Лекция 4

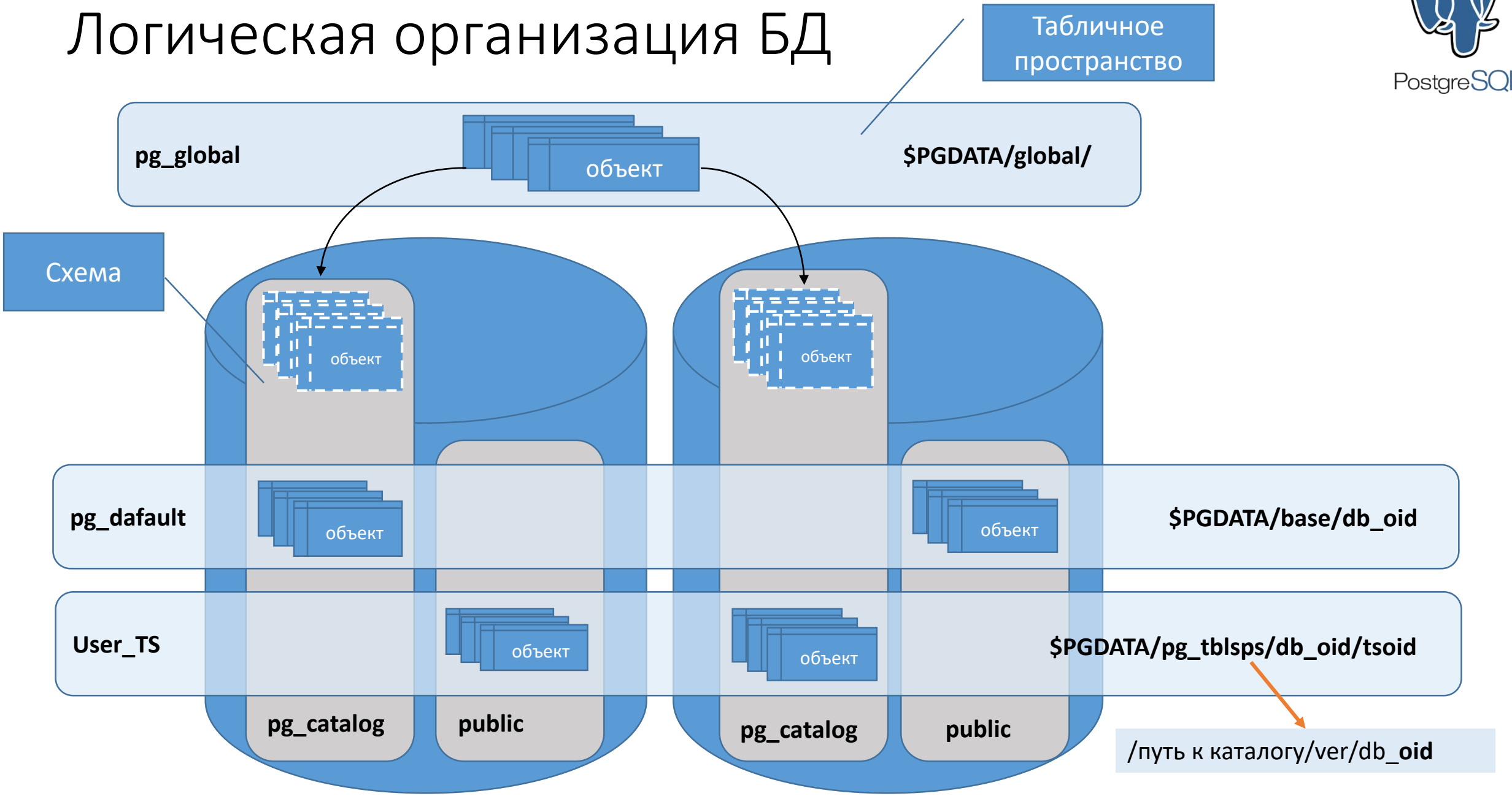
Создание БД

Кластер баз данных

- При инициализации кластера БД создается три базы данных:
  - ***postgres*** – используется при подключении по умолчанию пользователем postgres. Не является обязательной, но некоторые утилиты предполагают ее наличие
  - ***template0*** – базовый шаблон
    Не должен изменяться, нужен как минимум в двух ситуациях:
    1. для восстановления БД из резервной копии (**pg_dump**)
    2. при создании новой БД с кодировкой, отличной от указанной при инициализации кластера
  - ***template1*** – шаблон для создания новых БД
    - используется по умолчанию в команде **CREATE DATABASE**
    - может быть удален и создан заново на основе ***template0***
      (необходимо сбросить флаг **pg_database.datistemplate = false**)

```
select *
from pg_catalog.pg_database pd ;
```

Логическая организация БД



Создание БД

```
CREATE DATABASE имя
  [ [ WITH ] [ OWNER [=] имя_пользователя ]
       [ TEMPLATE [=] шаблон | DEFAULT]
       [ ENCODING [=] кодировка ]
       [ LC_COLLATE [=] категория_сортировки ]
       [ LC_CTYPE [=] категория_типов_символов ]
       [ TABLESPACE [=] табл_пространство | DEFAULT ]
       [ ALLOW_CONNECTIONS [=] true | false]
       [ CONNECTION LIMIT [=] -1 | предел_подключений ]
       [ IS_TEMPLATE [=] true | false] ]
```

- Необходимо быть суперпользователем или иметь специальное право CREATEDB
- Чтобы шаблон можно было использовать для создания базы, к нему не должно быть активных подключений
- Конфигурационные параметры уровня БД и разрешения уровня БД из шаблона не копируются
- CREATE DATABASE нельзя выполнять внутри блока транзакции

Примеры создания БД

```
CREATE DATABASE TestDB;

CREATE DATABASE sales
WITH
     OWNER salesapp
     TABLESPACE salesspace; -- табличное пространство д.б. создано заранее!

CREATE DATABASE hr
WITH
   ENCODING = 'UTF8'
   OWNER = hr
   CONNECTION LIMIT = 100   TEMPLATE template0;
--Параметры локали и кодировка должны соответствовать тем, что установлены в
--шаблоне, если только это не template0
```

Параметры БД

- **IS_TEMPLATE** (**datistemplate**)
  - **true** - пользователи с правом CREATEDB могут клонировать БД;
  - **false** - **только** суперпользователь и владелец базы данных могут её клонировать.
- **ALLOW_CONNECTIONS** (**datallowconn**)
  - **true** – пользовательские подключения к БД разрешены;
  - **false** - новые подключения к этой базе не допустимы.
    БД template0 помечена как **datallowconn = false** для избежания любых модификаций
- **CONNECTION LIMIT** (**datconnlimit**)
  - -1 (по умолчанию) - снимает ограничение.
  - не распространяется на суперпользователей и фоновые рабочие процессы

```
SELECT datname, datistemplate, datallowconn, datconnlimit
FROM pg_database;
```

Ошибки

- Нет разрешения на выполнение CREATE DATABASE
- Недостаточно прав в каталоге данных
- Недостаточно места на диске или другие проблемы в файловой системе
- Есть активные подключения к шаблону

Изменение параметров БД

```
ALTER DATABASE имя [ [ WITH ] параметр [ ... ] ]
ALTER DATABASE имя RENAME TO новое_имя
ALTER DATABASE имя OWNER TO {новый_владелец|CURRENT_USER|SESSION_USER}
ALTER DATABASE имя SET TABLESPACE новое_табл_пространство
```

- Изменение параметров на уровне базы данных:
  ALLOW_CONNECTIONS, CONNECTION LIMIT и ***IS_TEMPLATE***
- Изменение имени БД.
  Текущую БД переименовывать нельзя (подключитесь к другой БД)
- Изменение владельца БД
- Изменение табличного пространства **по умолчанию** для БД.
  Новое табличное пространство не должно содержать объекты этой БД

```
ALTER DATABASE db RENAME TO appdb;
ALTER DATABASE appdb CONNECTION LIMIT 10;
```

Удаление базы данных

```
DROP DATABASE имя;
```

- При удалении БД также удаляются все её объекты
  - **Удаление базы данных это необратимая операция!**
- Невозможно выполнить команду DROP DATABASE пока существует хоть одно подключение к заданной базе
  - Нужно подключиться к любой другой (в том числе и template1)
- Нужно быть владельцем БД или суперпользователем

Схемы

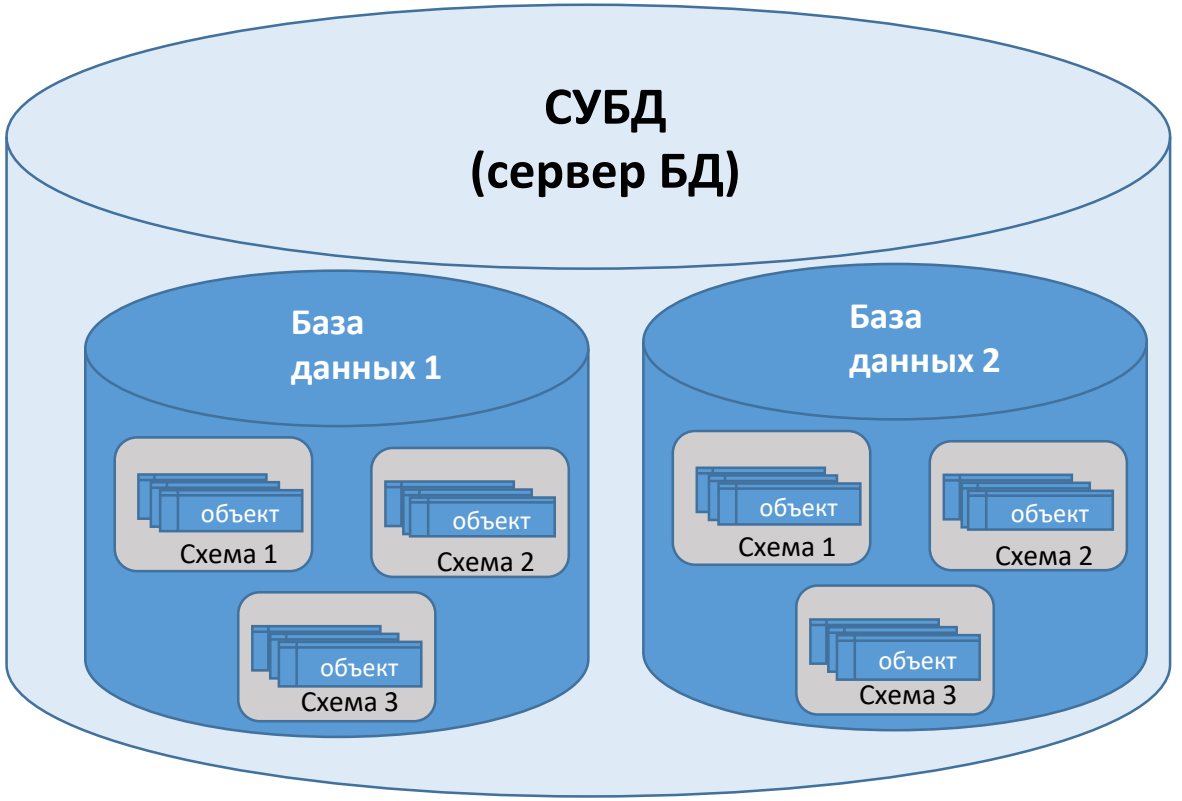
Именование объектов БД

Имя_сервера.Имя_БД.Имя_схемы.Имя_объекта

Пример: Mia-SQL.dbSQL.dbo.”Client”

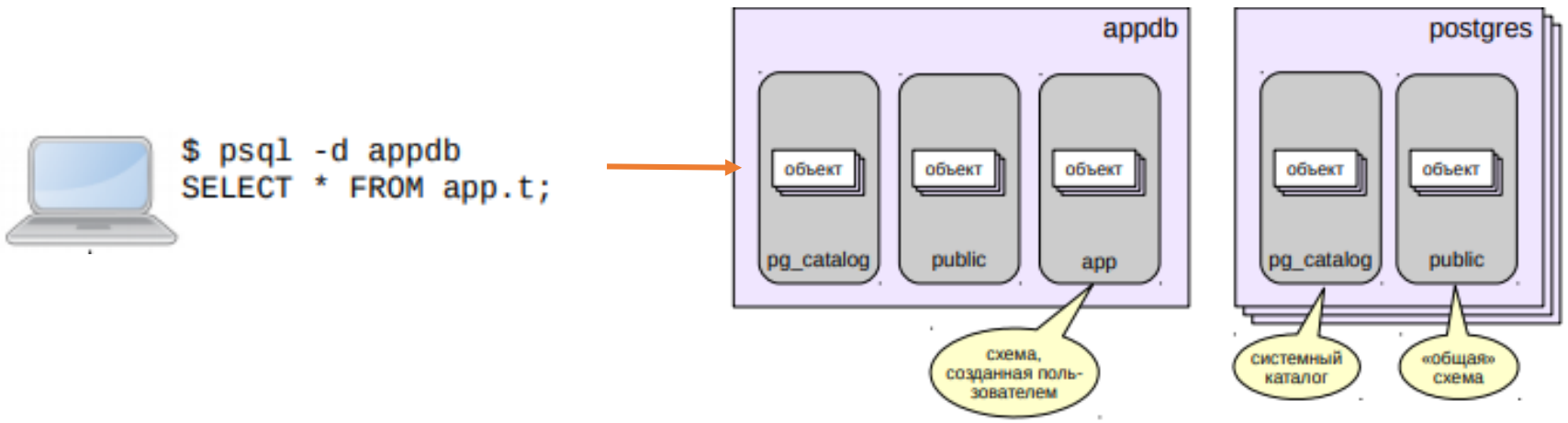
```
SELECT *
FROM dbo."NewTable";
```

- Правила именования объектов
  - Должно начинаться с буквы
  - Может быть длиной до 128 символов*
  - Должно содержать только символы A–Z, a–z, 0–9, _, $ и #
  - Не должно совпадать с именем другого объекта БД
  - Не должно совпадать с зарезервированным словом языка
  - Ограничителем имени является **двойная кавычка** (имена нарушающие правила или чувствительные к регистру)



Базы данных и схемы

- Кластер баз данных (экземпляр) состоит из баз данных
- База данных содержит различные схемы
- Схемы содержат объекты
  - В каждой БД есть набор стандартных схем: **pg_catalog, public**
  - Пользователь может создавать пользовательские схемы
- Клиент в БД может работать с объектами в любых схемах



Разрешение имен

- Указание схемы объекта
  - квалифицированное имя объекта явно определяет схему - **схема.имя**
  - имя без квалификатора проверяется в схемах, указанных в **пути поиска**

```
select name, setting
from pg_catalog.pg_settings ps
where name = 'search_path';
```

Специальные схемы

- Схема **pg_*catalog* –** схема для объектов системного каталога
- Схема ***public*** – схема для размещения объектов по умолчанию
- Схема **pg_*temp***
  - автоматически создается для временных таблиц **в каждом сеансе**
  - по окончании сеанса все объекты временной схемы удаляются

```
select *
from pg_catalog.pg_namespace pn ;
```

Управление схемами

- Создание схемы

```
CREATE SCHEMA имя_схемы ;
```

- Удаление пустой схемы (не содержащую объектов)

```
DROP SCHEMA имя_схемы;
```

- Удаление схемы со всеми содержащимися в ней объектами

```
DROP SCHEMA имя_схемы CASCADE;
```

Таблицы

Создание таблиц

```
CREATE [TEMP][UNLOGGED] TABLE [IF NOT EXISTS]
schema_name.table_name
(column_name datatype [DEFAULT expr][, ...])
```

```
CREATE TABLE "Sales"."CustomerOrders"
                    (OrderID     serial         NOT NULL,
                    OrderDate    date           NOT NULL,
                    CustomerID   int            NOT NULL,
                    Notes        varchar(200)   NULL);
```

Типы данных

- Символьные:
  - varchar(n) , char(n) , text
- Числовые:
  - smallint (int2), integer (int4), bigint (int8), money, numeric(scale, precision) и decimal(scale, precision)
  - real, double precision и float(p)
  - serial (int4), bigserial(int8) и smallserial(int2)
- Дата и время:
  - date, time и time with time zone
  - timestamp, timestamp with time zone (timestamptz)
- Бинарные:
  - bytea
- Логические и битовые:
  - Boolean и bit

Специальные типы данных

- Геометрические типы
- Типы, описывающие сетевые адреса
- XML
- Типы JSON
- Массивы
- Составные типы
- Диапазонные типы
- Пользовательские и доменные типы

https://postgrespro.ru/docs/postgresql/14/datatype

Генерируемые столбцы

- Столбцы, вычисляемые на основе других столбцов
  - сохранённые - вычисляются при записи (добавлении или изменении)
  - виртуальные - вычисляются при чтении (* не реализовано)
- Определяются при помощи предложения **GENERATED ALWAYS AS**

```
CREATE TABLE people (
    ...,
    height_cm numeric,
    height_in numeric GENERATED ALWAYS AS (height_cm / 2.54) STORED
);
```

https://postgrespro.ru/docs/postgresql/14/ddl-generated-columns

Создание таблицы на основе выборки

```
CREATE TABLE Table_name
 [ ( Column_name  [, ...] ) ]
AS
SELECT
```

- Количество столбцов, определенных в необязательном списке в круглых скобках, должно соответствовать количеству столбцов выборки
  - если вы указываете необязательный список столбцов в круглых скобках, вы не можете использовать звездочку (*) в операторе select
- Типы данных столбцов должны быть совместимыми
- Если необязательный список столбцов в круглых скобках содержит количество столбцов отличное от возвращаемого оператором select – возвращается ошибка

```
ERROR: CREATE TABLE/AS SELECT has mismatched column count
```

Создание копии таблицы

```
CREATE TABLE new_table_name
(LIKE source_table_name  { INCLUDING | EXCLUDING }
                         { COMMENTS | COMPRESSION | CONSTRAINTS
                          | DEFAULTS | GENERATED | IDENTITY | INDEXES
                          | STATISTICS | STORAGE | ALL });
```

```
CREATE TABLE student_1
(LIKE student INCLUDING INDEXES INCLUDING COMMENTS);
```

- Создает новую таблицу со структурой родительской таблицы
- CREATE TABLE LIKE не копирует какие-либо данные!

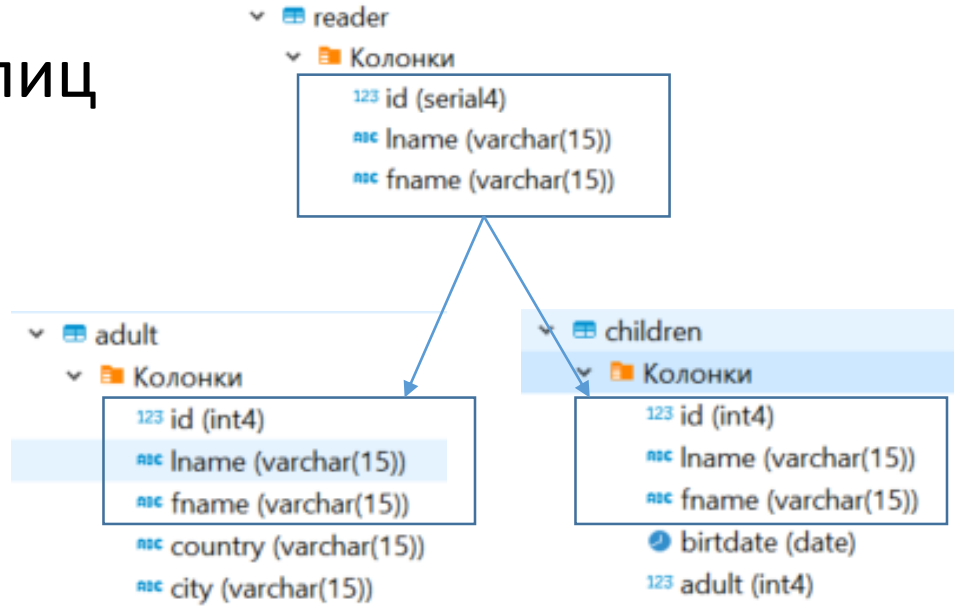
Наследование

- PostgreSQL реализует наследование таблиц

```
CREATE TABLE reader (id serial PRIMARY KEY,
Lname varchar(15),
Fname varchar(15));

CREATE TABLE adult(country varchar(15), city
varchar(15))
INHERITS(READER);

CREATE TABLE children(birtdate date, adult int)
INHERITS(READER);
```




```
SELECT * FROM ONLY reader r ;

SELECT r.tableoid, c.relname,*
FROM reader r
JOIN pg_class AS c ON r.tableoid = c.oid;
```

	tableoid	relname	id	lname	fname
1	58 412	adult	1	Petrov	Ivan
2	58 412	adult	2	Vasin	Ilia
3	58 416	children	3	Petrova	Maria

Наследование

- Таблица может наследоваться от нескольких родительских таблиц:
  - она будет объединять в себе все столбцы этих таблиц, а также столбцы, описанные непосредственно в её определении
  - если в определениях родительских и дочерней таблиц встретятся столбцы с одним именем и одним типом данных - эти столбцы будут «объединены»
- Дочерние таблицы:
  - автоматически наследуют от родительской таблицы ограничения-проверки и ограничения NOT NULL (если только для них не задано явно NO INHERIT).
  - Все остальные ограничения (уникальности, первичный ключ и внешние ключи) не наследуются
- Родительскую таблицу нельзя удалить, пока существуют унаследованные от неё
  - Для удаления таблицы вместе со всеми её потомками необходимо использовать параметр CASCADE

Создание временных таблиц

- Существуют в рамках одной сессии
- Размещаются в схеме **pg_temp**
- Не поддерживают ограничение FOREIGN KEY

```
CREATE TEMPORARY TABLE MyTempTable (cola INT PRIMARY KEY);
GO
INSERT INTO MyTempTable VALUES (1);
```

Изменение и удаление таблиц

```
ALTER TABLE table_name
{ ALTER COLUMN …
| ADD COLUMN …
| DROP COLUMN … [CASCADE]}
```

```
ALTER TABLE "Production"."Products"
RENAME COLUMN productid TO product_number;

ALTER TABLE "Production"."Products"
RENAME TO "Production"."Product";
```

```
ALTER TABLE "Production"."Products"
ADD COLUMN description text;
```

```
ALTER TABLE "Production"."Products"
DROP COLUMN description ;
```

```
ALTER TABLE "Production"."Products"
ALTER COLUMN  productname TYPE varchar(50);
```

```
ALTER TABLE "Production"."Products"
ALTER COLUMN unitprice SET DEFAULT 7.77;
```

```
DROP TABLE [IF EXISTS] table_name [CASCADE]
```

https://postgrespro.ru/docs/postgresql/14/ddl-alter