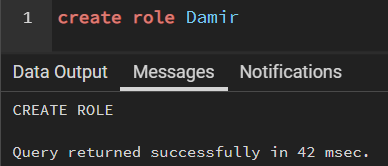
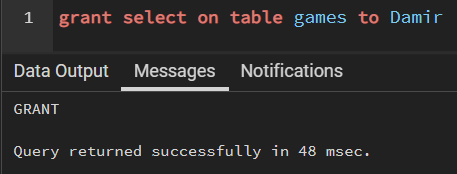
**Цель:** Ознакомиться с особенностями ролей в Postgres и научиться ими пользоваться

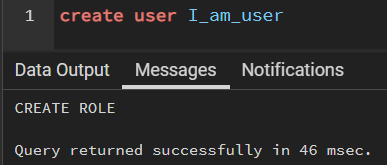
**Упражнение 19.1.** Создайте роль для доступа на чтение к демонстрационной базе данных без права создания сеансов работы с сервером БД.



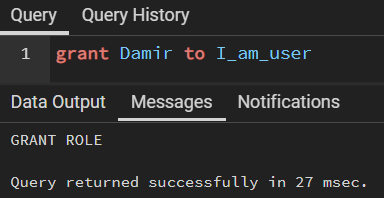
Дадим права на чтение таблицы games



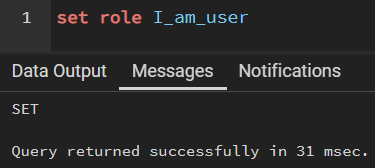
**Упражнение 19.2.** Создайте пользователя сервера БД и предоставьте ему привилегию использования роли, созданной в предыдущем упражнении. Проверьте, что этот пользователь может выполнять любые запросы на выборку из таблиц демонстрационной базы данных, но не может их обновлять.



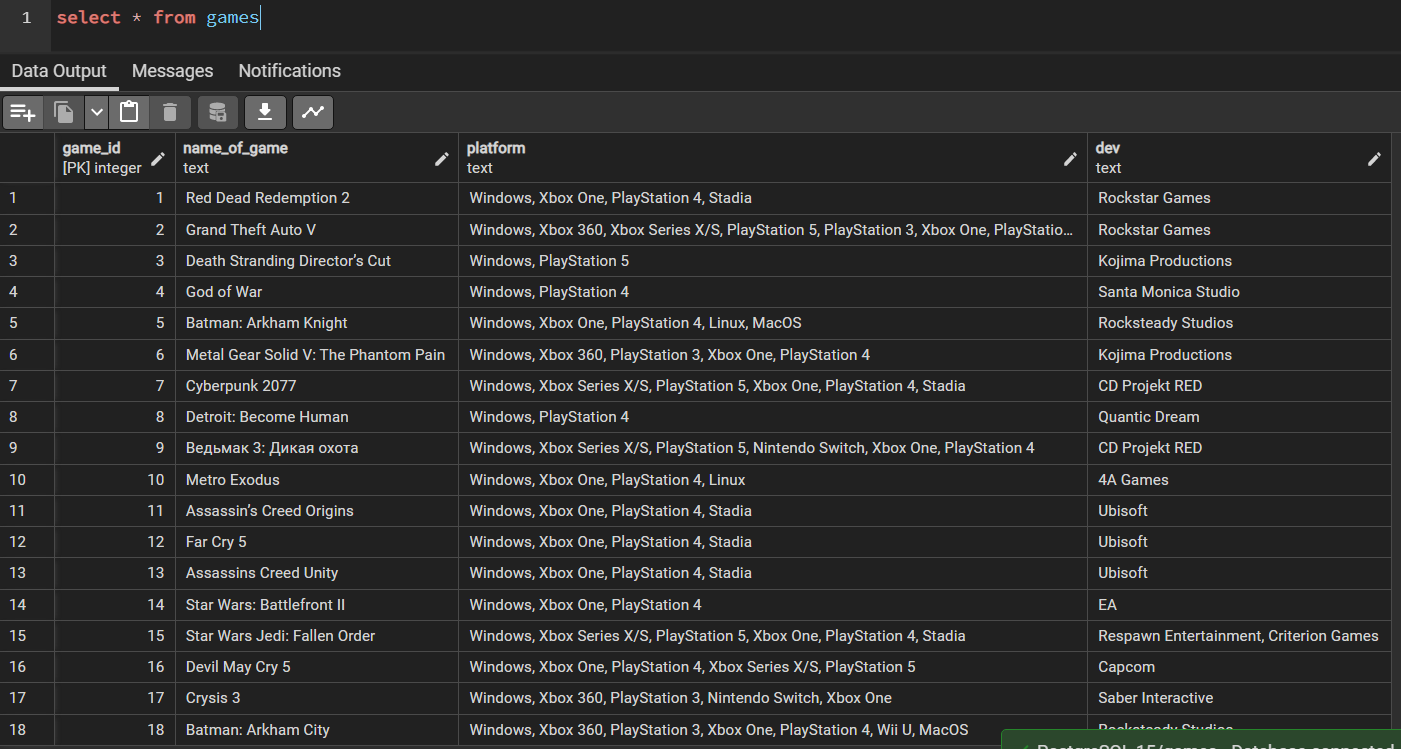
Передача роли



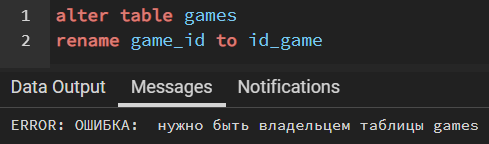
Переключение на пользователя I\_am\_user



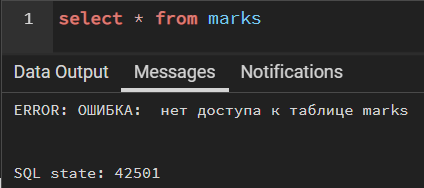
Проверка возможностей пользователя



Из-за недостатка прав, выдает ошибку



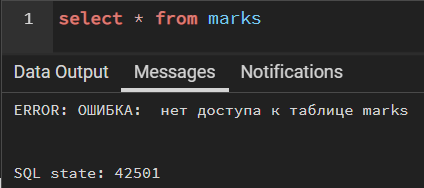
Так же не удается читать другие таблицы



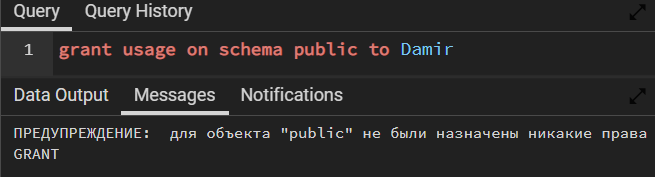
**Упражнение 19.3.** Постройте пример, показывающий, что для доступа к таблицам схемы необходимо также предоставить право использования (USAGE) этой схемы.

В моей базе данных все таблицы принадлежат схеме public

Так как у всех таблиц, схема public и при попытке пользователя запросить из таблицы, к которой у него нет прямого доступа, выдаётся ошибка:



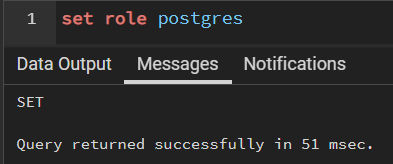
Теперь попробуем представить право использования этой схемы и проверить работоспособность

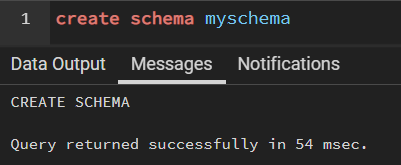


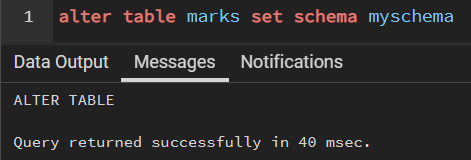
**[USAGE]** разрешает доступ к объектам, содержащимся в указанной схеме (при условии, что собственные требования к привилегиям объектов также выполняются).

Так как у нас роли Damir предоставлены разрешения только на чтение таблицы games c схемой public, СУБД не разрешит пользователю с такой ролью обращаться к другим таблицам. Поэтому создадим новую схему myschema и установим её для таблицы marks:

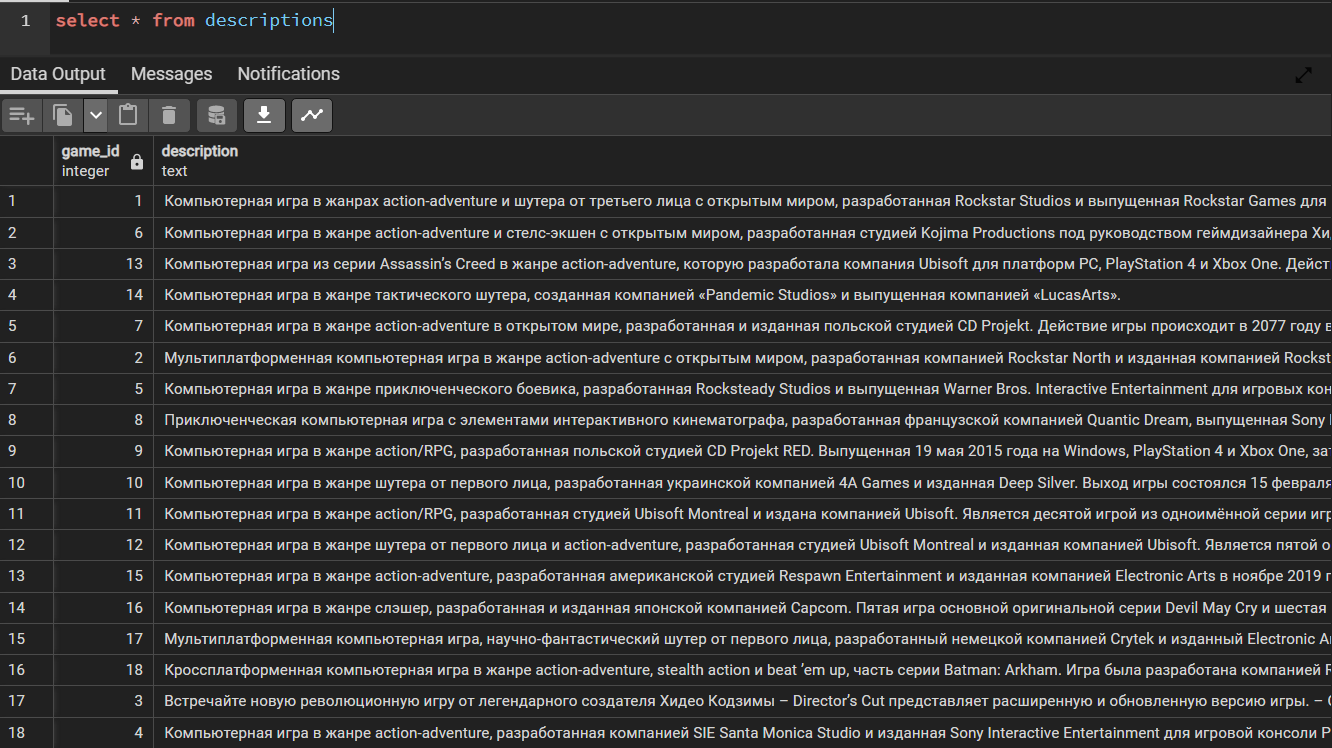
Меняем роль, ибо та не даст создать



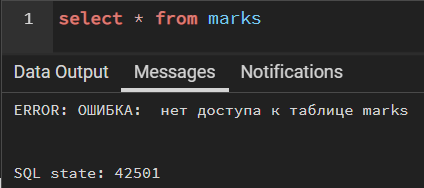




Теперь у пользователя есть доступ к другим таблицам так как они в public

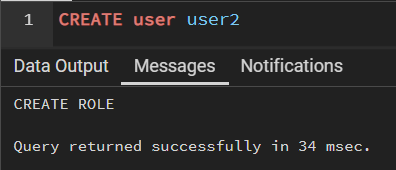
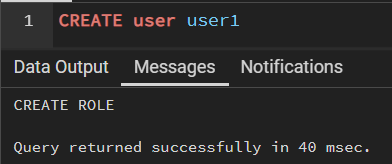


Но marks посмотреть не сможет



**Упражнение 19.4.** Реализуйте схему разграничения доступа на уровне строк для пользователей уровня приложения (не зарегистрированных как пользователи на сервере базы данных).

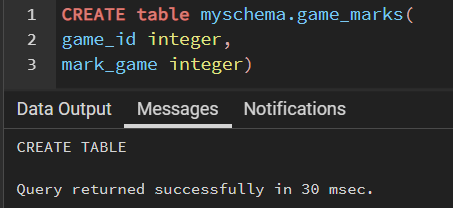
Для начала создадим 2-х новых пользователей:



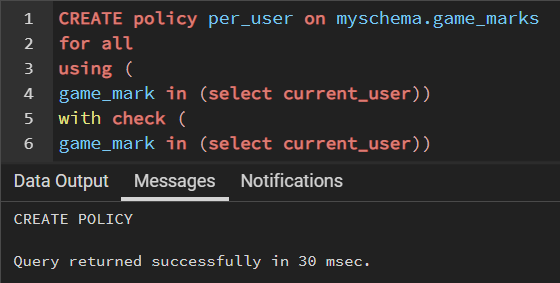
И создадим новую таблицу в схеме myschema со списком игр с полями:

Game\_id – id игры

Mark\_game – оценка игры

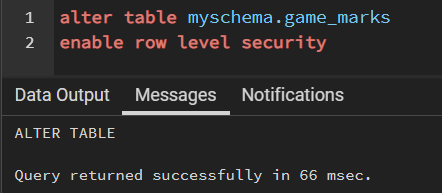


Мы определим одну политику, которая будет применяться для всех операций на таблице game\_marks:

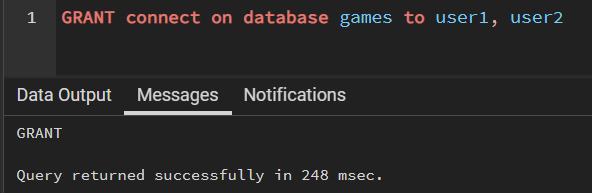
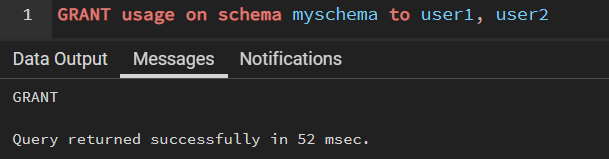
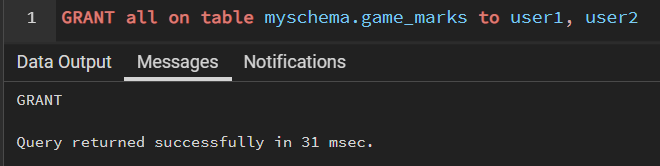


То есть, если один из пользователей пройдёт проверку, то он получит полный доступ к таблице game\_marks.

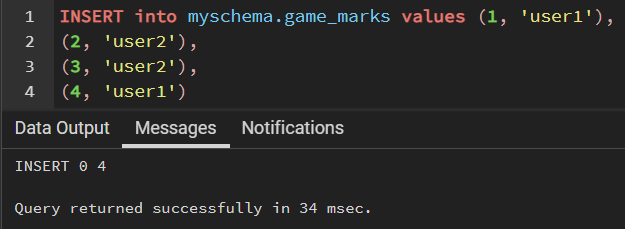
Для того чтобы политики, определенные в таблице, использовались, необходимо активировать безопасность на уровне строк для этой таблицы:



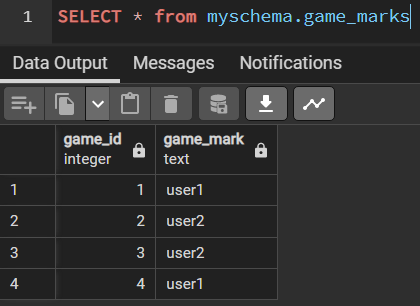
Далее необходимо разрешить пользователям создавать сеансы для работы с базой данных и разрешить доступ к схеме myschema и к таблице game\_marks, находящейся в этой схеме:

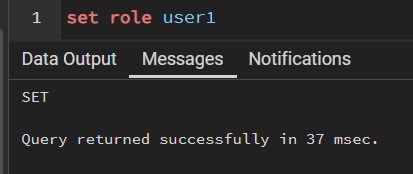
Чтобы убедиться, что владелец таблицы customs имеет доступ ко всем строкам этой таблицы независимо от работы средств разграничения доступа, добавим в эту таблицу несколько строк:



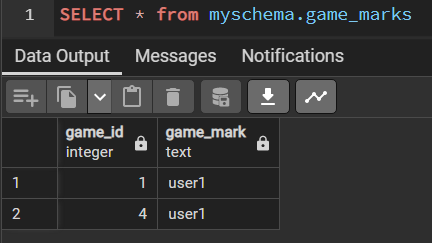
Выборка данных от имени владельца таблицы показывает, что все добавленные строки доступны:



Теперь подключимся к базе данных от имени пользвователя user1:

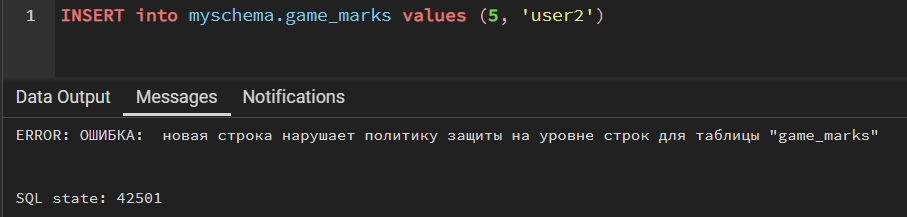


Попробуем вывести данные из таблицы customs:



Как можем видеть, СУБД выводит строки, только с совпадение game\_mark и имени пользователя.

Если мы захотим от имени user1 добавить строки с именем другого пользователя СУБД выдаст ошибку:



**Упражнение 19.5**. Реализуйте разграничение доступа на уровне строк, обеспечивающее три уровня:

1) совершенно секретный доступ;

2) секретный доступ;

3) открытый доступ.

При этом пользователь, имеющий доступ к более высокому уровню, должен также иметь доступ и к более низким. Нескольким пользователям сопоставьте различные уровни доступа из упомянутого списка (с помощью вспомогательных таблиц базы данных). Убедитесь, что разграничение действительно работает.

Для начала определим возможности типов доступов:

1 – совершенно секретный доступ – все возможности.

2 – секретный доступ – чтение всех строк и добавление

3 – открытый доступ – только чтение.

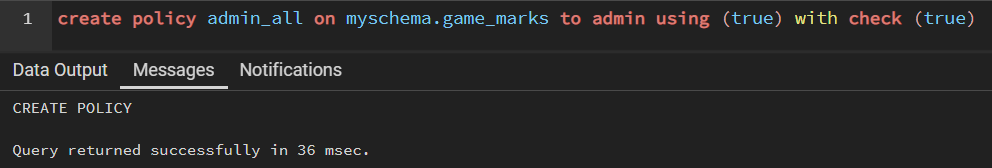
Создадим для каждого уровня роль:

1 – admin

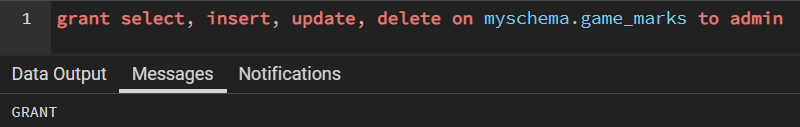
2 – moder

3 – guest

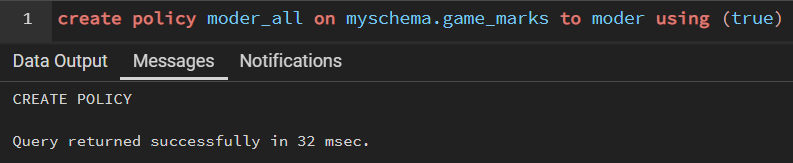
Создал политику дающую, пользователи с ролью admin полный доступ к таблице myschema.game\_marks:



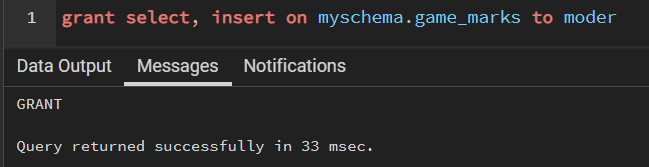
Дадим этой роли все возможности:



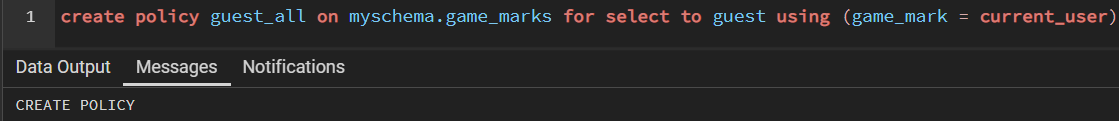
Создал политику дающую, пользователи с ролью model доступ просмотра всех строк к таблице myschema.game\_marks:



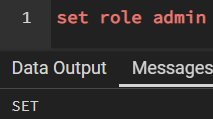
Дадим этой роли возможности читать все строки и добавлять:



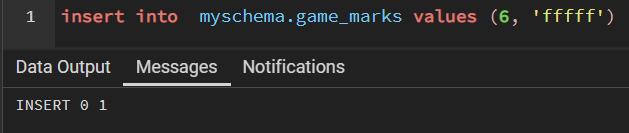
Создадим политику дающую обычным пользователям видеть только строки:



Переключимся на роль admin:

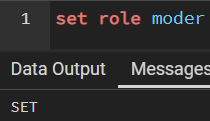


Мы можем выводить и изменять строки:

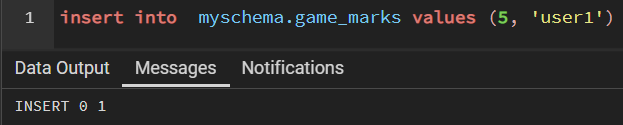
 

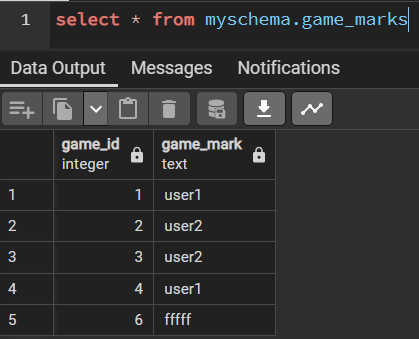
Переключимся на роль moder:

Можем выводить:

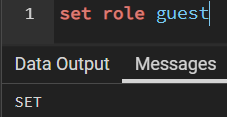


Можем добавлять:

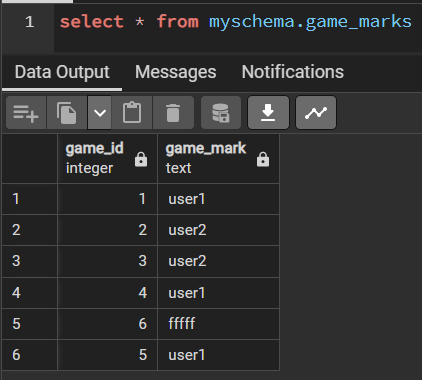




Переключимся на роль guest:



Смотреть можно



Трогать нет

