя источника резервной копии в древовидном элементе управления *pgAdmin*, щелкните правой кнопкой мыши, чтобы открыть контекстное меню, и выберите «*Backup ...*», чтобы открыть диалоговое окно *Backup*. Имя выбранного объекта появится в строке заголовка диалога.

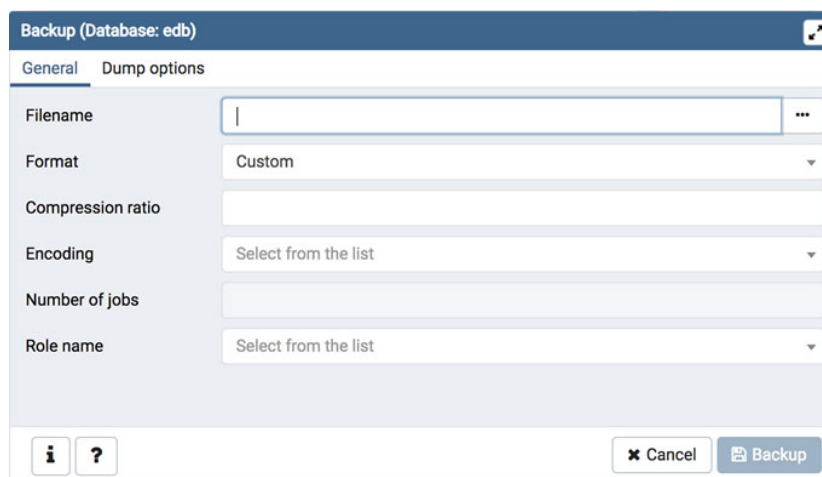


Рисунок 11 – Создание резервной копии БД

Использовать поля на вкладке *Общие*, чтобы указать параметры для резервной копии:

- ввести имя файла резервной копии в поле *Filename*. При желании выбрать значок *Browser* (...) справа, чтобы перейти в каталог и выбрать файл, который будет содержать архив.

- использовать раскрывающийся список в поле «*Format*», чтобы выбрать формат, который лучше всего подходит для приложения. Каждый формат имеет свои преимущества и недостатки:

- выбрать *Custom*, чтобы создать специальный файл архива, который можно использовать с *pg\_restore* для создания копии базы данных. Пользовательские форматы архивных файлов должны быть восстановлены с

помощью *pg\_restore*. Этот формат дает возможность выбрать, какие объекты базы данных восстановить из файла резервной копии. *Custom* формат архива рекомендуется для средних и больших баз данных, так как он сжат по умолчанию.

- выбрать *Tar*, чтобы создать файл архива tar, который можно восстановить с помощью *pg\_restore*. Формат tar не поддерживает сжатие.
- выбрать «*Plain*», чтобы создать текстовый файл сценария. Файл текстового сценария содержит операторы SQL и команды, которые можно выполнить в командной строке *psql*, чтобы воссоздать объекты базы данных и загрузить данные таблицы. Файл текстовой резервной копии может быть отредактирован в текстовом редакторе, если необходимо, перед использованием программы *psql* для восстановления объектов базы данных. *Plain*-формат обычно рекомендуется для небольших баз данных; дампы сценариев не рекомендуются для больших двоичных объектов. Команды SQL в скрипте восстанавливают базу данных до последнего сохраненного состояния базы данных. Простой текстовый сценарий может использоваться для восстановления базы данных на другом компьютере или (с изменениями) на других архитектурах.
- выбрать «*Каталог*», чтобы создать архив в формате каталога, подходящий для использования с *pg\_restore*. Этот формат файла создает каталог с одним файлом для каждой таблицы и *отбрасываемого большого двоичного* объекта, а также файл *оглавления*, описывающий выгруженные объекты в машиночитаемом формате, который может прочитать *pg\_restore*. Этот формат сжат по умолчанию.
  - использовать поле «*Compression Ratio*», чтобы выбрать уровень сжатия для резервной копии. Указать значение от нуля, чтобы обозначить использование без сжатия; указать максимальное значение сжатия, равное 9. Обратите внимание, что архивы tar не поддерживают сжатие.
  - использовать раскрывающийся список *Encoding*, чтобы выбрать метод кодировки символов, который следует использовать для архива.
  - использовать поле *Number of Jobs* (если применимо), чтобы указать количество таблиц, которые будут одновременно выгружаться в параллельном резервном копировании.
  - использовать раскрывающийся список рядом с *Rolename*, чтобы указать роль, которой принадлежит резервная копия.

Полное описание вкладки *Dump options*:

[https://www.pgadmin.org/docs/pgadmin4/latest/backup\\_dialog.html](https://www.pgadmin.org/docs/pgadmin4/latest/backup_dialog.html)

## Восстановление базы данных

Диалог *Restore* обеспечивает простой способ восстановления или обновления базы данных. Диалог *Backup* вызывает опцию утилиты *pg\_dump* клиента; диалог *Restore* вызывает опцию утилиты *pg\_restore* клиента.

Модно использовать *Query Tool* для воспроизведения скрипта, созданного во время простого текстового резервного копирования, сделанного с помощью диалога *Backup*. Для получения дополнительной информации о резервном копировании или восстановлении, пожалуйста, нужно обратиться к документации по [pg\\_dump](#) или [pg\\_restore](#).

Для восстановления БД необходимо создать БД с тем же именем на ПК.

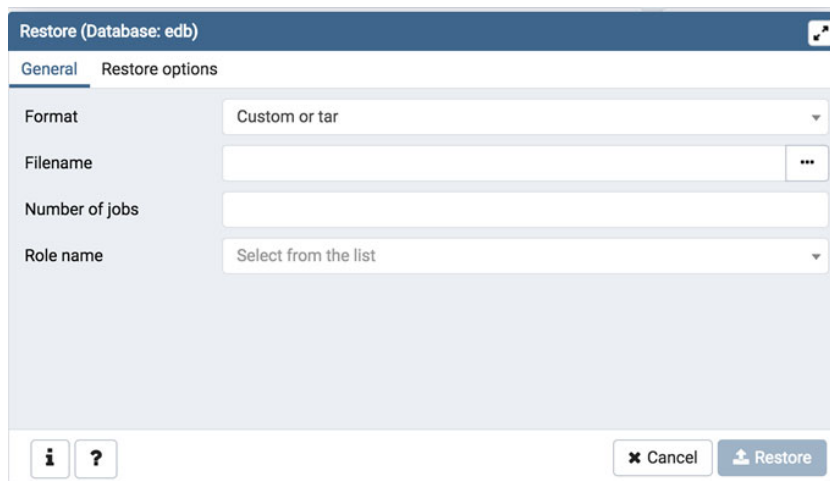


Рисунок 12 – Восстановление БД

Использовать поля на вкладке *Общие*, чтобы указать общую информацию о процессе восстановления:

- использовать раскрывающийся список в поле «Формат», чтобы выбрать формат файла резервной копии.
  - о выбрать Custom или tar для восстановления из пользовательского файла архива, чтобы создать копию объекта резервной копии.
  - о выбрать Каталог для восстановления из сжатого архива в формате каталога.
- ввести полный путь к файлу резервной копии в поле Filename . При желании выбрать значок «Обозреватель» (многоточие) справа, чтобы перейти в каталог и выбрать файл, содержащий архив.
- использовать поле Number of Jobs, чтобы указать, должно ли pg\_restore использовать несколько (одновременных) заданий для обработки восстановления. Каждое задание использует отдельное соединение с сервером.
- использовать раскрывающийся список рядом с Rolename, чтобы указать роль, которая будет использоваться для аутентификации на сервере во время процесса восстановления.

Полное описание параметров восстановления:  
[https://www.pgadmin.org/docs/pgadmin4/latest/restore\\_dialog.html](https://www.pgadmin.org/docs/pgadmin4/latest/restore_dialog.html)

### Вывод:

Реализовал свою базу данных с помощью pgAdmin4 по наработкам из лабораторной работы №2. Всё с учётом заданных ограничений на данные и их тип. Полученная БД была сохранена как LR\_3.sql, ERD к ней - LR\_3.pgerd