|  |
| --- |
|  |
| МИНОБРНАУКИ РОССИИ  Федеральное государственное бюджетное образовательное учреждение  высшего образования  **«МИРЭА – Российский технологический университет»**  **РТУ МИРЭА** |

Институт комплексной безопасности и специального приборостроения

Кафедра КБ-14 «Интеллектуальные системы информационной безопасности»

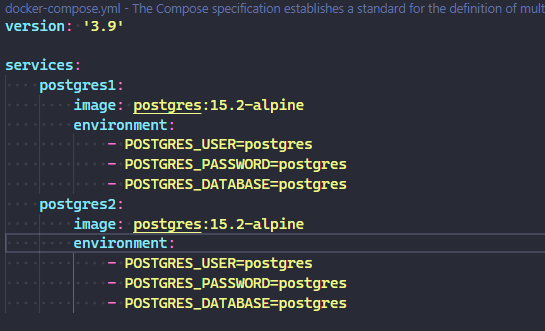
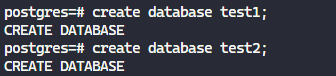
### Политики безопасности баз данных

### Практическая работа № 5

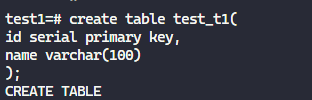
# ОТЧЁТ

# Выполнил студент группы БСБО-07-20 Любовский С.В.

# Выполнение задания 1.

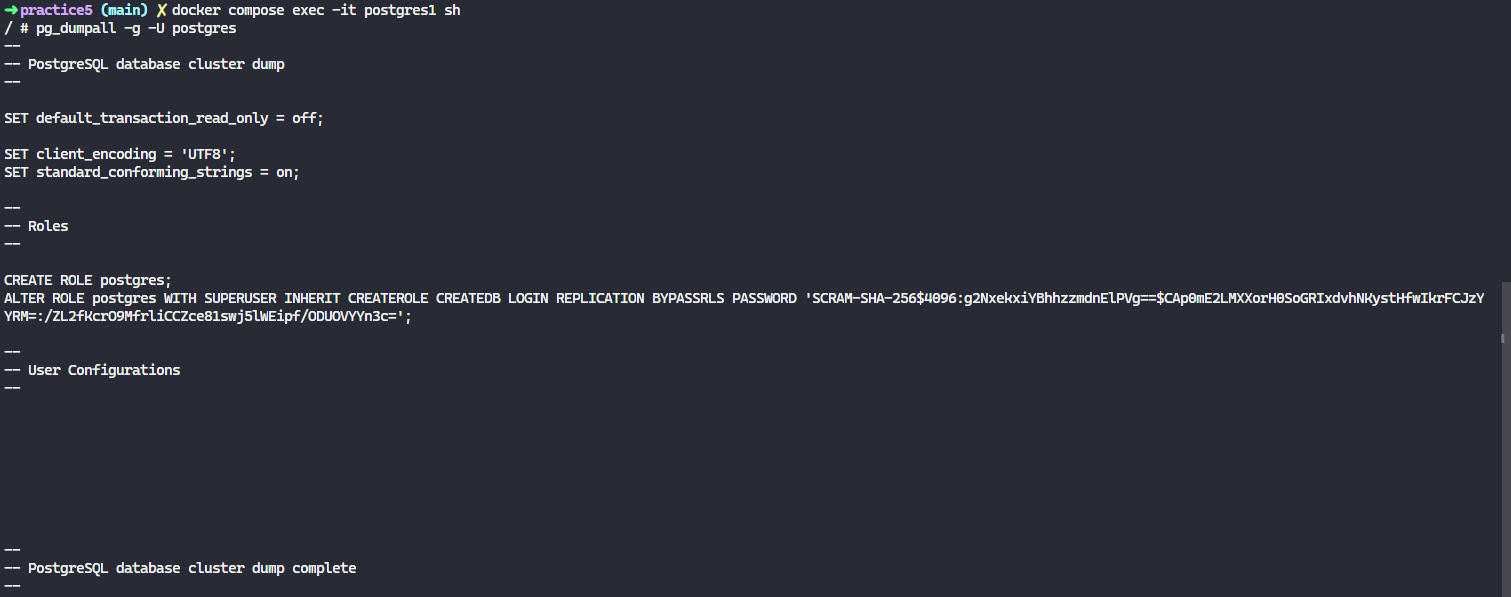
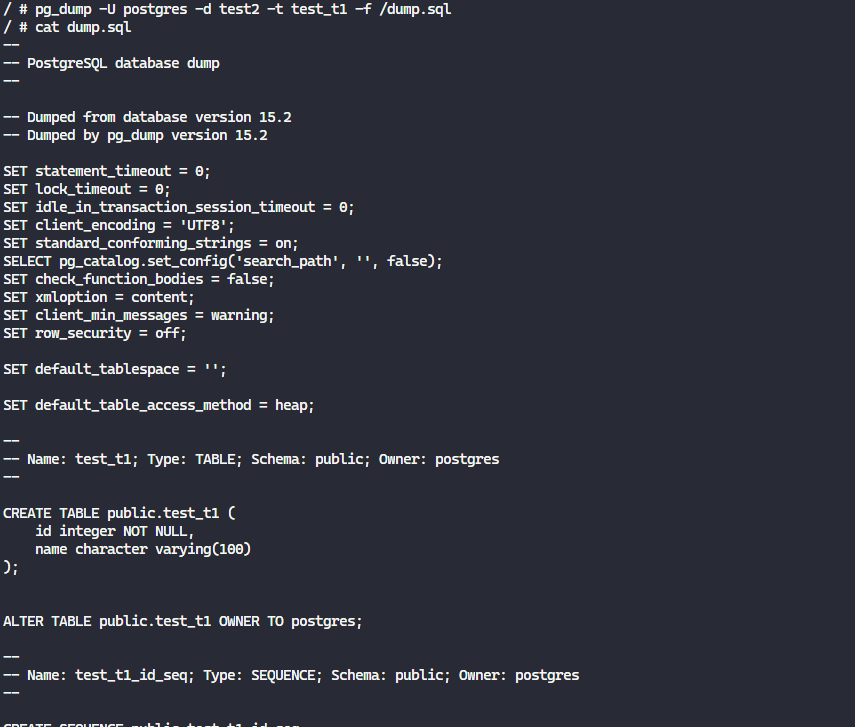
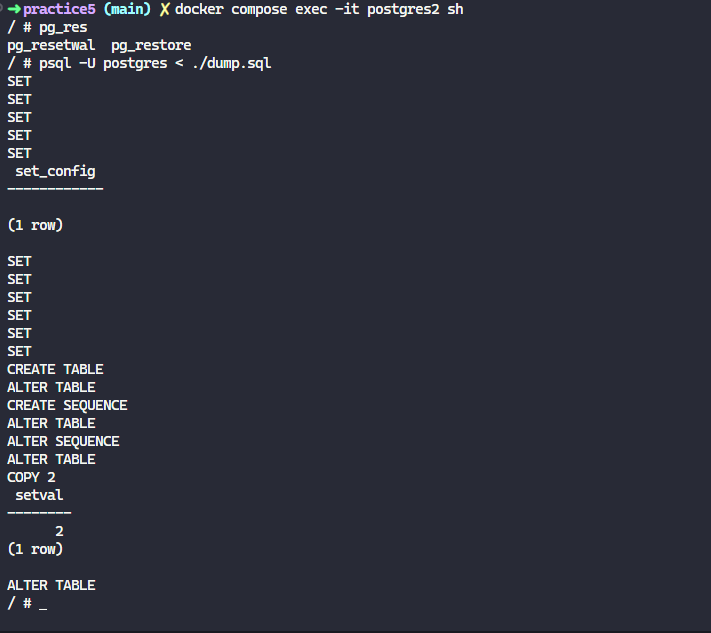
1. Создадим конфигурацию на два сервера  
   
2. Создадим тестовые таблиц на первом сервере  
   

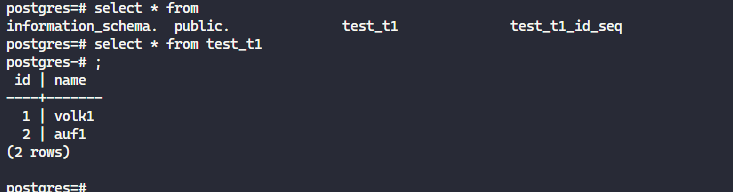
Создадим таблицу и данные в ней (для базы test1)





Тоже самое для базы test2

1. При помощи pg\_dumpall создадим дамп глобальных объектов  
   
2. При помощи pg\_dump создадим дамп только выбранных таблиц  
   
3. Используя эту резервную копию, восстановим данные на другом кластере  
     
   Проверим таблицу



# Выполнение задания 2.

1. Параметры журналирования

* log\_destination - В log\_destination указывается один или несколько методов протоколирования, разделённых запятыми. Выбран syslog как стандарт удобный для дальнейшей обработки.
* log\_directory - При включённом logging\_collector, определяет каталог, в котором создаются журнальные файлы. Использую дефолтное значение – log
* log\_filename - Значение трактуется как строка формата в функции strftime, поэтому в ней можно использовать спецификаторы % для включения в имена файлов информации о дате и времени. Использовано стандартное значение
* log\_rotation\_age - Определяет максимальное время жизни отдельного журнального файла, при включённом logging\_collector. После того как прошло заданное количество минут, создаётся новый журнальный файл. Использовано значение 1d.
* log\_rotation\_size - Определяет максимальный размер отдельного журнального файла, при включённом logging\_collector. Выбрано 100 MB.

2. Параметры резервного копирования:

* backup\_mode - Выбирает режим резервного копирования. Используется значение FULL для создания полной резервной копии, содержащей все файлы данных кластера, необходимых для его восстановления.
* archive\_mode - Когда параметр archive\_mode включён, полные сегменты WAL передаются в хранилище архива командой archive\_command. Помимо значения off (выключающего архивацию) есть ещё два: on (вкл.) и always (всегда). В обычном состоянии эти два режима не различаются, но в режиме always архивация WAL активна и во время восстановления архива, и при использовании ведомого сервера. В этом режиме все файлы, восстановленные из архива или полученные при потоковой репликации, будут архивироваться (снова). Выставлено значение on для дальнейшего использования параметра archive\_command.
* archive\_command - Команда локальной оболочки, которая будет выполняться для архивации завершённого сегмента WAL. Любое вхождение %p в этой строке заменяется путём архивируемого файла, а вхождение %f заменяется только его именем. Используется значение “tar -czvf /var/lib/postgres/bak/%f.tar.gz %p”.
* restore\_command - Команда оболочки ОС, которая выполняется для извлечения архивного сегмента файлов WAL. Этот параметр требуется для восстановления из архива, но необязателен для потоковой репликации. Используется значение “tar -xzvf /var/lib/postgres/bak/%f.tar.gz -C %p”
* recovery\_target\_timeline - Указывает линию времени для восстановления. По умолчанию производится восстановление той же линии времени, которая была текущей в момент создания базовой резервной копии. Выбрано значение current для восстановления той же линии времени, которая была текущей в момент создания базовой резервной копии.

3. Параметры безопасности паролей

* password\_encryption - Доступность различных методов аутентификации по паролю зависит от того, как пароли пользователей хешируются. Использовано значение scram-sha-256 т.к. Это наиболее безопасный из существующих на данный момент методов.
* password\_strength\_check - Модуль проверки пароля проверяет пароли пользователей всякий раз, когда они устанавливаются с помощью CREATE ROLE или ALTER ROLE. Если пароль будет сочтен слишком слабым, он будет отклонен, и выполнение команды завершится с ошибкой.

4. Параметры безопасности аутентификации

* authentication\_timeout - Устанавливает максимально допустимое время для завершения аутентификации клиента. Оставлено значение по умолчанию.
* ssl\_cert\_file - отправляется клиенту для идентификации сервера. (aka Публичный ключ)
* ssl\_key\_file – используется для дешифровки на стороне сервера (aka приватный ключ)

5. Параметры безопасности доступа к файлам

* data\_directory – Каталог с файлами данных. Можно установить только при первом запуске.
* data\_file\_mode – Права на файлы данных.
* data\_directory\_mode – Права на директории с данными.

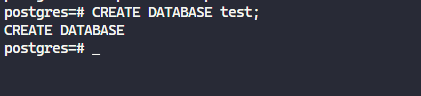
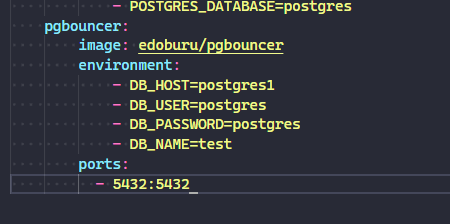
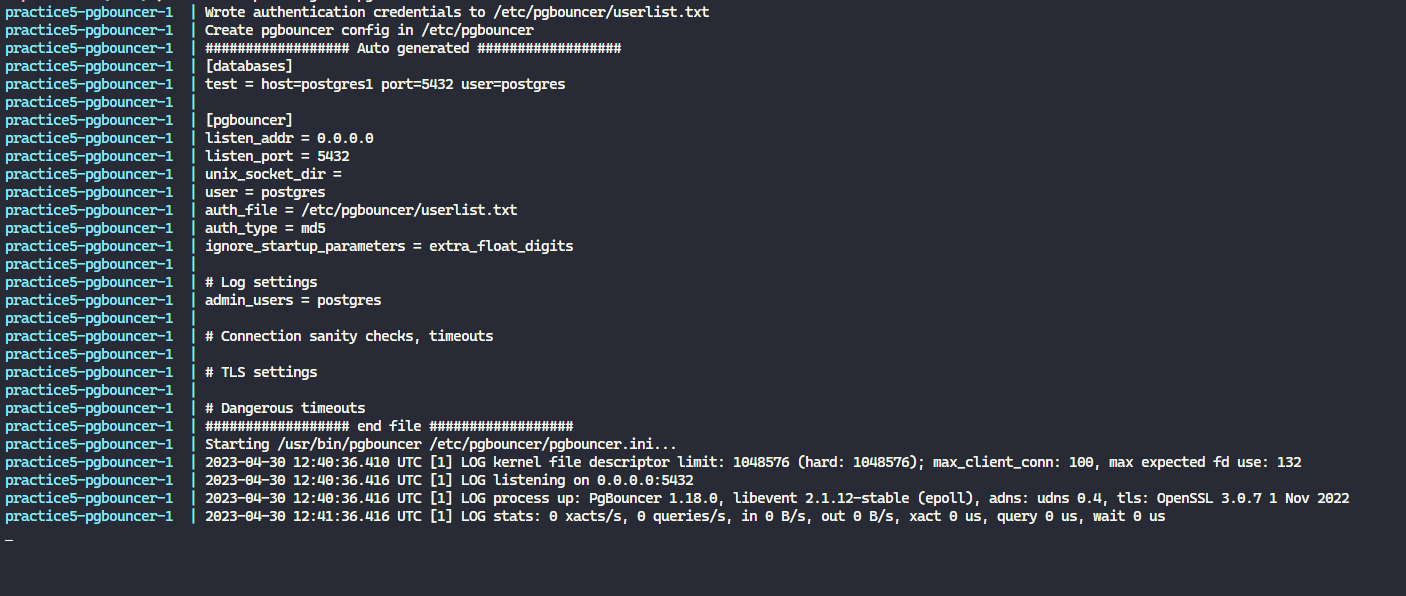
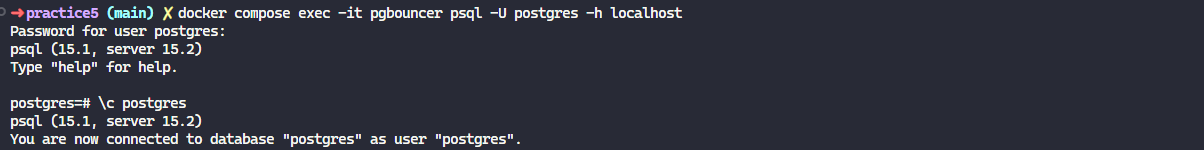
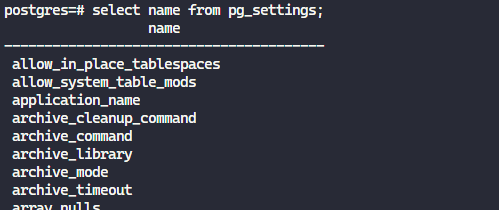
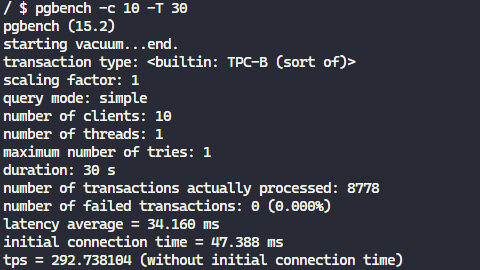
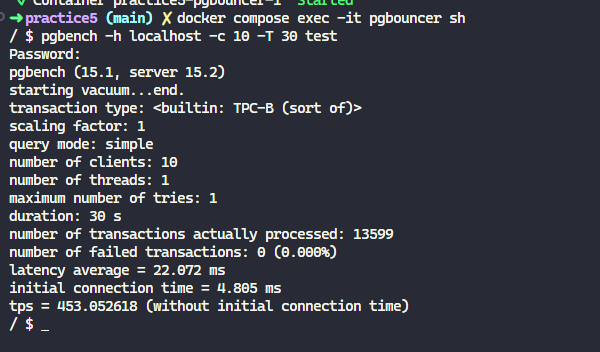
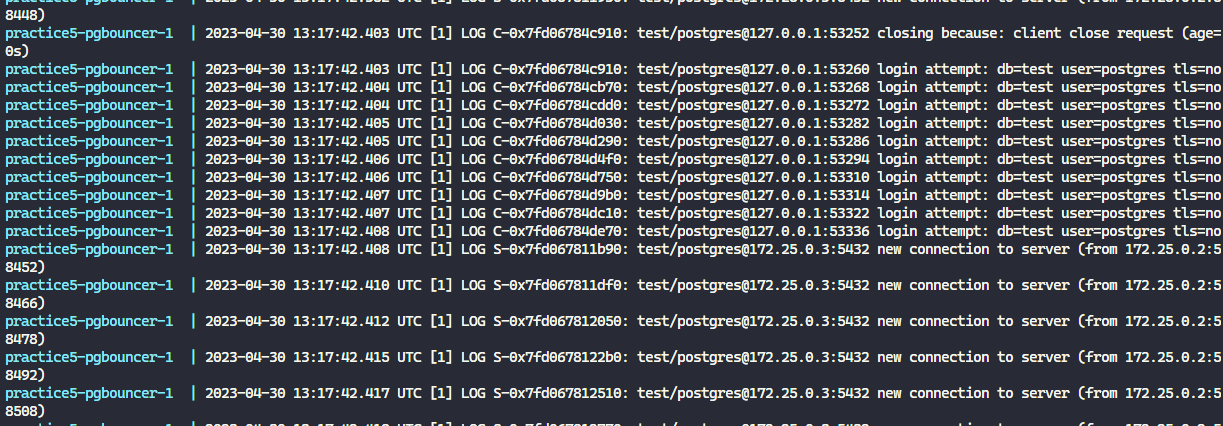
6. Параметры безопасности запросов

* max\_statement\_time - Если установлено ненулевое значение, любые запросы, которые занимают больше времени в секундах, будут прерваны. Значение по умолчанию равно нулю, и в этом случае ограничения не применяются.
* statement\_timeout - Задаёт максимальное время выполнения оператора (в миллисекундах), начиная с момента получения сервером команды от клиента, по истечении которого оператор прерывается. Установлен дефолт – 0.

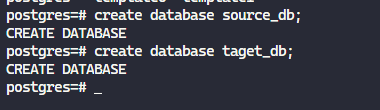
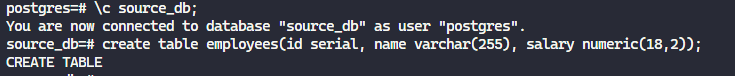
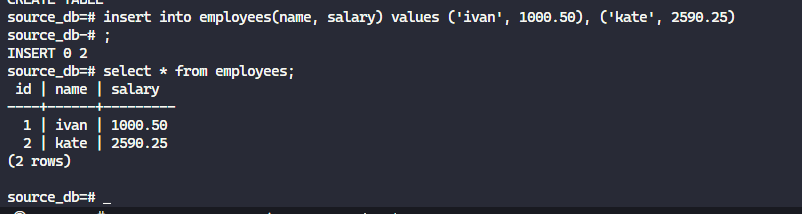
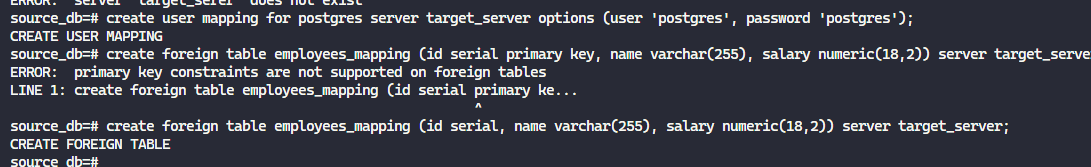
7. Параметры безопасности сеанса

* idle\_in\_transaction\_session\_timeout - Завершать любые сеансы, в которых открытая транзакция простаивает дольше заданного (в миллисекундах) времени. Это позволяет освободить все блокировки сеанса и вновь задействовать слот подключения; также это позволяет очистить кортежи, видимые только для этой транзакции. Установлено значение 0, с целью отключения данного функционала
* lock\_timeout - Задаёт максимальную длительность ожидания (в миллисекундах) любым оператором получения блокировки таблицы, индекса, строки или другого объекта базы данных. Если ожидание не закончилось за указанное время, оператор прерывается. Это ограничение действует на каждую попытку получения блокировки по отдельности и применяется как к явным запросам блокировки (например, LOCK TABLE или SELECT FOR UPDATE без NOWAIT) так и к неявным. При значении, равном нулю (по умолчанию), этот контроль длительности отключается.

# Выполнение задания 3.

1. Создадим тестовую базу данных  
   
2. Добавим pgbouncer в конфигурацию  
   
3. Запустим и проверим логи  
   
4. Подключитесь к базе данных через PgBouncer с помощью утилиты psql и выполните несколько простых SQL-запросов.  
     
   
5. Измените настройки PgBouncer для повышения производительности. Было увеличено кол-во подключений до 5000
6. Проверим бенчмарк в прямом подключении к базе и через pgbouncer  
   Сначала напрямую  
     
   Теперь через pgbouncer  
     
   Видно что он лучше справляется с множеством подключений
7. В логах можно найти подключения и отключения клиентов  
   

# Выполнение задания 4.

1. Создадим две БД  
   
2. В source\_db создадим таблицу employees  
   
3. Добавим пару записей  
   
4. Установим расширение postgres\_fdw  
   
5. Создадим сервер  
   
6. Создадим маппинг на внешнем сервере  
   
7. Выполним запрос ко внешней таблице  
   