

БЛОК 4. ОРГАНИЗАЦИЯ ДАННЫХ

ЛОГИЧЕСКОЕ УСТРОЙСТВО. БАЗЫ ДАННЫХ И СХЕМЫ

{ }

begin



ЦЕЛЬ УРОКА



1

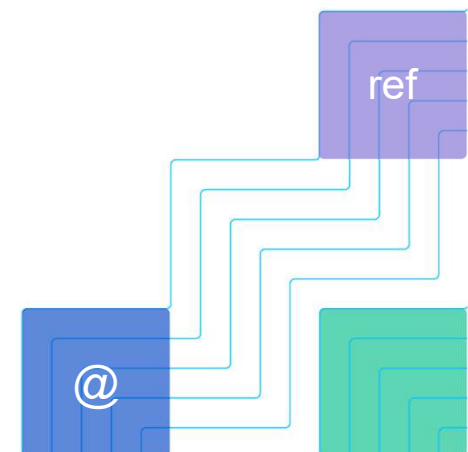
Понять логическое устройство Постгреса

2

Понять устройство базы, схемы, таблицы

3

Узнать какие еще существуют отношения



СОДЕРЖАНИЕ УРОКА

1

Логическое устройство PostgreSQL

2

Базы данных и схемы

3

Таблицы и другие отношения

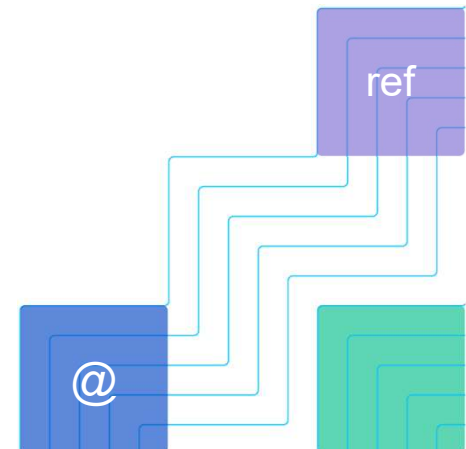
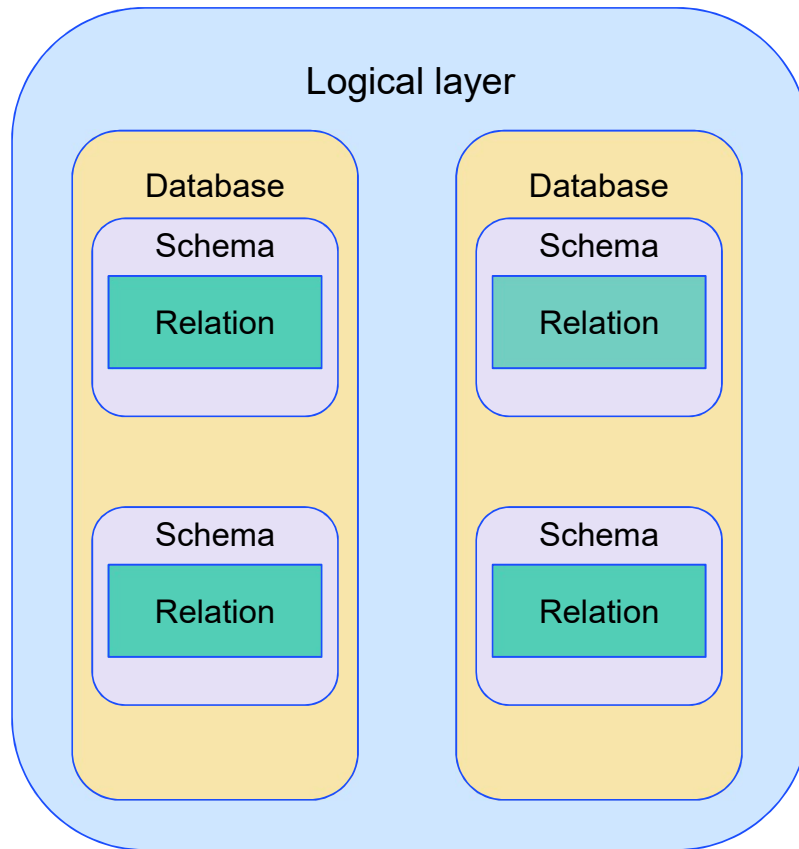


STEP
UP

ref

@

ЛОГИЧЕСКОЕ УСТРОЙСТВО POSTGRESQL





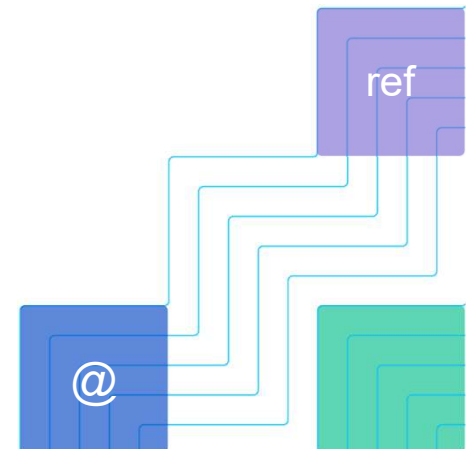
DATABASE



Является контейнером самого верхнего уровня
По умолчанию в любом кластере есть как минимум 3 БД:

- 1 postgres
- 2 template0
- 3 template1

Присутствует на логическом и физическом уровне



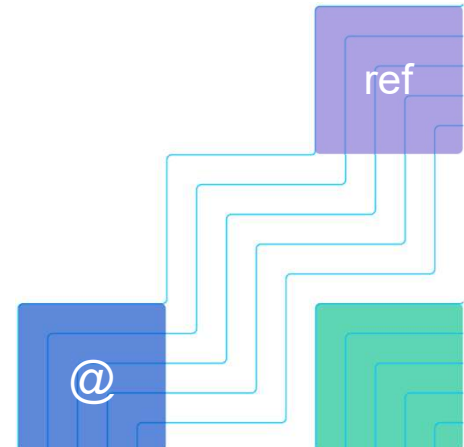


TEMPLATE0

- для восстановления из резервной копии
- по умолчанию даже нет прав на connect
- лучше всего не создавать в ней никаких объектов
- а еще лучше про нее забыть и не вспоминать ;)

TEMPLATE1

- используется как шаблон для создания новых баз данных
- в нем имеет смысл делать некие действия, которые не хочется делать каждый раз при создании новых баз данных
- например create extension или create schema
- но (как мне кажется) лучше не создавать объектов, так как для других пользователей это будет неочевидно

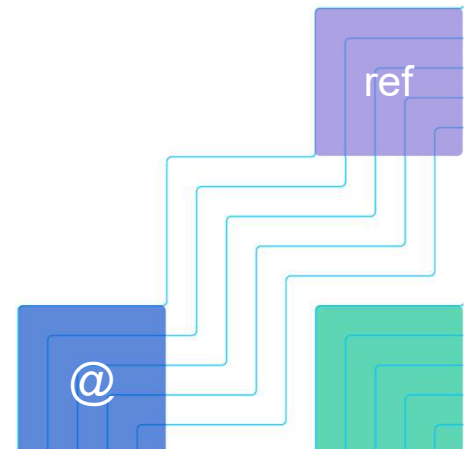




POSTGRES



- первая база данных для регулярной работы
- создается по умолчанию
- хорошая практика - также не использовать, но и не удалять - иногда нужна для различных утилит





ШАБЛОНЫ БД

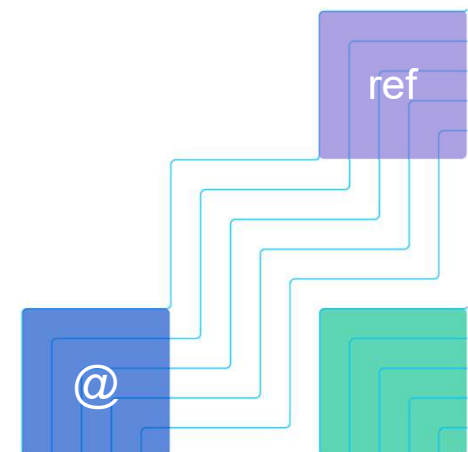


В качестве шаблона для создания БД можно использовать любую другую БД - при этом возникает соблазн использовать `create database with template` для копирования БД. Так делать **не рекомендовано**. Вот выдержка из документации



CREATE DATABASE will fail if any other connection exists when it starts; otherwise, new connections to the template database are locked out until CREATE DATABASE completes

Используйте штатные методы бэкапов или репликации. Обсудим на соответствующих темах




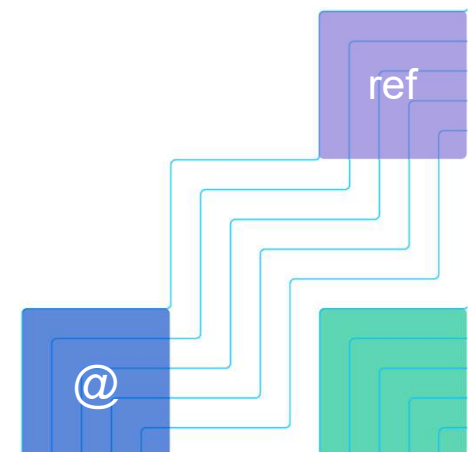
CREATE DATABASE



```
CREATE DATABASE имя
  [ [ WITH ] [ OWNER [=] имя_пользователя ]
    [ TEMPLATE [=] шаблон ]
    [ ENCODING [=] кодировка ]
    [ LOCALE [=] локаль[@провайдер] ]
    [ LC_COLLATE [=] категория_сортировки[@провайдер] ]
    [ LC_STYPE [=] категория_типов_символов ]
    [ TABLESPACE [=] табл_пространство ]
    [ ALLOW_CONNECTIONS [=] разр_подключения ]
    [ CONNECTION LIMIT [=] предел_подключений ]
    [ IS_TEMPLATE [=] это_шаблон ] ]
```

 <https://postgrespro.ru/docs/postgresql/14/sql-createdatabase>

 Список диалектов и правил сортировки
<https://postgrespro.ru/docs/postgresql/14/locale>



CREATE SCHEMA

```
CREATE SCHEMA имя_схемы [ AUTHORIZATION указание_роли ] [ элемент_схемы [ ... ] ]  
CREATE SCHEMA AUTHORIZATION указание_роли [ элемент_схемы [ ... ] ]  
CREATE SCHEMA IF NOT EXISTS имя_схемы [ AUTHORIZATION указание_роли ]  
CREATE SCHEMA IF NOT EXISTS AUTHORIZATION указание_роли
```

Здесь *указание_роли*:

имя_пользователя
| CURRENT_USER
| SESSION_USER



<https://postgrespro.ru/docs/postgresql/14/sql-createschema>



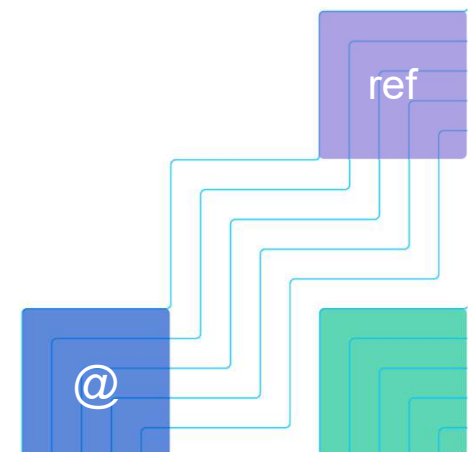
CREATE TABLE



```
CREATE [ { TEMPORARY | TEMP } | UNLOGGED ] TABLE [ IF NOT  
EXISTS ] имя_таблицы ( [  
    { имя_столбца тип_данных [ ограничение_столбца [ ... ] ]
```



<https://postgrespro.ru/docs/postgresql/14/sql-createtable>



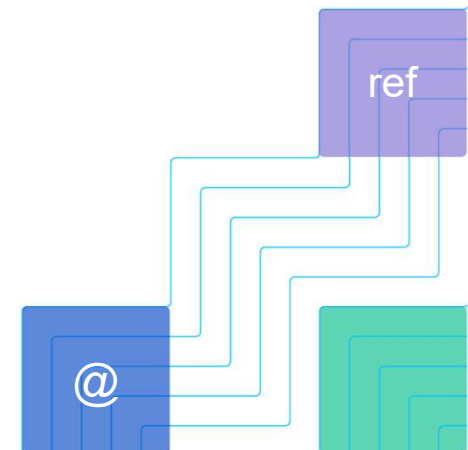


ВИДЫ ОТНОШЕНИЙ В БД



r = ordinary table,
i = index,
S = sequence,
v = view,
m = materialized view,
c = composite type,
t = TOAST table,
f = foreign table, внешняя таблица

<https://www.postgresql.org/docs/14/catalog-pg-class.html>





ИТОГИ ЗАНЯТИЯ



01



Поняли логическое
устройство Постгреса

02

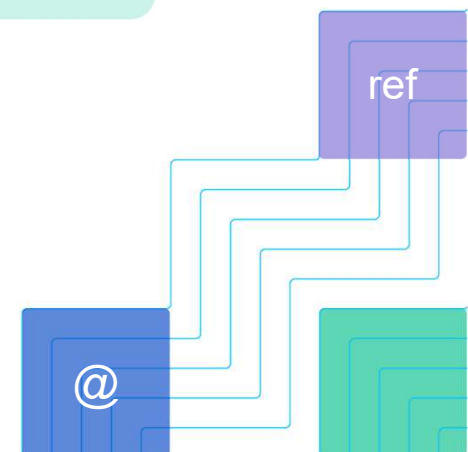


Поняли устройство
базы, схемы,
таблицы

03



Узнали какие еще
существуют
отношения



ПРОВЕРОЧНОЕ ЗАДАНИЕ

Цель задания: создать свои базы данных,
схемы и таблицы

{ }

begin



ЗАДАНИЕ НА САМОПРОВЕРКУ



01

Создать свою БД

02

Переключиться на
созданную БД

03

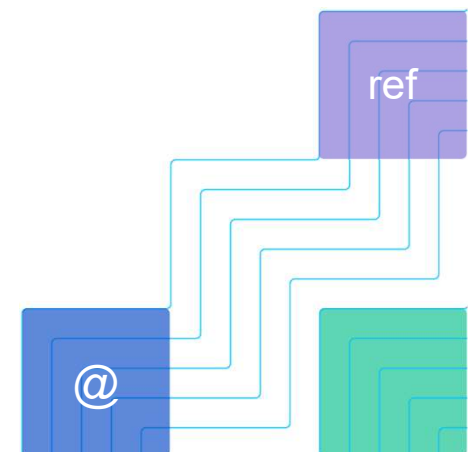
Создать схему

04

Создать таблицу в
созданной на 3 шаге
схеме

05

Убедиться, что таблица
создалась в нужной
схеме



СПАСИБО

На следующем занятии мы рассмотрим тему:

- Табличные пространства

{ }

End