



Физическая модель



Метаморфоза

Переход из одной формы в другую с приобретением нового внешнего вида и функций.

Переход из логической структуры к физической. Сущность становится таблицей, атрибуты — столбцами, уникальный идентификаторы — ключами, связи — ограничением целостности (констрейтами). Для того чтобы переход состоялся нужно осуществить описание объектов на языке SQL и выполнить скрипт создания объектов в СУБД.

Stop following me, you fucking freaks!



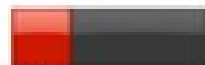
Key-Value



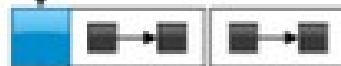
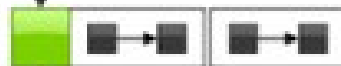
Key Value



Ordered Key-Value



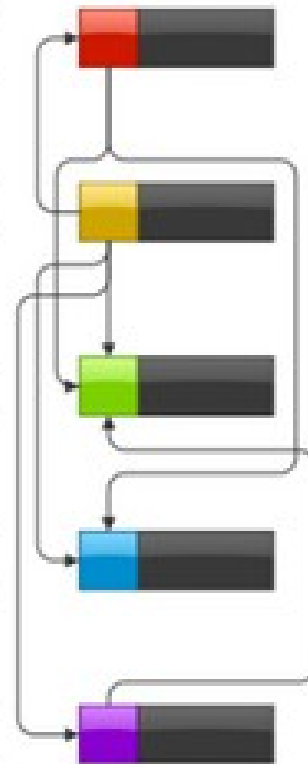
Big Table



Document,
Full-Text Search



Graph



SQL

```
graph TD; SQL[SQL] --> DDL[DDL]; SQL --> DML[DML];
```

SQL

DDL

язык определения данных
(Data Definition Language)

DML

язык манипулирования данными
(Data Manipulation Language)

DDL- язык для создания объектов базы данных

Комманды:

CREATE DATABASE	— создать базу данных
CREATE USER	— создать пользователя
CREATE TABLE	— создать таблицу
ALTER TABLE	— модифицировать таблицу
RENAME TO	— переименовать таблицу
CHANGE COLUMN	— изменить имя и тип данных столбца
MODIFY COLUMN	— изменить тип данных или позицию столбца.
ADD COLUMN	— добавить столбец в таблицу
DROP COLUMN	— удалить столбец из таблицы
DROP TABLE	— удалить таблицу

<https://www.postgresql.org/docs/current/sql-commands.html>

DML - язык манипулирования данными

Комманды:

SELECT	— извлечение данных
UPDATE	— модификация данных
DELETE	— удаление данных
INSERT INTO	— вставка новых данных в таблицу

Объекты базы данных

- Таблицы
- Ключи
- Индексы
- Констрейнты (связи)
- Представления
- Процедуры
- Триггера
- Функции

Создание пользователя

```
CREATE USER test  
WITH PASSWORD 'jw8s0F4';
```

<https://www.postgresql.org/docs/current/sql-createuser.html>

Создание DB

Синтаксис

```
CREATE DATABASE name
  [ [ WITH ] [ OWNER [=] user_name ]
    [ TEMPLATE [=] template ]
    [ ENCODING [=] encoding ]
    [ LOCALE [=] locale ]
    [ LC_COLLATE [=] lc_collate ]
    [ LC_CTYPE [=] lc_ctype ]
    [ TABLESPACE [=] tablespace_name ]
    [ ALLOW_CONNECTIONS [=] allowconn ]
    [ CONNECTION LIMIT [=] connlimit ]
    [ IS_TEMPLATE [=] istemplate ] ]
```

<https://www.postgresql.org/docs/current/sql-createdatabase.html>

CREATE DATABASE

```
CREATE DATABASE music  
WITH OWNER 'test'  
LOCALE 'ru_RU.utf8'  
TEMPLATE template0;
```

Типы данных

SERIAL — автоинкремент
INTEGER — целое число
NUMERIC — число с плавающей точкой
VARCHAR() — текстовые данные длиной до 255
TEXT — набор с максимальной длиной 65535
DATE — дата.
TIMESTAMP — дата и время.
BOOLEAN — логический тип
JSON — текстовый json
BLOB — массив двоичных данных.

Создание таблицы

Синтаксис

```
CREATE [ [ GLOBAL | LOCAL ] { TEMPORARY | TEMP } | UNLOGGED ] TABLE [ IF NOT EXISTS ] table_name
    [ ( column_name [, ...] ) ]
    [ USING method ]
    [ WITH ( storage_parameter [= value] [, ...] ) | WITHOUT OIDS ]
    [ ON COMMIT { PRESERVE ROWS | DELETE ROWS | DROP } ]
    [ TABLESPACE tablespace_name ]
AS query
[ WITH [ NO ] DATA ]
```

Примеры

<https://www.postgresql.org/docs/current/sql-createtableas.html>

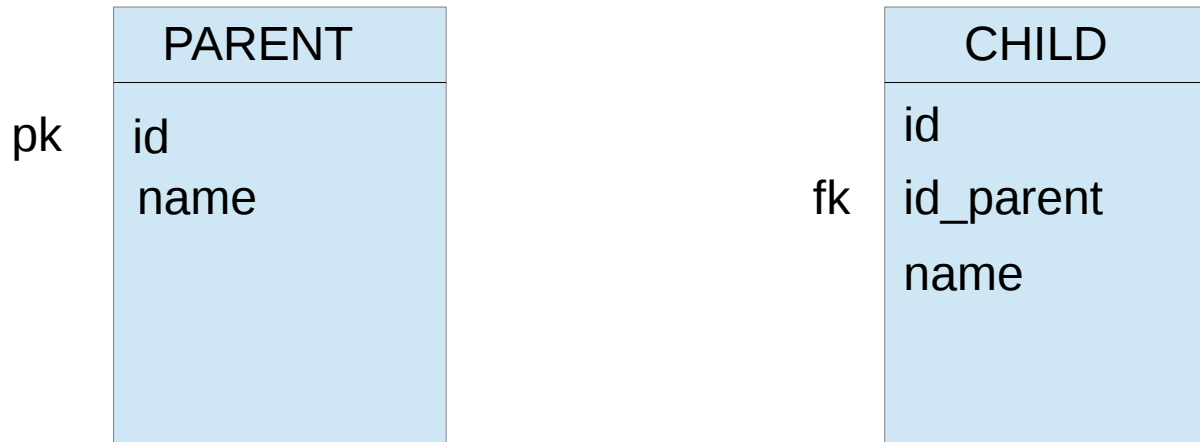
Как создать таблицу?

```
CREATE TABLE "project" (  
  "id_project" SERIAL PRIMARY KEY,  
  "title" VARCHAR(250) NOT NULL,  
  "dt_start" DATE,  
  "dt_end" DATE,  
  "desc_full" TEXT NOT NULL,  
  "desc_short" TEXT NOT NULL,  
  "status" INTEGER,  
  "href_avatar" VARCHAR(100) NOT NULL,  
  "tag" JSONB NOT NULL,  
  "id_parent_project" INTEGER NOT NULL,  
  "name_rev" TEXT NOT NULL  
);
```

Как добавить связь между таблицами ?

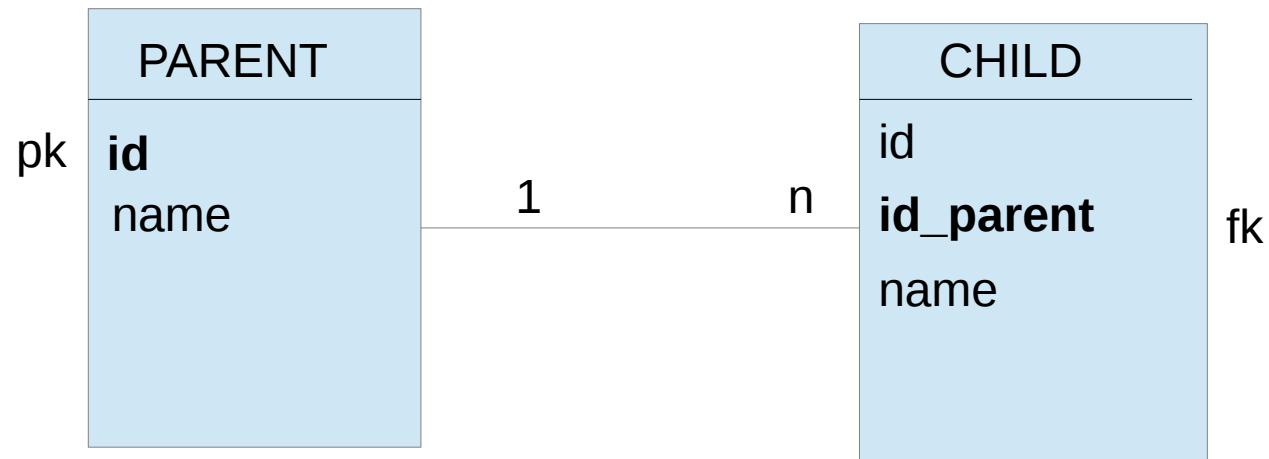
```
ALTER TABLE child  
ADD CONSTRAINT cnst_child_ref_parent  
FOREIGN KEY(id_parent)  
REFERENCES parent(id)  
ON DELETE RESTRICT
```

Пример



```
ALTER TABLE child
ADD CONSTRAINT cnst_child_ref_parent
FOREIGN KEY (id_parent)
REFERENCES parent(id)
ON DELETE RESTRICT
```

Результат



Как удалить таблицу ?

```
DROP TABLE table_name;
```

Вставка записей

Синтаксис

```
[ WITH [ RECURSIVE ] with_query [, ...] ]  
INSERT INTO table_name [ AS alias ] [ ( column_name [, ...] ) ]  
    [ OVERRIDING { SYSTEM | USER } VALUE ]  
    { DEFAULT VALUES | VALUES ( { expression | DEFAULT } [, ...] ) [, ...] | query }  
    [ ON CONFLICT [ conflict_target ] conflict_action ]  
    [ RETURNING * | output_expression [ [ AS ] output_name ] [, ...] ]
```

where *conflict_target* can be one of:

```
( { index_column_name | ( index_expression ) } [ COLLATE collation ] [ opclass ] [, ...] ) [ WHERE index_predicate ]  
ON CONSTRAINT constraint_name
```

and *conflict_action* is one of:

```
DO NOTHING  
DO UPDATE SET { column_name = { expression | DEFAULT } |  
                ( column_name [, ...] ) = [ ROW ] ( { expression | DEFAULT } [, ...] ) |  
                ( column_name [, ...] ) = ( sub-SELECT )  
                } [, ...]  
[ WHERE condition ]
```

INSERT — вставка записей

1) INSERT INTO table_name (id_table_name, login, passwd) VALUES (NULL, 'admin', '123456');

2) INSERT INTO table_name (id_table_name, login, passwd) VALUES (1, 'admin', '123456');

3) INSERT INTO table_name
VALUES (2, 'admin', '123456');

4) INSERT INTO table_name (login, passwd)
VALUES ('admin', '123456');

5) INSERT INTO table_name (id_table_name, login, password) VALUES (99, 'admin', '123456');

table_name

id_table_name	login	passwd
1	admin	123456
2	admin	123456
3	admin	123456
99	admin	123456

Удаление записей

Синтаксис

```
[ WITH [ RECURSIVE ] with_query [, ...] ]  
DELETE FROM [ ONLY ] table_name [ * ] [ [ AS ] alias ]  
    [ USING from_item [, ...] ]  
    [ WHERE condition | WHERE CURRENT OF cursor_name ]  
    [ RETURNING * | output_expression [ [ AS ] output_name ] [, ...] ]
```

<https://www.postgresql.org/docs/current/sql-delete.html>

DELETE - удаление записей

```
DELETE FROM table_name  
           WHERE id_table_name = 99
```

Изменение записей

Синтаксис

```
[ WITH [ RECURSIVE ] with_query [, ...] ]  
UPDATE [ ONLY ] table_name [ * ] [ [ AS ] alias ]  
    SET { column_name = { expression | DEFAULT } |  
        ( column_name [, ...] ) = [ ROW ] ( { expression | DEFAULT } [, ...] ) |  
        ( column_name [, ...] ) = ( sub-SELECT )  
    } [, ...]  
[ FROM from_item [, ...] ]  
[ WHERE condition | WHERE CURRENT OF cursor_name ]  
[ RETURNING * | output_expression [ [ AS ] output_name ] [, ...] ]
```

<https://www.postgresql.org/docs/current/sql-update.html>

Update

```
UPDATE    films
           SET    kind = 'Dramatic'
           WHERE   kind = 'Drama'
```


Литература

SQL. Полное руководство 3 изд [2019] Джеймс
Грофф, Пол Вайнберг, Эндрю Оппель