

Практическая работа № 1.

Резервное копирование и восстановление

Цель работы: научиться пользоваться утилитой pgAdmin для выполнения резервного копирования и восстановления базы данных.

Теоретический материал

Для предотвращения потери данных при сбоях периодически необходимо делать резервные копии рабочей базы данных. Если что-то пошло не так, то базу данных можно восстановить из резервной копии. Но в этой ситуации администратор сталкивается с дилеммой. С одной стороны, как правило, объём рабочих баз данных, используемых в современных организациях, составляет десятки гигабайт и более. Если делать резервные копии слишком часто, они займут слишком много места. С другой стороны, если делать их слишком редко, то при серьёзном сбое в работе сервера организация рискует потерять большой массив данных, который был внесён в базу данных с момента последнего резервного копирования. В связи с этим современные СУБД предоставляют инструментарий, который призван решить эту проблему.

Резервные копии подразделяются на полные и инкрементальные. При полном резервном копировании вся информация из базы данных сохраняется в резервную копию. При инкрементальном резервном копировании сохраняется только та часть данных, которая изменилась с момента последнего резервного копирования. При этом возможно два варианта: дифференциальное и кумулятивное резервное копирование. При дифференциальном резервном копировании происходит сравнение актуальной базы данных с последней резервной копией, и в новую резервную копию помещается только то, что было изменено. При кумулятивном резервном копировании в резервную копию помещаются страницы, которые были помечены как изменённые с момента последней резервной копии.

В соответствии с этим, для полного резервного копирования необходима только рабочая база данных, содержимое которой просто копируется в определённую папку. Для дифференциального резервного копирования необходима рабочая база данных и последняя полная резервная копия. Для кумулятивного резервного копирования нужна только рабочая база данных.

Обратный процесс – это восстановление базы данных из резервной копии. Для восстановления базы данных из полной резервной копии достаточно этой полной резервной копии. Для восстановления базы данных из дифференциальной резервной копии необходима эта резервная копия и полная резервная копия базы данных. Для восстановления базы данных из инкрементальной резервной копии необходима полная резервная копия и все сделанные после неё инкрементальные резервные копии. В зависимости от ситуации, предпочтительными могут быть разные варианты.

Выполнение резервного копирования

pgAdmin – это графический интерфейс для управления PostgreSQL, который предоставляет удобные инструменты для выполнения различных задач, включая администрирование баз данных, создание запросов и управление резервными копиями.

Этот метод особенно удобен для пользователей Windows, так как позволяет быстро создавать резервные копии через графический интерфейс.

1. Запустите pgAdmin и подключитесь к серверу.
2. Кликните правой кнопкой мыши на базе данных, которую хотите сохранить.
3. Выберите «Резервная копия» или «Backup», в зависимости от языка пользовательского интерфейса

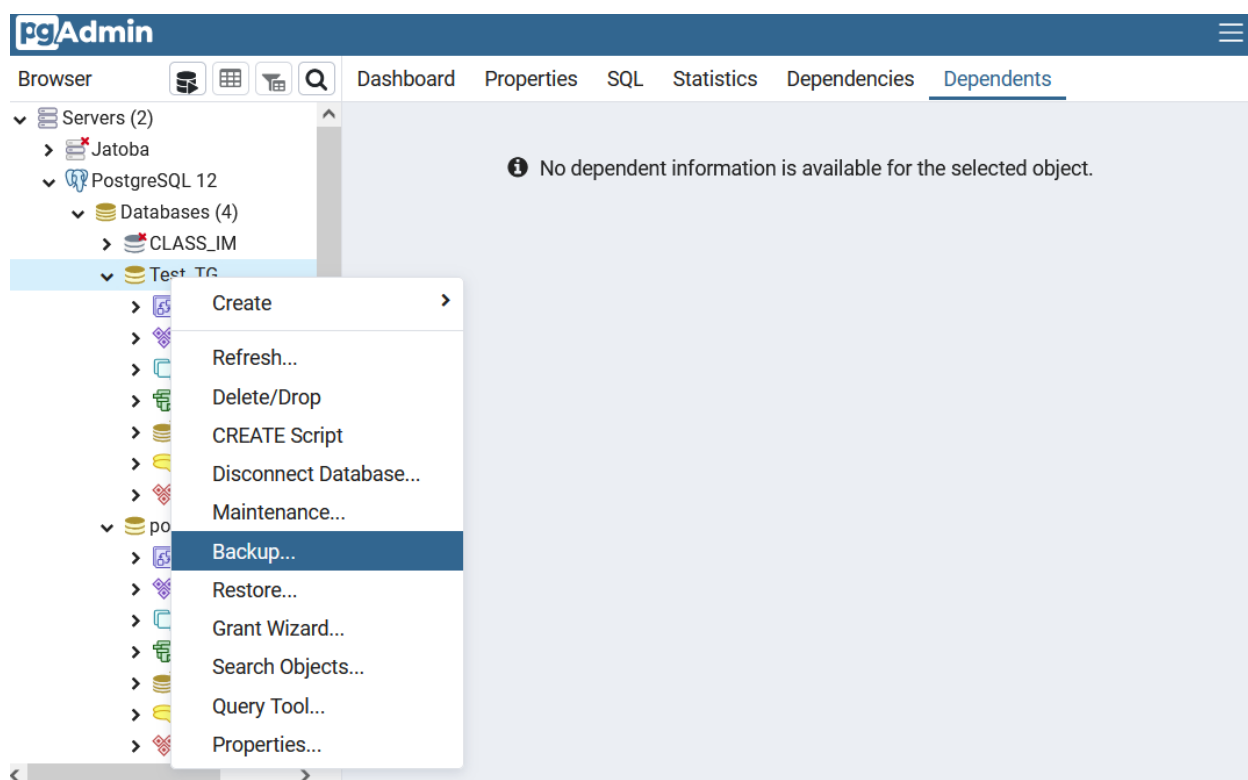


Рис.1. Меню резервного копирования в PgAdmin

4. В открывшемся окне укажите путь для сохранения резервной копии и выберите нужный формат. Самым простым форматом является Custom.
5. Нажмите «Резервная копия», дождитесь завершения процесса.
6. Найдите по тому, пути, который был указан в верхнем поле, созданный файл резервной копии.

Выполнение восстановления базы данных

1. Запустите pgAdmin и подключитесь к серверу
2. Найдите базу данных, которую вы хотите восстановить, и щёлкните по ней правой кнопкой мыши.
3. Используя интерфейс pgAdmin, отключитесь от базы данных, которую требуется восстановить. В противном случае восстановить её не удастся.
4. В появившемся меню выберите опцию «Восстановить» или «Restore», в зависимости от языка пользовательского интерфейса.
5. В открывшемся окне укажите путь к файлу с резервной копией (дампом), который вы хотите использовать для восстановления.
6. Нажмите кнопку «Восстановить» и дождитесь завершения процесса восстановления.

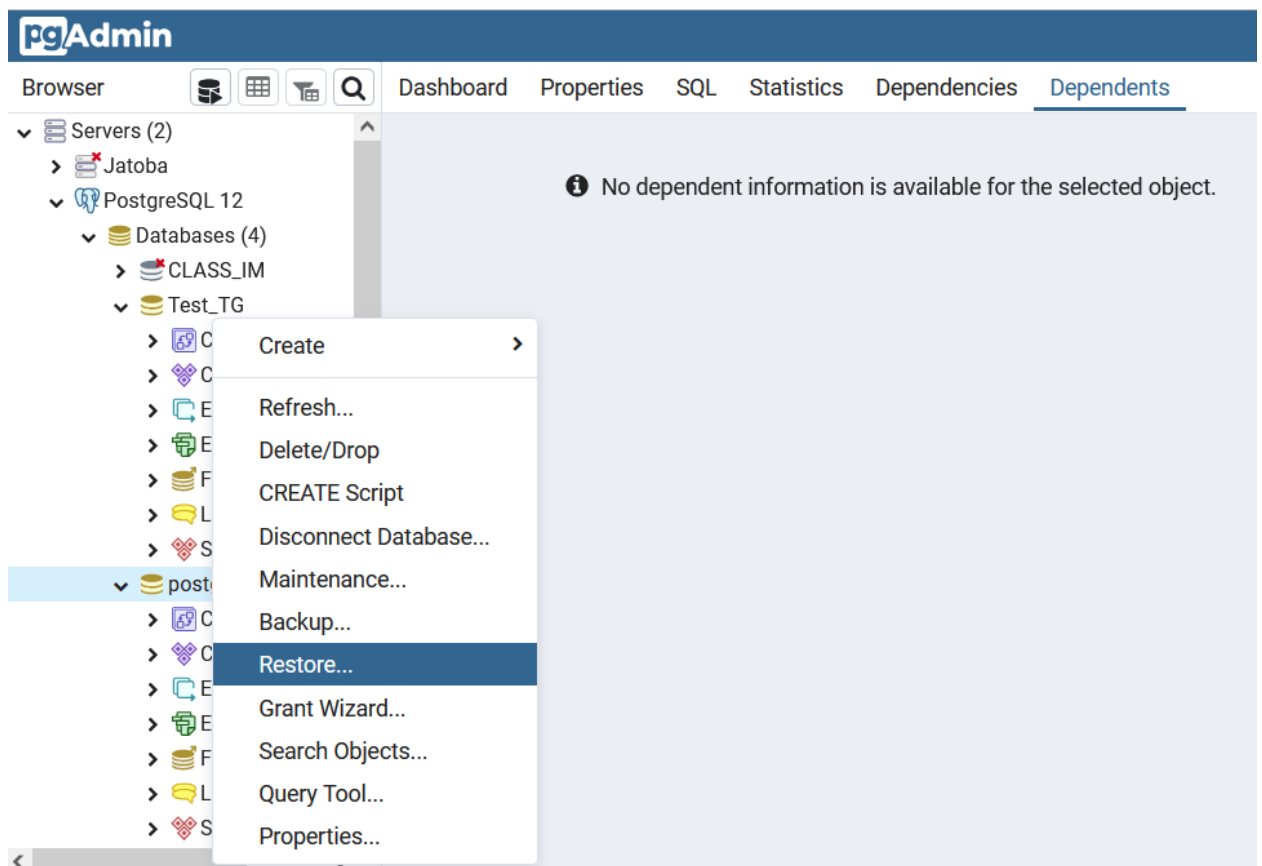


Рис. 2. Меню восстановления базы данных в PgAdmin.

Описание параметров резервного копирования

Диалог резервного копирования в pgAdmin состоит из двух вкладок. На первой вкладке находятся основные параметры.

Filename – имя файла, в который предполагается сохранить резервную копию. Справа от поля ввода находится кнопка с тремя точками, которая позволяет не вводить имя файла в текстовое поле вручную, а выбрать его из файловой системы.

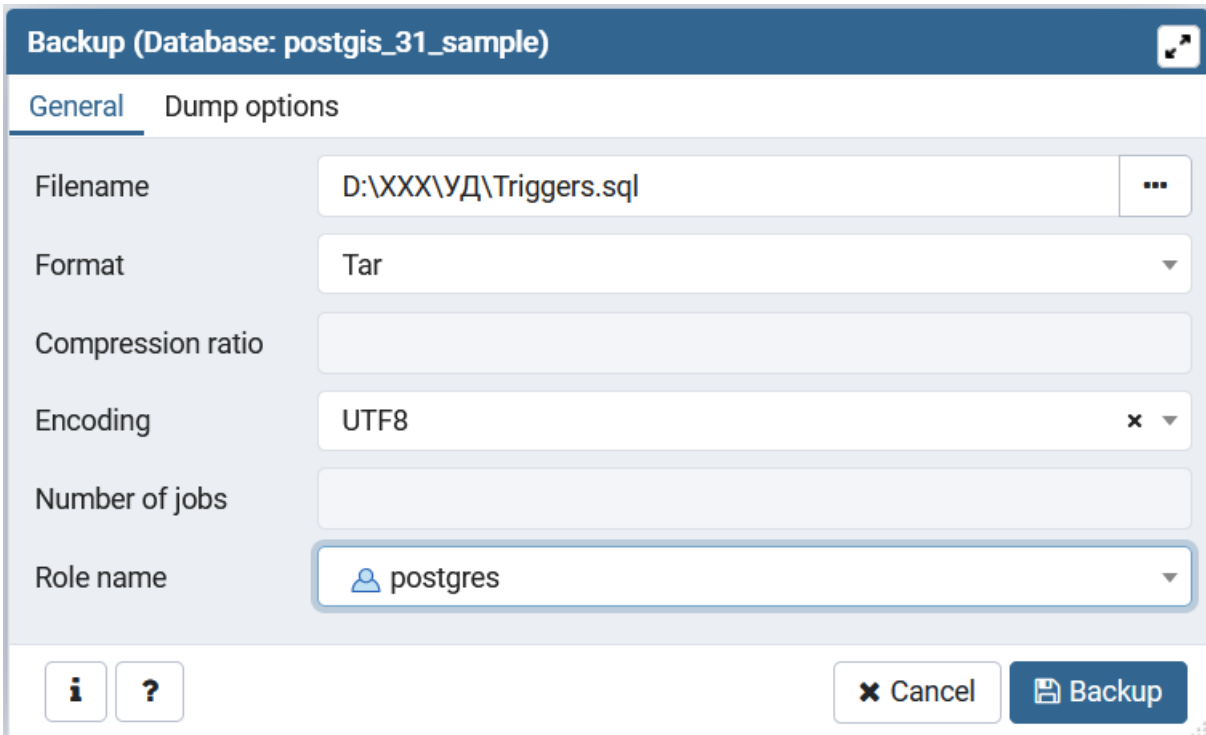
The image shows a screenshot of the 'Backup (Database: postgres_31_sample)' dialog box in pgAdmin. The 'General' tab is selected, and the 'Dump options' section is visible. The 'Filename' field contains 'D:\XXX\УД\Triggers.sql' with a browse button (three dots) to its right. The 'Format' dropdown is set to 'Tar'. The 'Compression ratio' field is empty. The 'Encoding' dropdown is set to 'UTF8'. The 'Number of jobs' field is empty. The 'Role name' dropdown is set to 'postgres'. At the bottom, there are buttons for 'Cancel' and 'Backup', along with information and help icons.

Рис. 3. Диалог резервного копирования.

Format – формат создаваемого файла. В принципе, формат файла не зависит от того расширения файла, которое было введено в поле Filename. Форматы файла в pgAdmin следующие.

- **Custom:** собственный архивный файл, который можно использовать с `pg_restore` для создания копии базы данных. Пользовательские форматы файлов архивов необходимо восстанавливать с помощью `pg_restore`. Этот формат дает возможность выбрать, какие объекты базы данных восстанавливать из файла резервной копии. Пользовательский формат архива рекомендуется для средних и больших баз данных, поскольку по умолчанию он сжимается.
- **Tar:** архивный файл tar, который можно восстановить с помощью `pg_restore`. Формат tar не поддерживает сжатие.
- **Plain:** текстовый файл сценария. Файл сценария в виде простого текста содержит инструкции и команды SQL, которые можно выполнить в командной строке `psql` для воссоздания объектов базы данных и загрузки данных таблицы. При желании файл резервной

копии в виде обычного текста можно отредактировать в текстовом редакторе перед использованием программы `psql` для восстановления объектов базы данных. Простой формат обычно рекомендуется для небольших баз данных; дампы сценариев не рекомендуются для больших двоичных объектов. Команды SQL в сценарии восстановят базу данных до последнего сохраненного состояния базы данных. Простой текстовый сценарий можно использовать для восстановления базы данных на другом компьютере или (с изменениями) на других архитектурах.

- **Directory:** архив в формате каталога, подходящий для использования с `pg_restore`. Этот формат файла создает каталог с одним файлом для каждой таблицы и выгружаемого большого объекта, а также файлом оглавления, описывающим выгруженные объекты в машиночитаемом формате, который может прочитать `pg_restore`. Этот формат по умолчанию сжат.

Compression ratio – уровень сжатия резервной копии. Укажите нулевое значение, чтобы означать отсутствие использования сжатия; укажите максимальное значение сжатия 9. Обратите внимание, что `tar`-архивы не поддерживают сжатие.

Encoding – метод кодировки символов, который следует использовать для архива.

Number of jobs – количество таблиц, которые будут одновременно создаваться в параллельном резервном копировании. Это поле не всегда доступно.

Rolename – роль, которой принадлежит резервная копия.

На второй закладке много опций, которые позволяют более гибко настраивать резервное копирование. Эти настройки разбиты на группы.

Sections – часть объекта, для которой будет выполнено резервное копирование.

- **Pre-data:** включить все элементы определения данных, не включенные в списки элементов данных или `Post-data`.
- **Data:** выполнить резервное копирование фактических данных таблицы, содержимого больших объектов и значений последовательности.
- **Post-data:** включить определения индексов, триггеров, правил и ограничений, отличных от проверенных проверочных ограничений.

Type of objects – сведения о типе объектов, для которых будет выполняться резервное копирование.

- **Only data:** ограничить резервное копирование данными.
- **Only schema:** ограничить резервное копирование объектами базы данных на уровне схемы.

- **Blobs:** включить большие объекты из резервной копии. По умолчанию эта опция включена, но, как правило, в корпоративных базах данных большую часть объёма базы данных составляют большие файлы, которые редко меняются, – документы, изображения и т.п., поэтому иногда их рекомендуется исключить.

Do not save – отключить резервное копирование для выбранных типов вспомогательных объектов базы данных.

- **Owner:** исключить команды, устанавливающие право собственности на объект.
- **Privileges:** исключить команды, создающие привилегии доступа.
- **Tablespaces:** исключить табличные пространства.
- **Unlogged table data:** исключить содержимое нерегистрируемых таблиц.
- **Publications:** исключить публикации.
- **Subscriptions:** исключить подписки.
- **Security labels:** исключить метки безопасности.

Queries – используйте эти дополнительные поля, чтобы указать тип операторов, которые должны быть включены в резервную копию.

- **Use INSERT commands:** выгрузить данные в виде операторов INSERT, а не использовать команду COPY. Обратите внимание: это может замедлить восстановление из резервной копии.
- **Use column inserts:** выгрузить данные в виде операторов INSERT и включить явные имена столбцов. Обратите внимание: это может замедлить восстановление из резервной копии.
- **Include CREATE DATABASE statement:** включить в резервную копию команду, которая создает новую базу данных при восстановлении резервной копии.
- **Include DROP DATABASE statement:** включить в резервную копию команду, которая удалит любой существующий объект базы данных с тем же именем перед воссозданием объекта во время резервного копирования.
- **Load via Partition root:** при выгрузке инструкции COPY или INSERT для секционированной таблицы нацеливаться на корень иерархии секционирования, которая ее содержит, а не на сам раздел.

Disable – указать тип операторов, которые следует исключить из резервной копии.

- **Triggers:** включить команды, которые будут отключать триггеры в целевой таблице во время загрузки данных (активен при создании резервной копии только данных).

- **\$ quoting:** включить заключение тел функций в знак доллара; если этот параметр отключен, тело функции будет заключено в кавычки с использованием стандартного строкового синтаксиса SQL.

Miscellaneous – прочие параметры резервного копирования.

- **With OID(s):** включить в резервную копию внутренние идентификаторы системных объектов.
- **Verbose messages:** указать pg_dump включить подробные сообщения.
- **Force double quote on identifiers:** принудительно заключить в кавычки все идентификаторы.
- **Use SET SESSION AUTHORIZATION:** включить оператор, который будет использовать команду SET SESSION AUTHORIZATION для определения владельца объекта (вместо команды ALTER OWNER).

Для использования более детальных параметров работы с базой данных необходимо для резервного копирования использовать утилиту командной строки pg_dump. При установке PostgreSQL она будет доступна в стандартном интерпретаторе командной строки операционной системы.

Описание параметров восстановления базы данных

Диалог восстановления базы данных в pgAdmin состоит из двух вкладок. На первой вкладке находятся основные параметры.

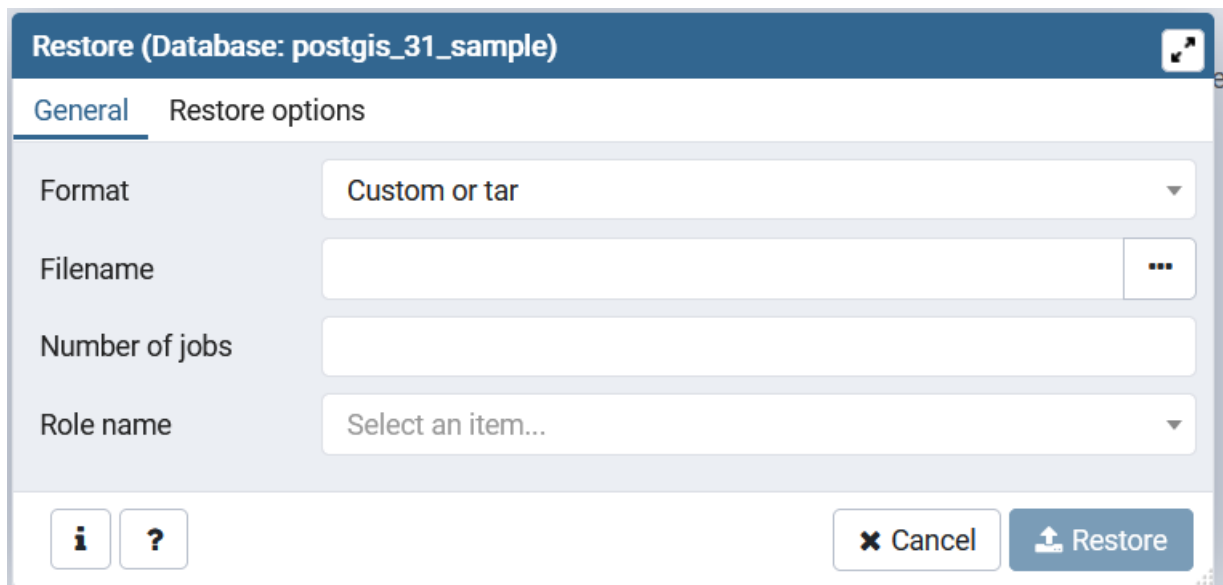


Рис 4. Диалог восстановления базы данных из резервной копии в pgAdmin.

Окно предоставляет простой способ использования резервной копии в формате Custom, tar или Directory, созданной с помощью диалогового окна резервного копирования, для воссоздания базы данных или объекта базы данных. Диалоговое окно резервного копирования вызывает параметры

клиентской утилиты `pg_dump`; Диалоговое окно восстановления вызывает параметры клиентской утилиты `pg_restore`.

Кроме использования этого диалога, можно сделать восстановление базы данных с использованием инструмента Query Tool, если резервная копия была сделана в режиме Plain и представляет собой просто SQL скрипт. Если требуется использовать формат восстановления с помощью диалога, нужно ввести параметры.

Format – формат файла резервной копии, из которой нужно восстановить базу данных. Он должен совпадать с форматом резервной копии, который был выбран при её создании.

Filename – имя файла с полным путём к резервной копии, из которой нужно восстановить базу данных. Справа от поля ввода находится кнопка с тремя точками, которая позволяет не вводить имя файла в текстовое поле вручную, а выбрать его из файловой системы.

Number of jobs – количество одновременных заданий, которое `pg_restore` должен использовать для обработки восстановления. Каждое задание использует отдельное соединение с сервером.

Role name – роль, которая будет использоваться для аутентификации на сервере во время процесса восстановления.

На второй закладке находятся более детальные параметры восстановления базы данных из резервной копии. Они разделены на группы в соответствии с предназначением.

Sections – часть объекта, которая будет восстановлена из резервной копии.

- **Pre-data:** восстановить все элементы определения данных, не включенные в списки элементов данных или Post-data.
- **Data:** выполнить восстановление фактических данных таблиц, содержимого больших объектов и значений последовательности.
- **Post-data:** восстановить определения индексов, триггеров, правил и ограничений, отличных от проверенных проверочных ограничений.

Type of objects – сведения о типе объектов, которые будут восстанавливаться из резервной копии.

- **Only data:** ограничить восстановление данными.
- **Only schema:** ограничить восстановление объектами базы данных на уровне схемы. Фактически, при этом из резервной копии можно создать пустую базу данных со всеми созданными объектами.

Do not save – отключить восстановление для выбранных типов вспомогательных объектов базы данных.

- **Owner:** исключить команды, устанавливающие право собственности на объект.
- **Privileges:** исключить команды, создающие привилегии доступа.
- **Tablespaces:** исключить табличные пространства.

- **Comments:** исключить команды, устанавливающие комментарии.
- **Unlogged table data:** исключить содержимое нерегистрируемых таблиц.
- **Publications:** исключить публикации.
- **Subscriptions:** исключить подписки.
- **Security labels:** исключить метки безопасности.
- **Table access methods:** исключить методы доступа к таблице.

Queries – используйте эти дополнительные поля, чтобы указать тип операторов, которые должны выполнены в ходе восстановления базы данных.

- **Include CREATE DATABASE statement:** выполнить команду, которая создает новую базу данных при восстановлении резервной копии.
- **Clean before restore:** удалить каждый существующий объект базы данных (и данные) перед восстановлением.
- **Include IF EXISTS:** добавить предложение IF EXISTS для удаления баз данных и других объектов. Этот параметр недоступен, если также не установлена опция «**Clean before restore**».
- **Single transaction:** выполнить восстановление как одну транзакцию (то есть обернуть выдаваемые команды в BEGIN/COMMIT). Это гарантирует, что либо все команды будут выполнены успешно, либо изменения не будут применены. Эта опция подразумевает параметр `pg_restore --exit-on-error`.

Disable – указать тип операторов, которые следует исключить из процесса восстановления базы данных.

- **Triggers:** включить команды, которые будут отключать триггеры в целевой таблице во время загрузки данных (активен при создании резервной копии только данных).
- **No data for failed tables:** игнорировать данные, которые не удалось обработать триггерами при вставке.

Miscellaneous / Behavior – прочие настройки.

- **Verbose messages:** указать `pg_restore` включить подробные сообщения.
- **Use SET SESSION AUTHORIZATION:** включить оператор, который будет использовать команду SET SESSION AUTHORIZATION для определения владельца объекта (вместо команды ALTER OWNER).
- **Exit on error:** указать `pg_restore` выйти из восстановления в случае ошибки при отправке команд SQL. По умолчанию используется продолжение и отображение количества ошибок в конце восстановления.

Задание

1. С использованием скрипта для учебной базы данных CANTEENDISHES либо своего курсового проекта развернуть в pgAdmin новую базу данных. Предпочтительным является работа со скриптом из своего курсового проекта.
2. Если в базе данных не заведены данные, завести их.
3. Выполнить резервное копирование с параметрами по умолчанию.
4. Внести изменения в данные.
5. Выполнить восстановление базы данных.
6. Убедиться, что изменения в данных были откатаны.
7. Выполнить резервное копирование в формате Plain с разными вариантами набора параметров Sections. Проанализировать, как какой параметр влияет на содержание скрипта, который при этом генерируется (хотя бы три варианта настроек).
8. Выполнить резервное копирование в формате Plain с разными настройками Type of objects: проверить, что получается в скрипте, если выбрать Only Data и Only Schema.
9. Выполнить резервное копирование в формате Directory
10. Восстановить базу данных из резервной копии в формате Directory.

Оформление отчёта

Отчёт должен иметь титульный лист с указанием ФИО и группы студента, а так же темы практической работы. После этого идёт краткое описание базы данных, с которой идёт работа (около 500 знаков, описывающих суть предметной области). Затем необходимо привести схему базы данных и несколько запросов на вставку данных, которые позволяют завести данные. Дальше там должны быть представлены либо запросы на вставку данных, либо скриншот (кому как удобнее – через SQL или через инструмент View / Edit data), на котором видно, какие данные вставлены после выполнения резервного копирования. Потом должен быть скриншот из Query Tool, на котором видно, что изменения откатаны. Затем должно быть описание, какие команды есть, а каких команд нет в зависимости от настроек параметров Sections и Type of objects. Затем должен быть скриншот с результатами восстановления базы данных из резервной копии в формате Directory.