



Привилегии



Доступ к объектам БД

Виды привилегий для разных объектов

Выдача и отзыв привилегий и право перевыдачи

Привилегии по умолчанию

Просмотр привилегий

Примеры управления доступом

Владелец объекта

роль, создавшая объект

может быть изменен (`ALTER OBJECT OWNER TO new_role`)

Доступ к объекту имеют

суперпользователь — полный

владелец объекта — полный, но может быть ограничен

другие роли — в рамках выданных привилегий

Право выдачи и отзыва привилегий имеют

суперпользователь

владелец

роль, получившая привилегию с указанием `with grant option`

Виды привилегий

Для таблиц

SELECT	чтение данных
INSERT	вставка строк
UPDATE	изменение строк
DELETE	удаление строк
TRUNCATE	очистка таблицы
REFERENCES	ссылка на таблицу во внешнем ключе
TRIGGER	создание триггеров на таблице

Для представлений

SELECT	чтение данных
TRIGGER	создание триггеров на представлении

Для функций

EXECUTE	выполнение
---------	------------

Виды привилегий

Для последовательностей

SELECT	использование currval
UPDATE	использование nextval и setval
USAGE	использование currval и nextval

Для табличных пространств

CREATE	создание объектов в табличном пространстве
--------	--

Для баз данных

CREATE	создание схем в базе данных
CONNECT	подключение
TEMPORARY	создание временных таблиц

Для схем

CREATE	создание объектов в схеме
USAGE	доступ к объектам в схеме

Пример выдачи

```
GRANT SELECT,UPDATE ON [TABLE] tbl TO role;
```

разрешить чтение и изменение таблицы *tbl* для роли *role*

```
GRANT EXECUTE ON ALL FUNCTIONS IN SCHEMA scm TO role;
```

разрешить выполнение всех функций в схеме *scm* для роли *role*

Пример отзыва

```
REVOKE SELECT,UPDATE ON [TABLE] tbl FROM role;
```

запретить чтение и изменение таблицы *tbl* для роли *role*

```
REVOKE EXECUTE ON ALL FUNCTIONS IN SCHEMA scm FROM role;
```

запретить выполнение всех функций в схеме *scm* для роли *role*

Роль получает привилегии

непосредственно (в том числе как владелец)

через групповые роли, в которые она входит

через псевдороль `public`, в которую входят все роли

Получение привилегий зависит от атрибута `inherit`

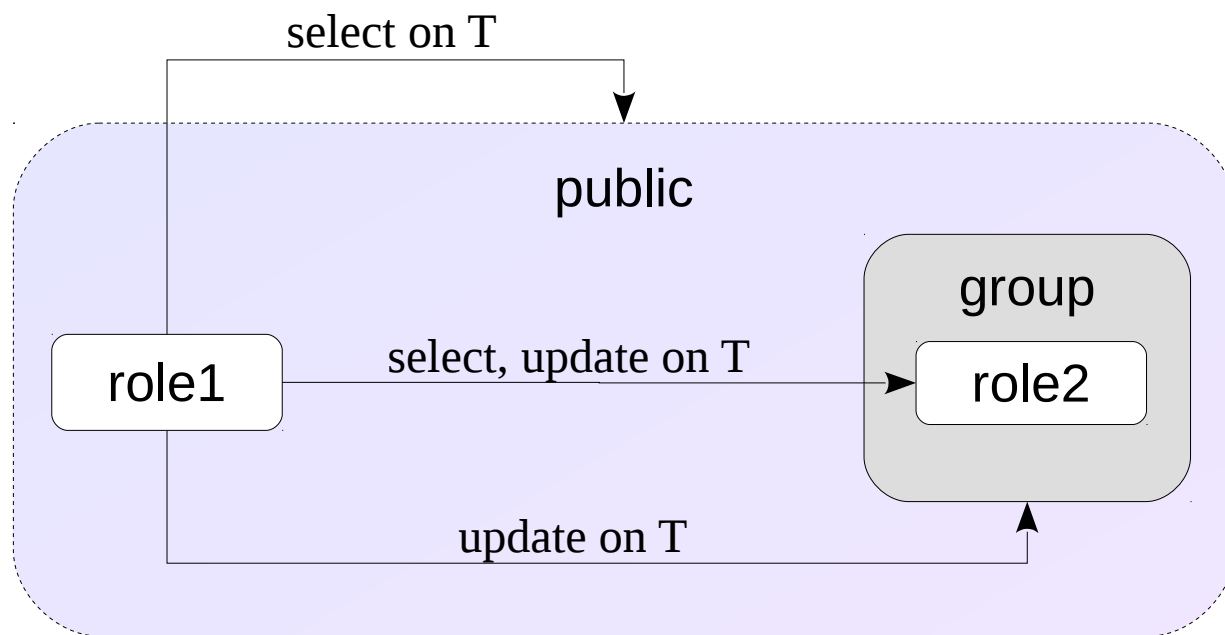
```
CREATE ROLE role [INHERIT];
```

автоматически добавляются
привилегии, которыми обладает
групповая роль

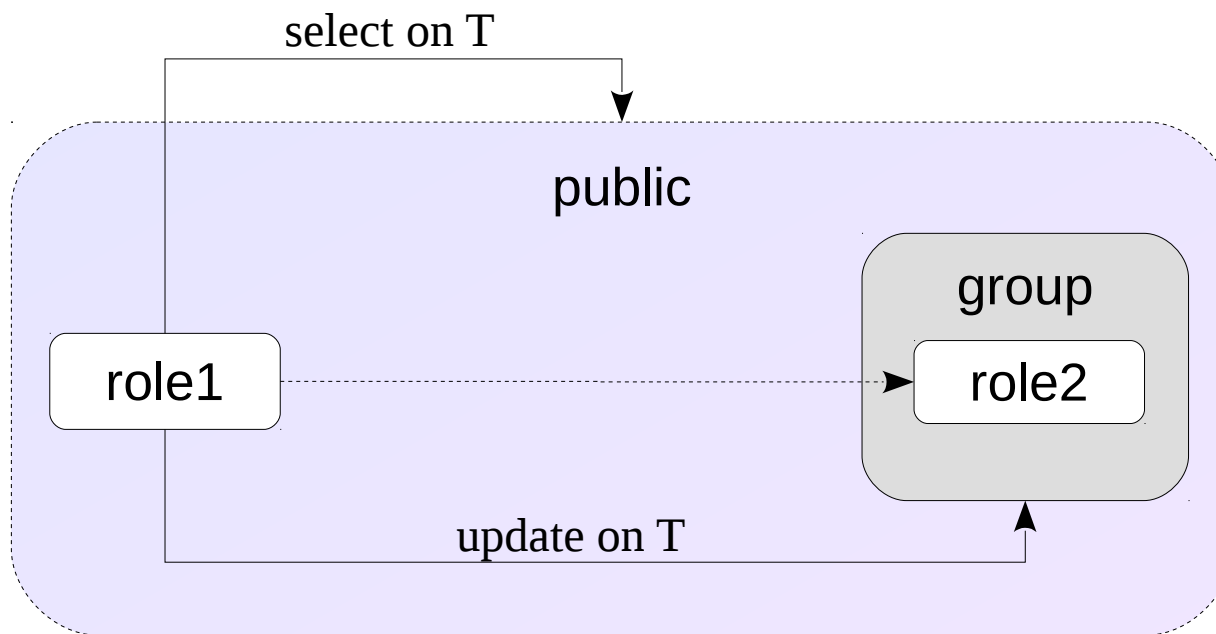
```
CREATE ROLE role NOINHERIT;
```

привилегии группы доступны
только при явном переключении:
`SET ROLE group;`

Если *role1* выполнит
REVOKE SELECT, UPDATE ON *t* FROM *role2*,
какие привилегии останутся у *role2*?



Если *role1* выполнит
REVOKE SELECT, UPDATE ON *t* FROM *role2*,
у *role2* все равно останутся обе привилегии.



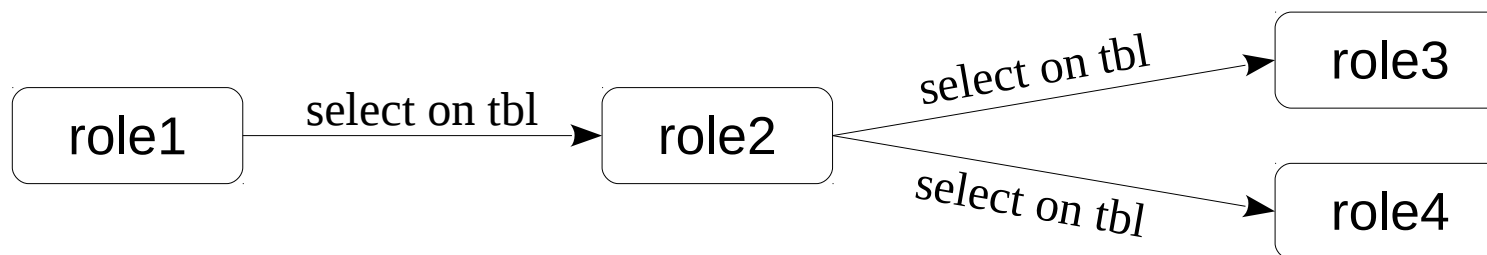
Выдача привилегии с правом перевыдачи

```
role1: GRANT SELECT ON tbl TO role2 WITH GRANT OPTION;
```

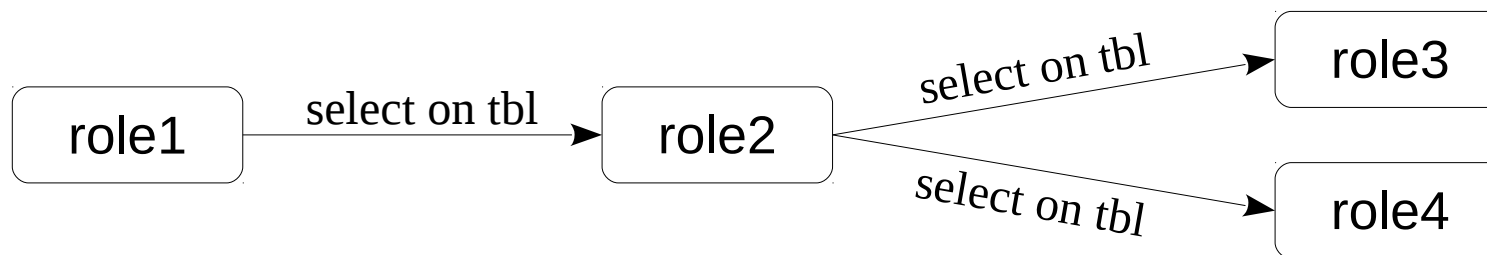
теперь и *role2* может передавать права

```
role2: GRANT SELECT ON tbl TO role3 WITH GRANT OPTION;
```

```
role2: GRANT SELECT ON tbl TO role4 WITH GRANT OPTION;
```



Одна и та же привилегия может быть независимо выдана несколькими ролями



Отзыв самой привилегии

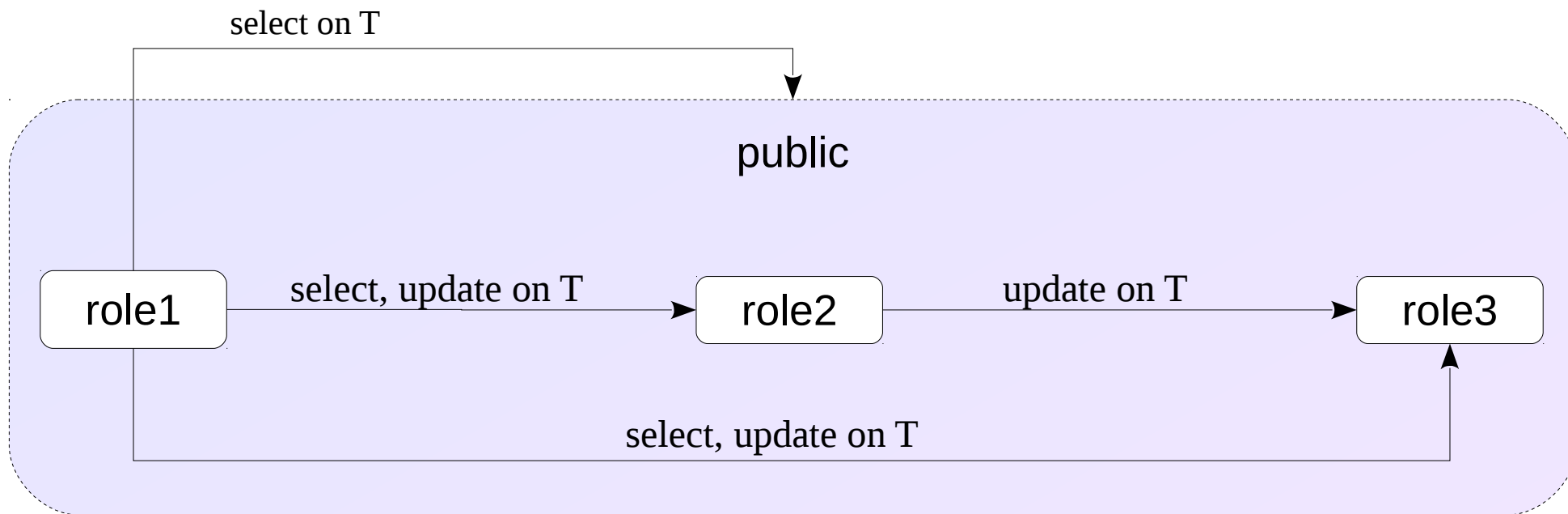
```
role1: REVOKE SELECT ON tbl FROM role2 CASCADE;
```

Отзыв права передачи

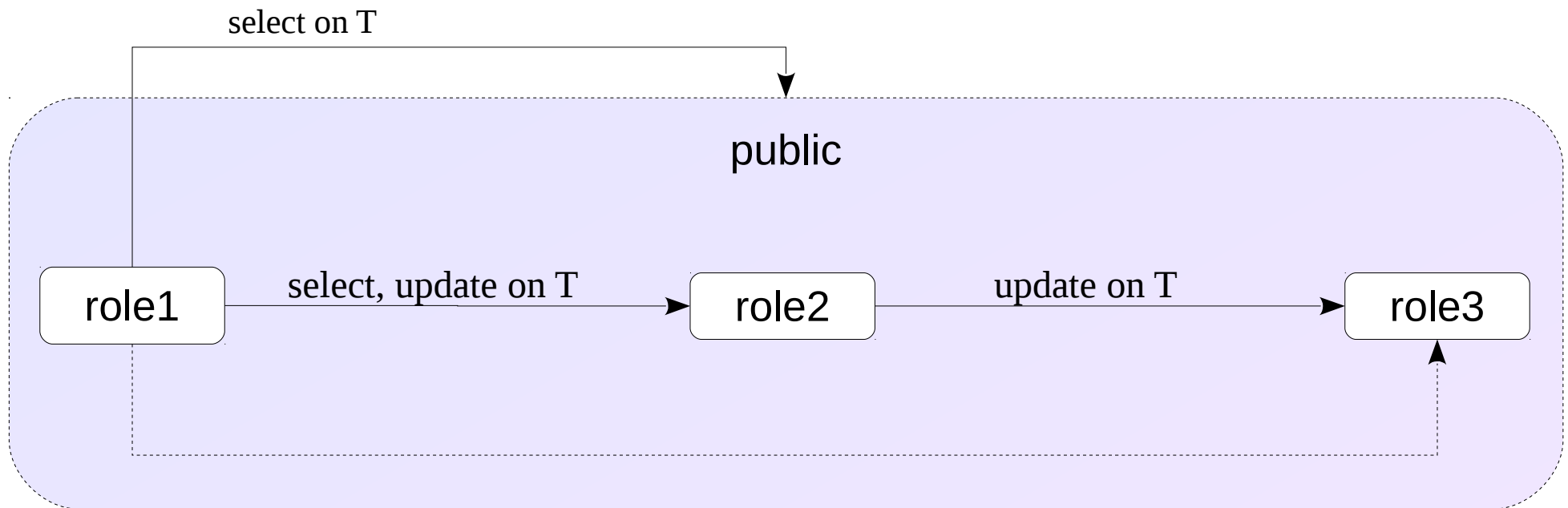
```
role1: REVOKE GRANT OPTION FOR  
SELECT ON tbl FROM role2 CASCADE;
```

Если роль переывдавала привилегию другим ролям,
указание CASCADE обязательно

Если *role1* выполнит
REVOKE SELECT, UPDATE ON *t* FROM *role3*,
какие привилегии останутся у *role3*?



Если *role1* выполнит
REVOKE SELECT, UPDATE ON *t* FROM *role3*,
у *role3* все равно останутся обе привилегии.



По умолчанию роль `public` получает ряд привилегий

для баз данных

`CONNECT` (подключение)

`TEMPORARY` (создание временных таблиц)

для схемы `public`

`CREATE` (создание объектов)

`USAGE` (доступ к объектам)

для схем `pg_catalog`
и `information_schema`

`USAGE` (доступ к объектам)

для функций

`EXECUTE` (выполнение)

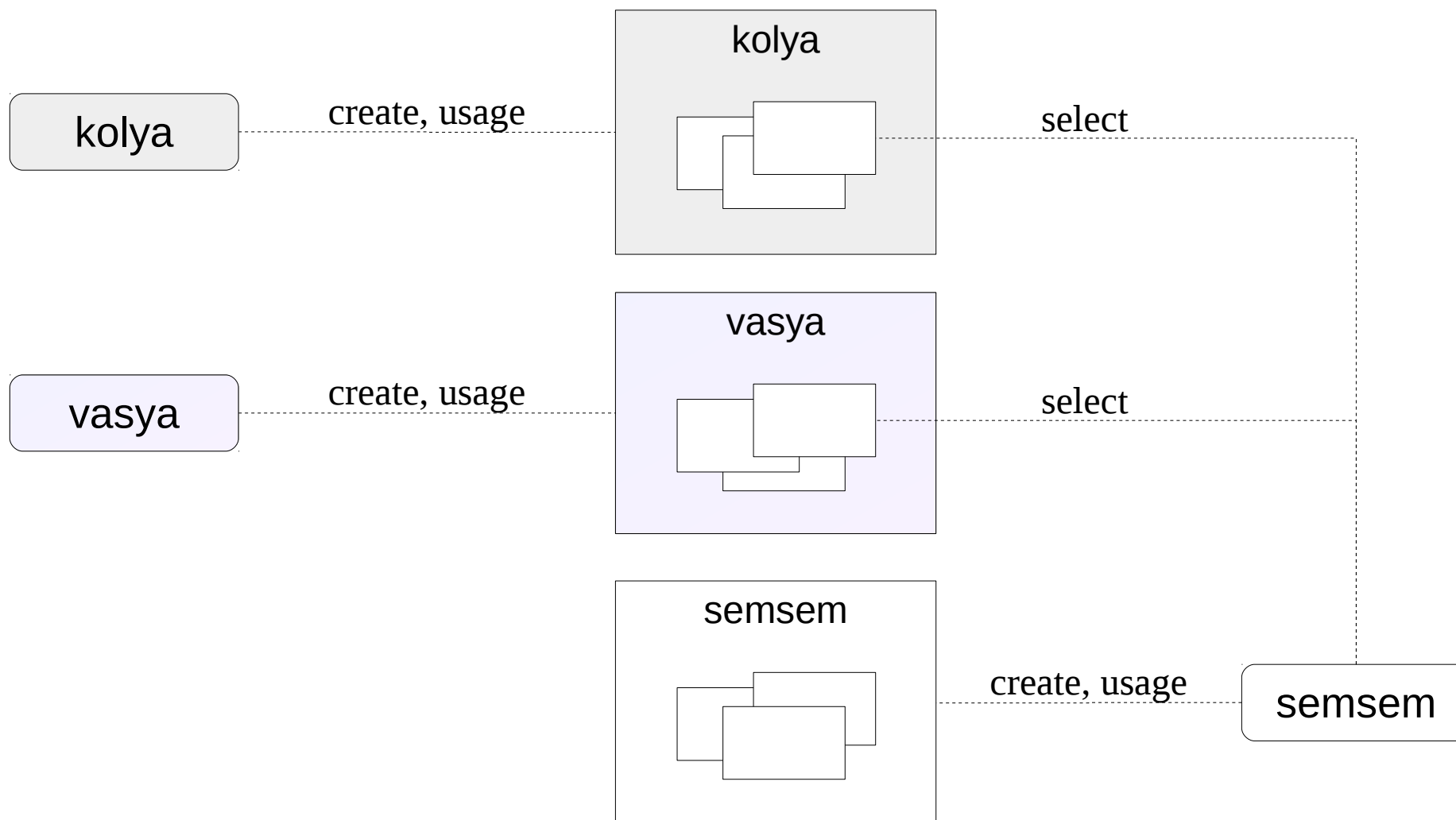
Дополнительные привилегии по умолчанию

`ALTER DEFAULT PRIVILEGES`

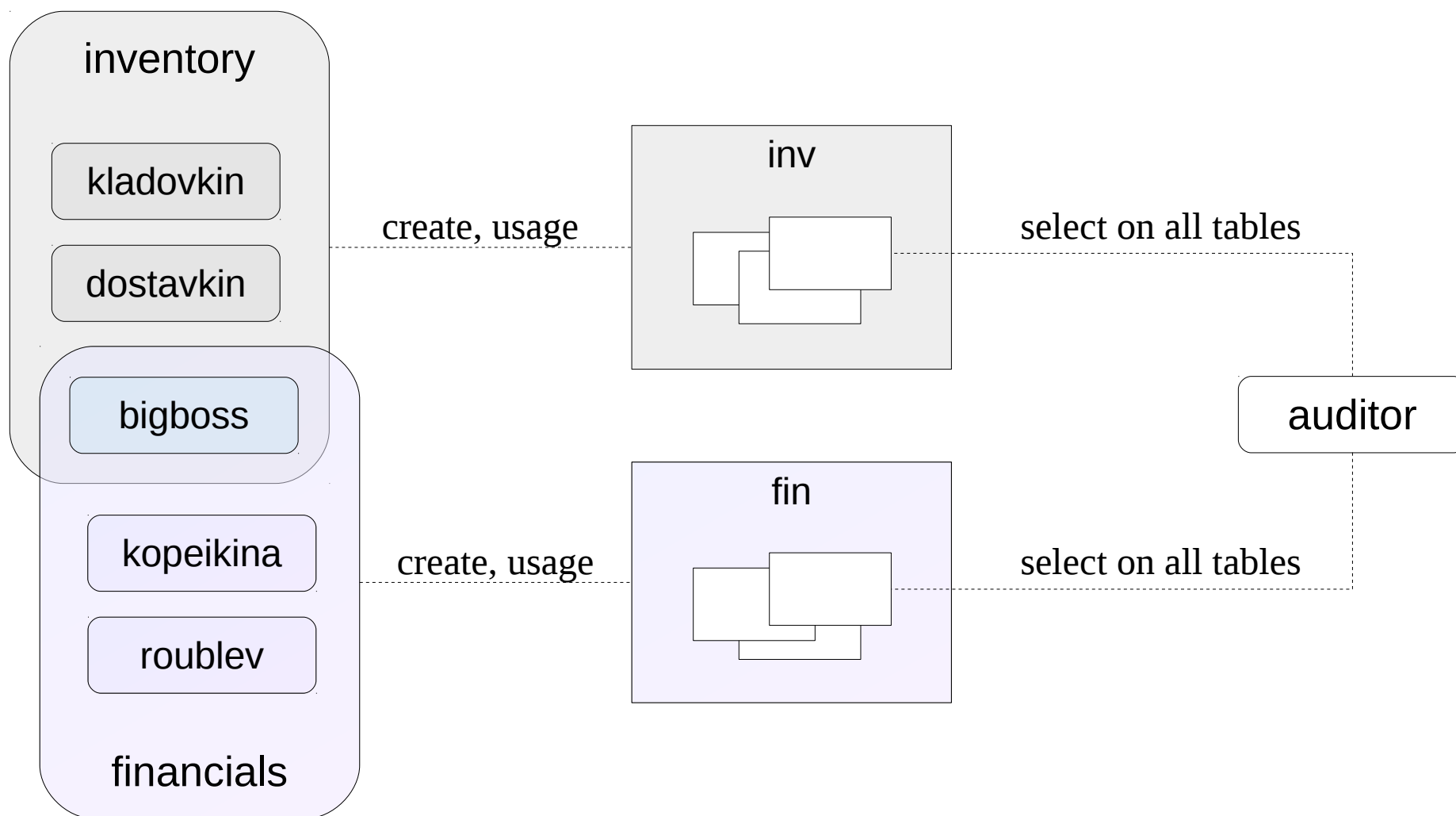
`\ddp`



Пример 1



Пример 2



Рассмотрели привилегии как средство предоставления доступа

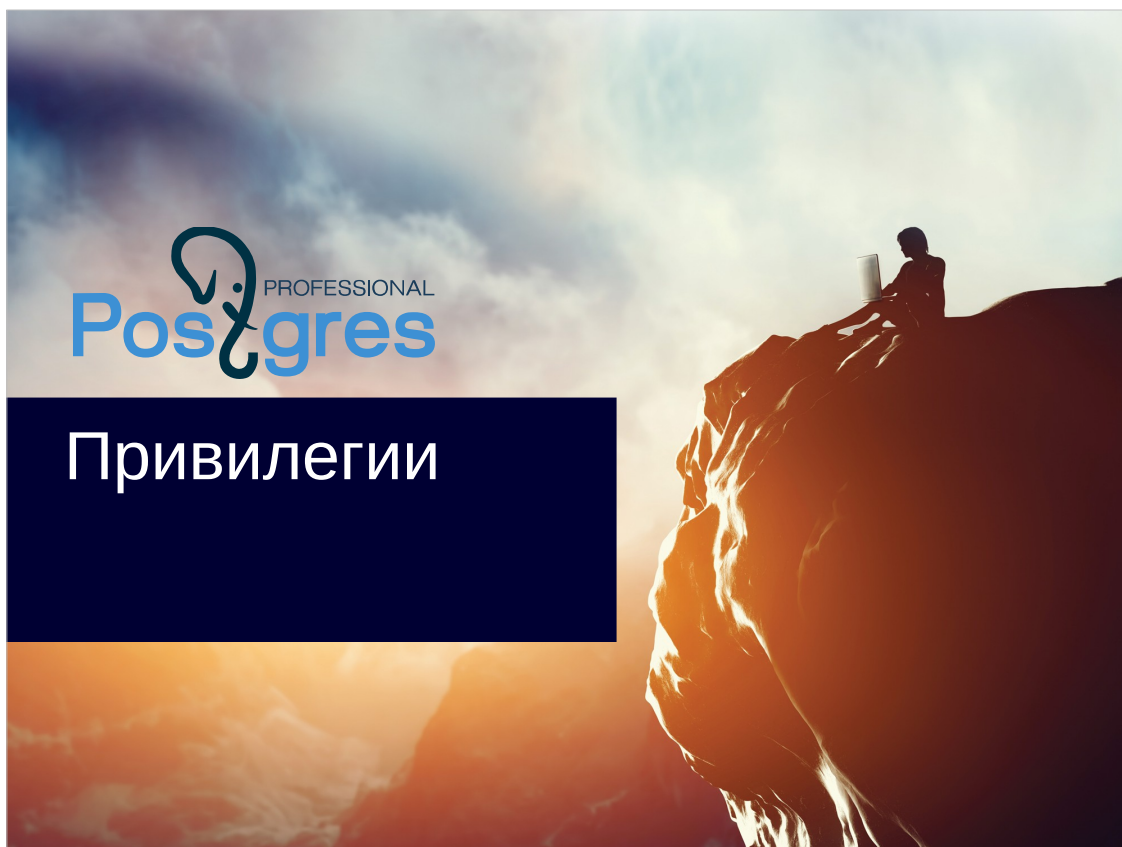
Узнали, как выдавать и отзывать привилегии, в том числе право перевыдачи

Научились просматривать выданные привилегии

Рассмотрели несколько примеров управления доступом

Организовать работу таким образом, чтобы одни пользователи имели полный доступ к таблицам, а другие могли только запрашивать, но не изменять информацию.

1. Создать базу данных DB11 и две роли: writer и reader.
2. Отозвать у роли public все привилегии на схему public, выдать эти привилегии роли writer и только usage — роли reader.
3. Настроить привилегии по умолчанию так, чтобы роль reader получала доступ на чтение к таблицам, принадлежащим writer в схеме public.
4. Создать пользователя w1 в группе writer и r1 в группе reader.
5. Под пользователем writer создать таблицу.
6. Убедиться, что w1 имеет полный доступ к таблице, а r1 — только на чтение.



Авторские права

Курс «Администрирование PostgreSQL 9.4. Базовый курс» разработан в компании Postgres Professional (2015 год).

Авторы: Егор Рогов, Павел Лузанов

Использование материалов курса

Некоммерческое использование материалов курса (презентации, демонстрации) разрешается без ограничений. Коммерческое использование возможно только с письменного разрешения компании Postgres Professional. Запрещается внесение изменений в материалы курса.

Обратная связь

Отзывы, замечания и предложения направляйте по адресу:

edu@postgrespro.ru

Отказ от ответственности

Компания Postgres Professional не несет никакой ответственности за любые повреждения и убытки, включая потерю дохода, нанесенные прямым или косвенным, специальным или случайным использованием материалов курса. Компания Postgres Professional не предоставляет каких-либо гарантий на материалы курса. Материалы курса предоставляются на основе принципа «как есть» и компания Postgres Professional не обязана предоставлять сопровождение, поддержку, обновления, расширения и изменения.

Доступ к объектам БД

Виды привилегий для разных объектов

Выдача и отзыв привилегий и право перевыдачи

Привилегии по умолчанию

Просмотр привилегий

Примеры управления доступом

Владелец объекта

роль, создавшая объект

может быть изменен (`ALTER OBJECT OWNER TO new_role`)

Доступ к объекту имеют

суперпользователь — полный

владелец объекта — полный, но может быть ограничен

другие роли — в рамках выданных привилегий

Право выдачи и отзыва привилегий имеют

суперпользователь

владелец

роль, получившая привилегию с указанием `with grant option`

Изначально полный доступ к объекту имеет владелец этого объекта и суперпользователь (роль, созданная с атрибутом `superuser`).

Владелец — это роль, создавшая объект, хотя в дальнейшем владельца можно переопределить.

Суперпользователь всегда имеет полный доступ. Доступ владельца может быть ограничен.

Кроме того, доступ к объекту можно выдать какой-либо роли с помощью механизма привилегий.

Право выдачи и отзыва привилегий имеют владелец и суперпользователь (это право не может быть ограничено). Если роли были выданы привилегии с правом перевыдачи (`with grant option` — похоже на `with admin option`, рассмотренный в теме «Роли»), то такая роль может выдать аналогичную привилегию другой роли.

<http://www.postgresql.org/docs/current/static/ddl-priv.html>

Виды привилегий

Для таблиц

SELECT	чтение данных
INSERT	вставка строк
UPDATE	изменение строк
DELETE	удаление строк
TRUNCATE	очистка таблицы
REFERENCES	ссылка на таблицу во внешнем ключе
TRIGGER	создание триггеров на таблице

Для представлений

SELECT	чтение данных
TRIGGER	создание триггеров на представлении

Для функций

EXECUTE	выполнение
---------	------------

Список возможных привилегий отличается для объектов различных типов.

Привилегии для основных объектов приведены на этом и следующем слайдах.

<http://www.postgresql.org/docs/current/static/sql-grant.html>

Для последовательностей

SELECT	использование currval
UPDATE	использование nextval и setval
USAGE	использование currval и nextval

Для табличных пространств

CREATE	создание объектов в табличном пространстве
--------	--

Для баз данных

CREATE	создание схем в базе данных
CONNECT	подключение
TEMPORARY	создание временных таблиц

Для схем

CREATE	создание объектов в схеме
USAGE	доступ к объектам в схеме

Продолжение списка привилегий.

Пример выдачи

```
GRANT SELECT,UPDATE ON [TABLE] tbl TO role;  
разрешить чтение и изменение таблицы tbl для роли role  
  
GRANT EXECUTE ON ALL FUNCTIONS IN SCHEMA scm TO role;  
разрешить выполнение всех функций в схеме scm для роли role
```

Пример отзыва

```
REVOKE SELECT,UPDATE ON [TABLE] tbl FROM role;  
запретить чтение и изменение таблицы tbl для роли role  
  
REVOKE EXECUTE ON ALL FUNCTIONS IN SCHEMA scm FROM role;  
запретить выполнение всех функций в схеме scm для роли role
```

Привилегии выдаются командой `grant`, имеющей много различных вариантов для разных типов объектов. Некоторые формы приведены на этом слайде.

<http://www.postgresql.org/docs/current/static/sql-grant.html>

Отзыв привилегий выполняется командой `revoke` и по синтаксису похож на выдачу.

<http://www.postgresql.org/docs/current/static/sql-revoke.html>

На слайде приведены только несколько примеров, больше — в демонстрации дальше в этой теме.

Роль получает привилегии

непосредственно (в том числе как владелец)
через групповые роли, в которые она входит
через псевдороль public, в которую входят все роли

Получение привилегий зависит от атрибута inherit

<code>CREATE ROLE <i>role</i> [INHERIT];</code>	автоматически добавляются привилегии, которыми обладает групповая роль
<code>CREATE ROLE <i>role</i> NOINHERIT;</code>	привилегии группы доступны только при явном переключении: <code>SET ROLE <i>group</i>;</code>

Роль может получать привилегии для доступа к объекту разными способами.

Во-первых, суперпользователь имеет неограниченный доступ ко всем объектам. Но не потому, что у него привилегии, а просто для него не выполняются проверки доступа.

Во-вторых, если роль является владельцем объекта, то она имеет привилегии для полного доступа (если только она сама или суперпользователь не отзовут эти привилегии).

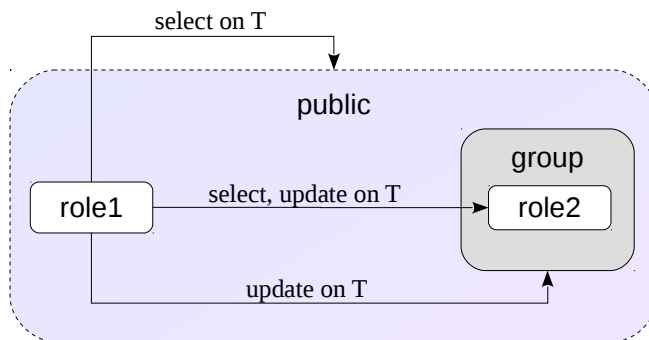
В-третьих, роль обладает привилегиями, выданными ей другими ролями.

В-четвертых, роль с атрибутом inherit (по умолчанию) автоматически обладает как своими привилегиями, так и привилегиями всех групп, в которые она входит. Это касается и «псевдороди» public, в которую неявно входят все роли.

Если же роль создана без атрибута inherit, то она может воспользоваться привилегиями группы, только выполнив команду `set role` и таким образом «переключившись» на эту групповую роль. Все действия, совершаемые ролью, будут совершаться от имени групповой роли (например, групповая роль будет владельцем созданных объектов).

Роль с атрибутом inherit тоже может выполнить `set role`, если это необходимо.

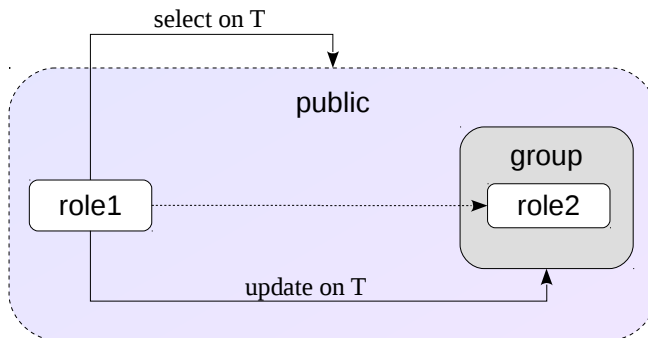
Если `role1` выполнит
`REVOKE SELECT, UPDATE ON t FROM role2`,
какие привилегии останутся у `role2`?



Команды для создания ситуации, показанной на рисунке:

```
\c - postgres
create user role1;
create user role2;
create role "group" role role1, role2;
\c - role1
create table t(n numeric);
grant select on t to public;
grant select, update on t to role2;
grant update on t to "group";
```

Если `role1` выполнит
`REVOKE SELECT, UPDATE ON t FROM role2,`
 у `role2` все равно останутся обе привилегии.



Роль `role2` будет обладать привилегией `select`, так как входит в группу `public`.

Роль `role2` будет обладать привилегией `update`, так как входит в группу `group`.

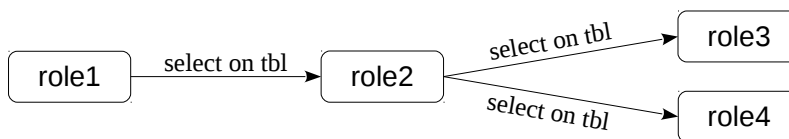
Выдача привилегии с правом перевыдачи

```
role1: GRANT SELECT ON tbl TO role2 WITH GRANT OPTION;
```

теперь и role2 может передавать права

```
role2: GRANT SELECT ON tbl TO role3 WITH GRANT OPTION;
```

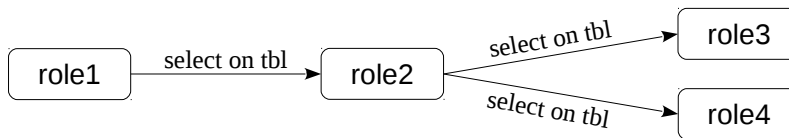
```
role2: GRANT SELECT ON tbl TO role4 WITH GRANT OPTION;
```



Одна и та же привилегия может быть независимо выдана несколькими ролями

При выдаче роли полномочия можно передать ей право дальнейшей перевыдачи (и отзыва) этого полномочия. Если роль воспользуется этим правом, образуется иерархия ролей.

<http://www.postgresql.org/docs/current/static/sql-grant.html>



Отзыв самой привилегии

```
role1: REVOKE SELECT ON tbl FROM role2 CASCADE;
```

Отзыв права перевыдачи

```
role1: REVOKE GRANT OPTION FOR  
SELECT ON tbl FROM role2 CASCADE;
```

Если роль перевыдавала привилегию другим ролям,
указание CASCADE обязательно

11

Отозвать привилегию можно с помощью revoke. Роль может отозвать привилегию только у той роли, которой она его выдала. Например, role1 не может отозвать привилегию непосредственно у role3 или role4.

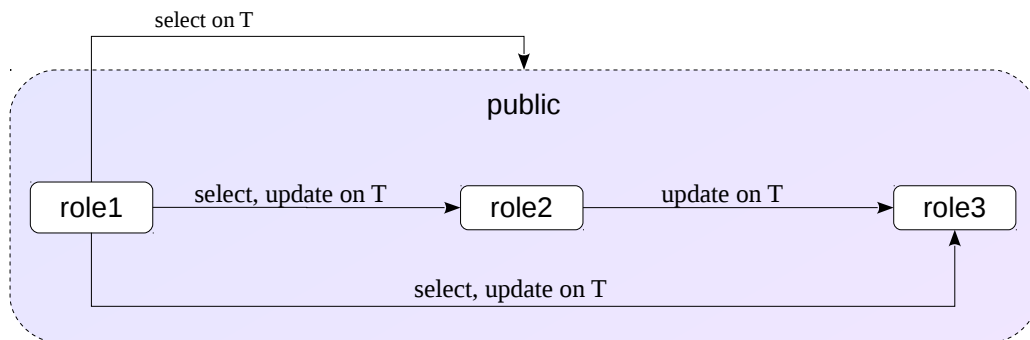
Однако при отзыве привилегии у role1 привилегия будет автоматически отозвана у всех ролей в иерархии. Для этого надо указать ключевое слово cascade.

Если иерархия непуста и cascade не указан, возникнет ошибка.

Право перевыдачи можно отозвать, не отзывая у роли само полномочие. Это выполняется с помощью revoke grant option for. Слово cascade имеет здесь такое же значение, как и при отзыве привилегии.

<http://www.postgresql.org/docs/current/static/sql-revoke.html>

Если `role1` выполнит
`REVOKE SELECT, UPDATE ON t FROM role3,`
какие привилегии останутся у `role3`?



12

Команды для создания ситуации, показанной на рисунке:

`\c - postgres`

`create user role1;`

`create user role2;`

`create user role3;`

`\c - role1`

`create table t(n numeric);`

`grant select on t to public;`

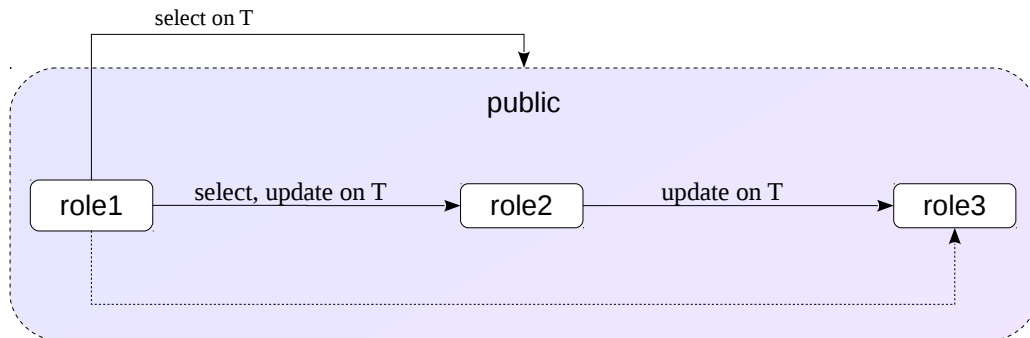
`grant select, update on t to role2 with grant option;`

`grant select, update on t to role3;`

`\c - role2`

`grant update on t to role3;`

Если `role1` выполнит
`REVOKE SELECT, UPDATE ON t FROM role3,`
 у `role3` все равно останутся обе привилегии.



Роль `role3` будет обладать привилегией `select`, так как входит в группу `public`.

Роль `role3` будет обладать привилегией `update`, так как эту привилегию ей выдала не только `role1` (но и `role2`).

По умолчанию роль `public` получает ряд привилегий

для баз данных	CONNECT (подключение) TEMPORARY (создание временных таблиц)
для схемы <code>public</code>	CREATE (создание объектов) USAGE (доступ к объектам)
для схем <code>pg_catalog</code> и <code>information_schema</code>	USAGE (доступ к объектам)
для функций	EXECUTE (выполнение)

Дополнительные привилегии по умолчанию

```
ALTER DEFAULT PRIVILEGES  
\ddp
```

14

По умолчанию псевдороль `public` получает ряд привилегий (то есть, фактически, их получают все роли):

- подключение и создание временных таблиц для **всех** баз данных
- использование схемы **public** и создание в ней объектов
- использование схем **pg_catalog** и **information_schema**
- выполнение **всех** функций

Это сложно «заметить» и такое поведение может оказаться нежелательным. В таком случае надо явно отзывать привилегии.

<http://www.postgresql.org/docs/current/static/sql-grant.html>

<http://www.postgresql.org/docs/current/static/ddl-schemas.html>

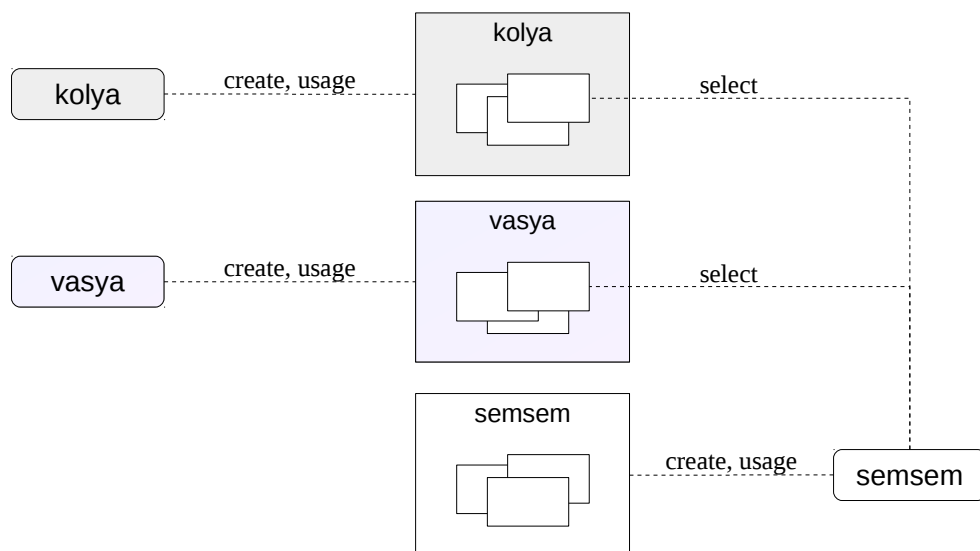
Можно настроить дополнительные привилегии, выставляемые по умолчанию, с помощью `alter default privileges`. (Но избавиться таким образом от привилегий `public` не получится — изначально «привилегии по умолчанию» пусты.)

<http://www.postgresql.org/docs/current/static/sql-alterdefaultprivileges.html>



В демонстрации будут показаны различные вариации команд `grant` и `revoke`, рассмотрены примеры работы с привилегиями и показано, как посмотреть выданные привилегии для объекта.

Пример 1



Механизмы управления доступом (роли, схемы, привилегии) весьма гибки и позволяют в разных случаях организовать работу удобным образом. Приведем два примера.

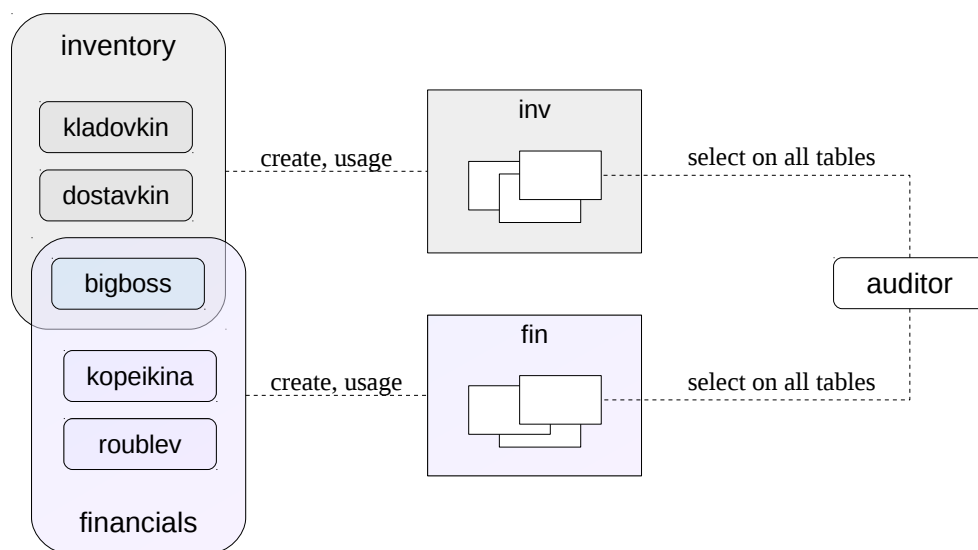
Пример простого использования.

Семен Семенович и его студенты Коля и Вася занимаются научными исследованиями и хранят результаты измерений в базе данных.

Системный администратор создал на факультетском сервере БД каждому из них своего пользователя и одноименную схему. Путь поиска он оставил по умолчанию. Групповые роли не используются.

Каждый пользователь является владельцем объектов, которые он создает в своей схеме. Кроме того, Коля и Вася дают доступ к некоторым своим таблицам Семену Семеновичу, чтобы он мог посмотреть их результаты.

Пример 2



17

Более сложный пример.

На выделенный сервер БД установлена система автоматизации предприятия. Она состоит из двух модулей — складского и финансового.

Для каждого из модулей создана своя схема (`inv` для склада и `fin` для финансов) и своя групповая роль (`inventory` для склада и `financials` для финансов). Групповая роль является владельцем всех объектов в своей схеме.

В складском отделе работают Кладовкин и Доставкин. Для каждого создан свой пользователь и включен в группу `inventory`.

В финансовом отделе работают Копейкина и Рублев. Для каждого создан свой пользователь и включен в группу `financials`.

Для генерального директора предприятия на всякий случай создан пользователь, включенный в обе группы, хотя вряд ли он когда-либо воспользуется системой.

Для проведения аудита создан специальная роль `auditor`, которой выдан доступ на чтение ко всем объектам двух схем.

Рассмотрели привилегии как средство предоставления доступа

Узнали, как выдавать и отзывать привилегии, в том числе право перевыдачи

Научились просматривать выданные привилегии

Рассмотрели несколько примеров управления доступом

Организовать работу таким образом, чтобы одни пользователи имели полный доступ к таблицам, а другие могли только запрашивать, но не изменять информацию.

1. Создать базу данных DB11 и две роли: writer и reader.
2. Отозвать у роли public все привилегии на схему public, выдать эти привилегии роли writer и только usage — роли reader.
3. Настроить привилегии по умолчанию так, чтобы роль reader получала доступ на чтение к таблицам, принадлежащим writer в схеме public.
4. Создать пользователя w1 в группе writer и r1 в группе reader.
5. Под пользователем writer создать таблицу.
6. Убедиться, что w1 имеет полный доступ к таблице, а r1 — только на чтение.

19

Решение

```
# create database db11;
# create user writer;
# create user reader;

# \c db11
# revoke all on schema public from public;
# grant all on schema public to writer;
# grant usage on schema public to reader;

# alter default privileges for role writer in schema public
grant select on tables to reader;

# create user w1 in role writer;
# create user r1 in role reader;

# \c - writer
# create table t(n numeric);

# \c - w1
# insert into t values (42);
-- ok
# \c - r1
# select * from t;
-- ok
# update t set n=n+1;
-- ошибка
```