Język SQL w bazie PostgreSQL
(poziom podstawowy)

Michał Mackiewicz
www.gis-support.pl

MINISTERSTWO
ŚRODOWISKA

Sfinansowano ze środków
Narodowego Funduszu
Ochrony Środowiska
i Gospodarki Wodnej

Agenda

1. Wprowadzenie do relacyjnych baz danych oraz bazy PostgreSQL z rozszerzeniem o PostGIS
2. Instalacja bazy PostgreSQL z rozszerzeniem PostGIS w środowisku Windows
3. Narzędzia do pracy z bazami danych w QGIS
4. Zasilenie bazy danych z użyciem QGIS
5. Praca z klientem bazy danych (PgAdmin, psql)
6. Typy danych, konwersja typów.Wstęp do projektowania relacyjnej bazy danych.
7. Podstawy języka SQL. Wybieranie danych, instrukcja SELECT i filtrowanie wyników.
8. Funkcje skalarne i agregujące. Sortowanie wyników zapytania. Limitowanie wyników
9. Zaawansowane zapytania i przetwarzanie wyników. Grupowanie i operacje na zestawach danych. Operacje na zbiorach
10. Złączenia wewnętrzne i zewnętrzne
11. Podzapytania proste, złożone, jedno i wielowierszowe.
12. Operatory (ALL, ANY, IN, DISTINCT).
13. Modyfikacja/usuwanie danych. Wstawianie danych do tabel i widoków. Tworzenie tabel i import danych.
14. Kryteria poprawności geometrii. Ocena poprawności geometrii i metody naprawy niepoprawnych obiektów.
15. Układy współrzędnych w bazie danych

Dane

W trakcie szkolenia będą wykorzystane realne dane pochodzące z rejestrów publicznych:

Geoserwis GDOŚ
Bank Danych o Lasach
Państwowy Rejestr Granic
Baza Danych Ogólnogeograficznych

Wprowadzenie do relacyjnych baz danych oraz bazy PostgreSQL z rozszerzeniem o PostGIS

Definicja bazy danych

Techniczna: *dane cyfrowe gromadzone zgodnie z zasadami przyjętymi dla danego programu komputerowego specjalizowanego do gromadzenia i przetwarzania tych danych* (Wikipedia)

Prawna: *zbiór danych lub jakichkolwiek innych materiałów i elementów zgromadzonych według określonej systematyki lub metody, indywidualnie dostępnych w jakikolwiek sposób, w tym środkami elektronicznymi, wymagający istotnego, co do jakości lub ilości, nakładu inwestycyjnego w celu sporządzenia, weryfikacji lub prezentacji jego zawartości* (Ustawa o ochronie baz danych z 27.07.2001)

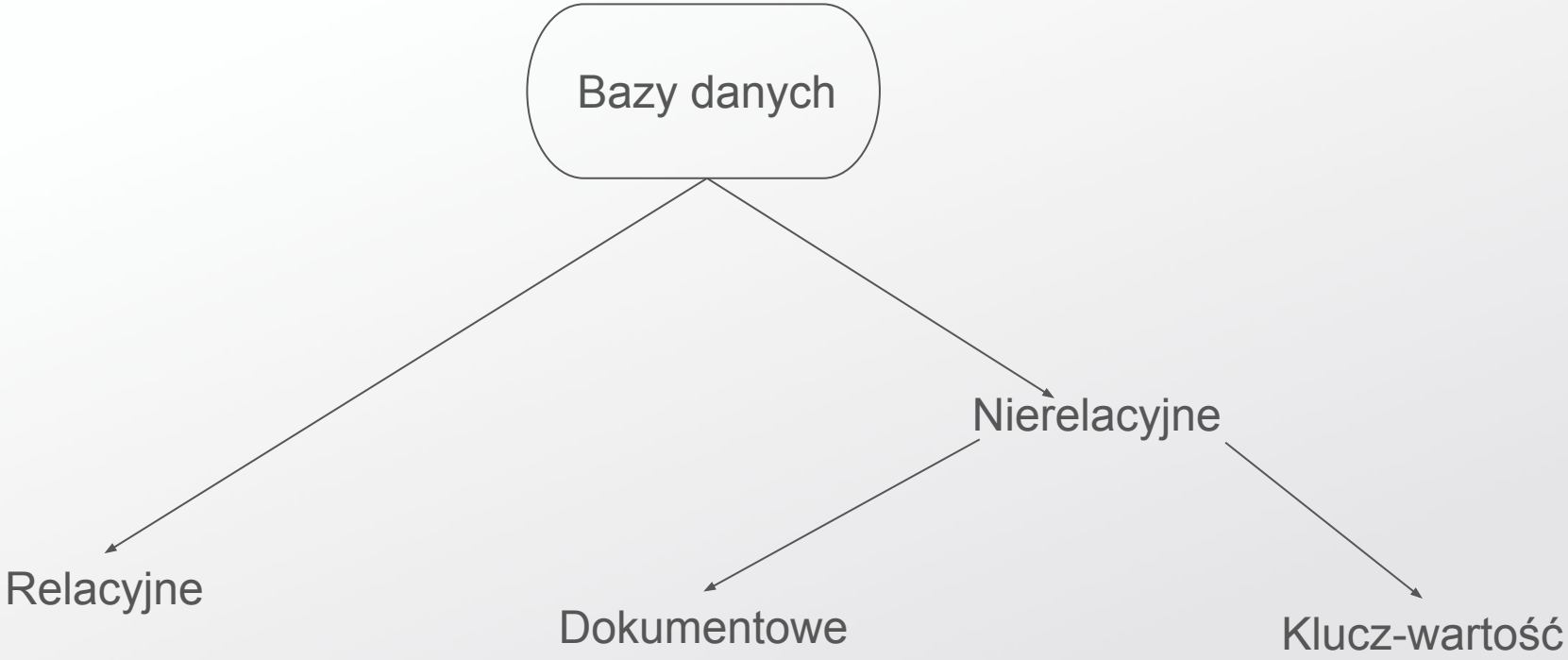
System zarządzania

Oprogramowanie bądź system informatyczny służący do zarządzania bazą danych. System zarządzania bazą danych może być również serwerem bazy danych (SBD) lub też może udostępniać bazę danych lokalnie – na określonym komputerze.

Zadania systemu zarządzania bazą danych:

- administrowanie danymi,
- zapewnienie integralności danych,
- narzędzia dostępu i przeszukiwania danych,
- narzędzia usprawniające wyszukiwanie danych (np. indeksy),
- narzędzia autoryzacji użytkowników,
- narzędzia umożliwiające równoczesny dostęp wielu użytkowników

Modele danych



Baza przechowująca dane w formie par klucz (identyfikator tekstowy lub liczbowy) - wartość (dowolne dane, zwykle tekst)

```
{
   "ŚW": "Świerk pospolity",
   "SO": "Sosna pospolita",
   "JD": "Jodła pospolita",
   "BK": "Buk zwyczajny",
   "BRZ": "Brzoza brodawkowata",
   "OL": "Olsza czarna",
   "JB": "Jabłoń domowa",
   "SW": "Sosna wejmutka",
   "CZ": "Czeremcha pospolita",
   "JW": "Klon jawor",
   "DB": "Dąb szypułkowy"
}
```

- Brak określonego modelu danych dla Wartości
- Brak obsługi relacji pomiędzy danymi
- Bardzo szybki dostęp do danych, jeśli znamy klucz (w przeciwnym razie poszukiwanie odbywa się sekwencyjnie)
- Zastosowanie jako pomocnicza struktura danych (cache)

Przykłady systemów: Redis, Tokyo Cabinet, Memcached

Baza przechowująca dane w formie dokumentów, najczęściej w formacie JSON.

```
[{
   "kod": "ŚW",
   "nazwa": {
       "polska": "Świerk
pospolity",
       "łacińska": "Picea abies"
   },
   "systematyka": {
       "rząd":"sosnowce",
       "rodzina": "sosnowate"
   },
   "wiek rębnosci": 120
},
{"kod":"SO", "nazwa_pl":"Sosna",
"wiek_rebny": "120"}
]
```

- Brak określonego modelu danych wewnątrz dokumentu
- Dane mogą być zagnieżdżone
- Brak obsługi relacji pomiędzy danymi
- Szybki dostęp do danych po identyfikatorze, możliwe też przeszukiwanie po wybranych atrybutach
- Zastosowanie - aplikacje Web, przechowywanie danych o dużej zmienności

Przykłady systemów: MongoDB, CouchDB

Baza relacyjna

Baza przechowująca dane w formie tabel o ustalonej strukturze

	sub_area	prot_categ	a_year	gat_gl	wiek	udzial	typ_dstan	typ_udz_1
2	4.59	OCH CENNE	2016	SO	46	7	IGLASTE	DOMINATED
3	4.18	OCH CENNE	2016	BRZ	38	4	LISCIASTE	MIXED
4	7.97	OCH CENNE	2016	BK	152	8	LISCIASTE	PURE
5	1.42	OCH CENNE	2016	MD	47	7	IGLASTE	DOMINATED
6	4.15	OCH CENNE	2016	SO	87	6	IGLASTE	DOMINATED
7	1.51	OCH CENNE	2016	SO	97	10	IGLASTE	PURE
8	10.81	OCH CENNE	2016	DB	137	7	LISCIASTE	DOMINATED
9	0.81	OCH CENNE	2016	DB	8	4	LISCIASTE	MIXED
10	2.96	OCH CENNE	2016	BK	20	5	LISCIASTE	DOMINATED
11	2.29	OCH CENNE	2016	MD	50	3	IGLASTE	MIXED
12	5.19	OCH CENNE	2016	SO	97	7	IGLASTE	DOMINATED
13	4.56	OCH CENNE	2016	BK	122	5	LISCIASTE	DOMINATED
14	4.78	OCH CENNE	2016	SO	97	9	IGLASTE	PURE
15	4.63	OCH CENNE	2016	DB	137	4	LISCIASTE	MIXED
16	8.85	OCH CENNE	2016	DB	162	5	LISCIASTE	DOMINATED
17	3.96	OCH CENNE	2016	DB	162	7	LISCIASTE	DOMINATED

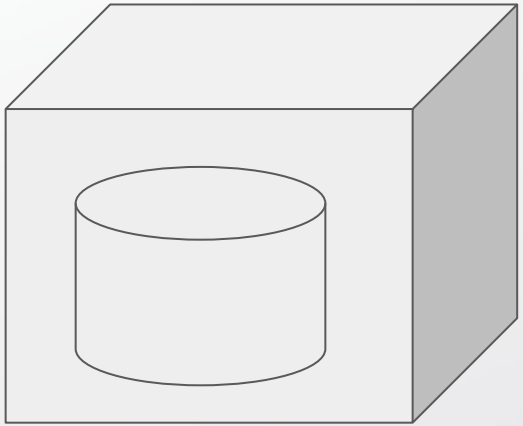
Przykłady systemów: Oracle, SQL Server, Postgres, MySQL

- Zbiór atrybutów dla pojedynczej tabeli jest z góry określony
- Obsługa relacji pomiędzy danymi
- Nie tak szybkie jak bazy klucz-wartość i dokumentowe dla poszukiwania po identyfikatorze, ale bardzo duże możliwości efektywnego przeszukiwania i analizy danych
- Zastosowanie - aplikacje Web, GIS, systemy bankowe, rejestry publiczne, aplikacje mobilne

Środowisko

Baza przechowująca dane w pojedynczym pliku na lokalnym dysku urządzenia

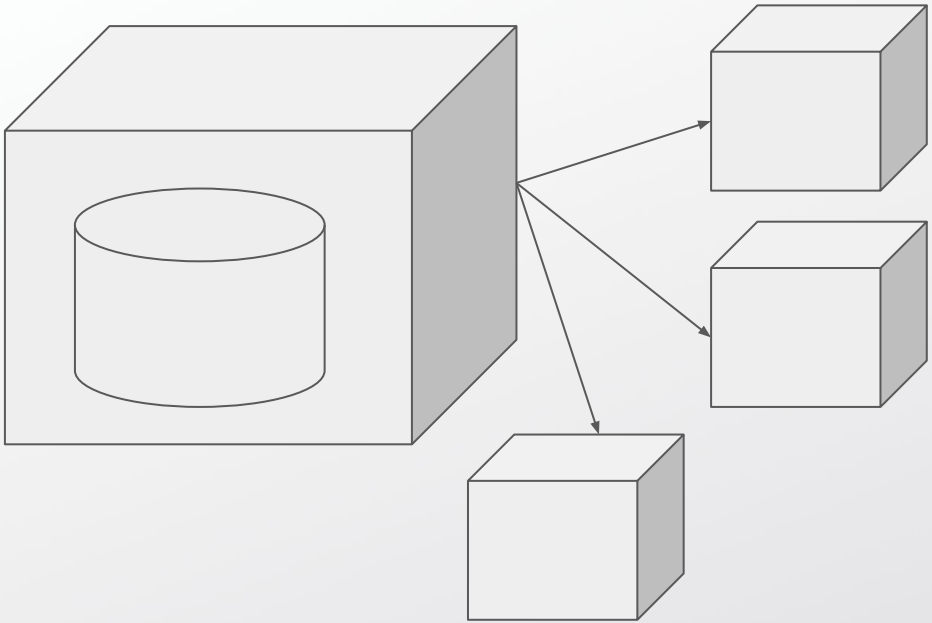
Przykłady systemów: SQLite, SpatiaLite, GeoPackage, Geobaza plikowa ESRI



- Baza jest zwykłym plikiem
- Nie wymaga stale działającego programu-serwera bazy
- Nie posiada systemu zarządzania i autoryzacji użytkowników
- Tylko jeden użytkownik może korzystać z bazy jednocześnie

Baza przechowująca dane w specjalnej strukturze plików na serwerze

Przykłady systemów: MySQL, Postgres, MS SQL, MySQL, Oracle
Geobaza wielodostępna ESRI



- Dane są przechowywane na serwerze i udostępniane poprzez sieć
- Pliki źródłowe nie są czytelne dla innego oprogramowania
- Z bazy może korzystać wielu użytkowników jednocześnie
- Wymaga stale działającego programu-serwera bazy danych

Omówienie instalacji i konfiguracji PostgreSQL w środowisku Windows

PostgreSQL

- PostgreSQL jest relacyjną bazą danych typu klient-serwer
- Hasło reklamowe - "najbardziej zaawansowana baza danych typu open source"
- Następca bazy Ingres, rozwój od 1995 roku
- Aktualna wersja stabilna 11, rozwojowa 12
- Posiada własną licencję typu open source - "PostgreSQL License"
- Pakiety dostępne na platformy Mac, Linux i Windows, działa też na BSD i Solaris

Warunki licencyjne

Licencja PostgreSQL należy do grupy tzw. liberalnych licencji open source, jest zbliżona do BSD lub MIT. Prawa autorskie należą do The PostgreSQL Global Development Group oraz Uniwersytetu Kaliforni. Warunki:

- Zezwolenie na użytkowanie, kopiowanie, modyfikację, rozpowszechnianie oprogramowania i dokumentacji do dowolnych celów i bez pisemnego zezwolenia, pod warunkiem:
- zachowania informacji o oryginalnych autorach
- niewystępowania z żadnymi roszczeniami odnośnie oprogramowania do oryginalnych autorów.

Licencja nie zawiera zapisu typu *copyleft*, co oznacza że dozwolone jest tworzenie płatnych i zamkniętych wersji (np. EnterpriseDB Advanced Server, PostgresXL).

Zalecanym sposobem instalacji w środowisku Windows jest użycie instalatora od EnterpriseDB.

Należy wybrać pakiet "PostgreSQL", a nie "EnterpriseDB Advanced Server" gdyż ten drugi jest wersją testową płatnego oprogramowania.

Możliwa jest instalacja więcej niż jednej wersji PostgreSQL na jednym systemie (np. w celu testów, aktualizacji). Aktualizacja polega na instalacji nowszej wersji, przeniesieniu danych i wyłączeniu starej.

Szczegółowy opis instalacji znajduje się w zeszycie ćwiczeń.

PostgreSQL bez dodatków posiada wsparcie dla danych geometrycznych 2D, ale bez obsługi układów współrzędnych, integracji z oprogramowaniem GIS i funkcji analiz przestrzennych.

Dodatkiem rozszerzającym PostgreSQL o te funkcje jest PostGIS - osobny projekt rozwijany przez inną grupę programistów, ale bardzo dobrze zintegrowany z PostgreSQL.

Instalacja PostGIS w środowisku Windows jest wykonywana z użyciem narzędzia "StackBuilder"

W innych systemach najlepiej skorzystać z tzw. metapakietów (apt-get install postgresql-11-postgis, brew install postgis)

Instalacja systemu PostgreSQL tworzy **klaster baz danych** (**database cluster**). Jest to kolekcja baz danych utrzymywanych przez pojedynczy proces serwera PostgreSQL. Klaster posiada swoje miejsce na dysku oraz zbiór kont użytkowników.

Wewnątrz klastra znajdują się **bazy danych** (**database**). Bazy są izolowanymi od siebie przestrzeniami do przechowywania danych - w typowej instalacji PostgreSQL, z poziomu jednej bazy nie można odwołać się do danych zapisanych w innej. Istnieje rozszerzenie, które pozwala na takie odwołania - Foreign Data Wrapper (FDW).

Baza danych jest podzielona na **schematy** (**schema**). Każda baza zawiera schemat **public**. Każdy użytkownik może w nim stworzyć tabelę. Możliwe jest tworzenie dodatkowych schematów w bazie, które można wykorzystać do logicznego grupowania danych (np. na własne i importowane ze źródeł zewnętrznych, aktualne i archiwalne) lub ograniczenia dostępu do tworzenia tabel.

W schemacie znajdują się **relacje (relations)**. Relacja w rozumieniu PostgreSQL to zbiór wierszy posiadających określone kolumny. Relacją może być:

- **tabela** (**table**) - podstawowy typ relacji, jest zapisywana na dysku, nie musi zależeć od danych już istniejących w bazie.
- **tabela obca** (**foreign table**) - tabela która odwołuje się do zewnętrznego źródła danych, może być używana tak jak natywna tabela, choć czasem z ograniczeniami (np. źródło nie pozwala na zapis)
- **widok** (**view**) - relacja która jest wynikiem wykonania określonego zapytania, dane widoku nie są zapisywane na dysku, a przeliczane na bieżąco z danych źródłowych.
- zmaterializowany widok (**materialized view**) - hybryda widoku i tabeli: dane są wynikiem zapytania na już istniejących danych, ale są zapisywane na dysku i odświeżane na żądanie.

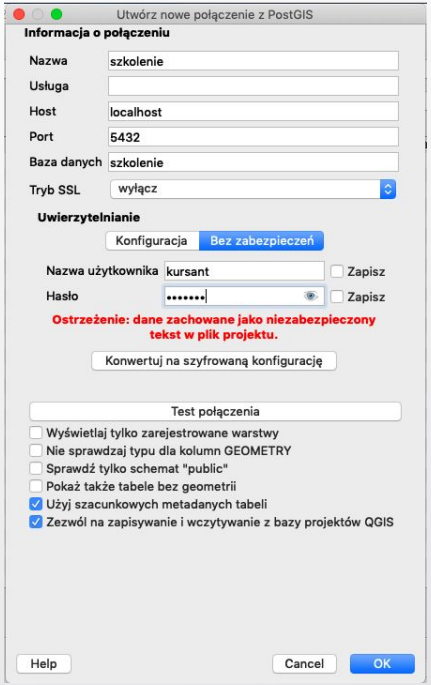
Widoki są szerzej omówione w szkoleniu zaawansowanym. Tabele obce są poza zakresem tego szkolenia.

Narzędzia do pracy z bazami danych w QGIS

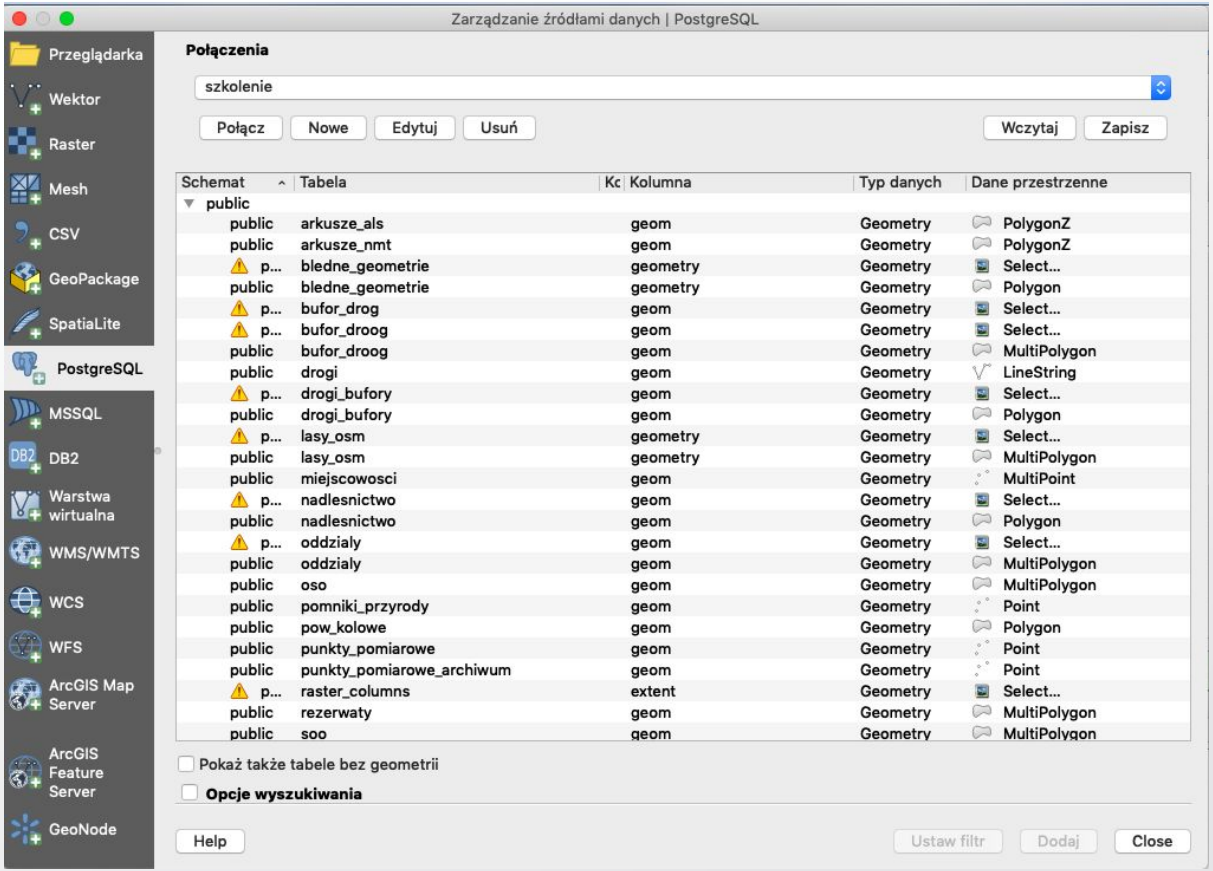
Narzędzia do obsługi

- psql - program działający w linii komend (cmd)
- pgAdmin - program działający w środowisku przeglądarki internetowej, posiada interfejs graficzny, umożliwia podgląd danych przestrzennych - instalowany razem z PostgreSQL
- QGIS - program GIS projektowany pierwotnie jako przeglądarka do danych przestrzennych w bazie PostGIS

Warstwa -> Dodaj warstwę -> Dodaj warstwę PostGIS



Połączenie z QGIS

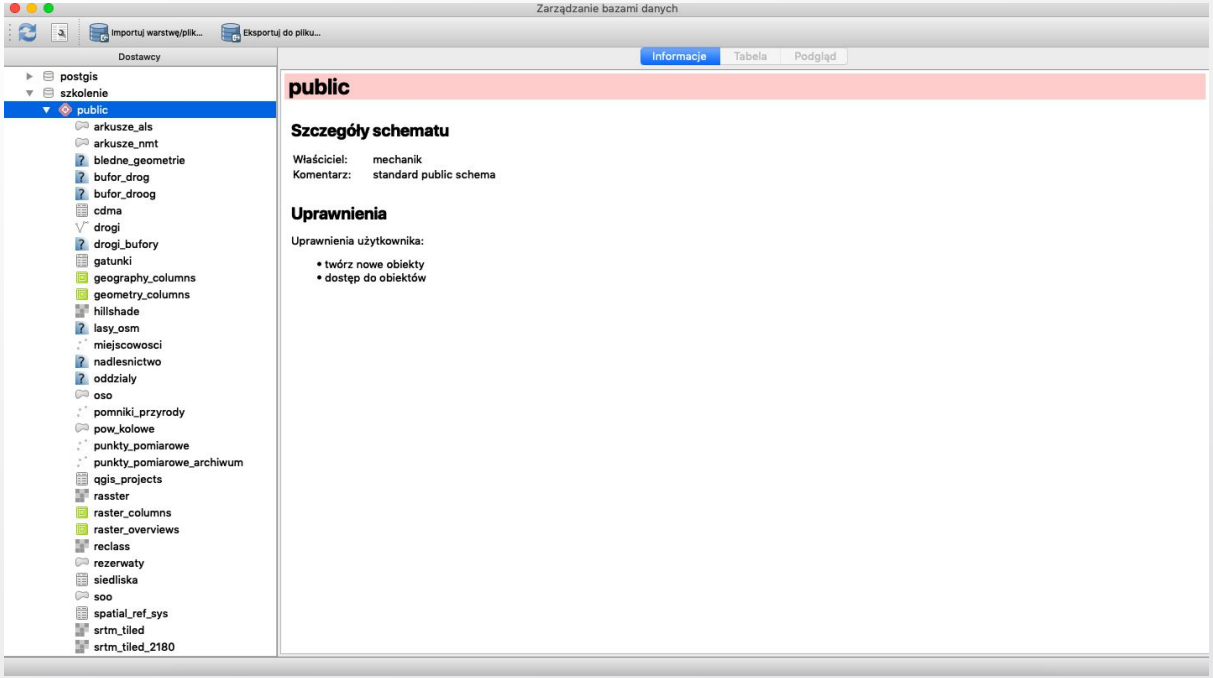


- **Połączenie**: zestaw parametrów (adres serwera, port, nazwa bazy, login i hasło). Dla każdej bazy danych musi istnieć osobne połączenie, nawet gdy znajdują się na tym samym serwerze.
- **Typ danych**: format przechowywania danych przestrzennych na serwerze - Geometry dla współrzędnych płaskich, Geography dla współrzędnych geograficznych
- **SRID**: liczbowy identyfikator układu współrzędnych (kod EPSG)
- **Metadane tabeli**: informacja o liczbie obiektów, ich typie geometrii oraz układzie współrzędnych. Szacunkowe metadane są obliczane dużo szybciej.
- **ID obiektu:** niepowtarzalny identyfikator obiektu. Jeśli tabela posiada zdefiniowany klucz główny, to będzie on automatycznie użyty. Jeśli nie, ale możliwe jest wskazanie kolumny (lub ich kombinacji) która da niepowtarzalnie wartości, to można go wskazać ręcznie. Jeśli i ten warunek nie będzie mógł być spełniony, to dane będą dostępne w trybie tylko do odczytu niezależnie od uprawnień użytkownika.

Zasilenie bazy z użyciem QGIS

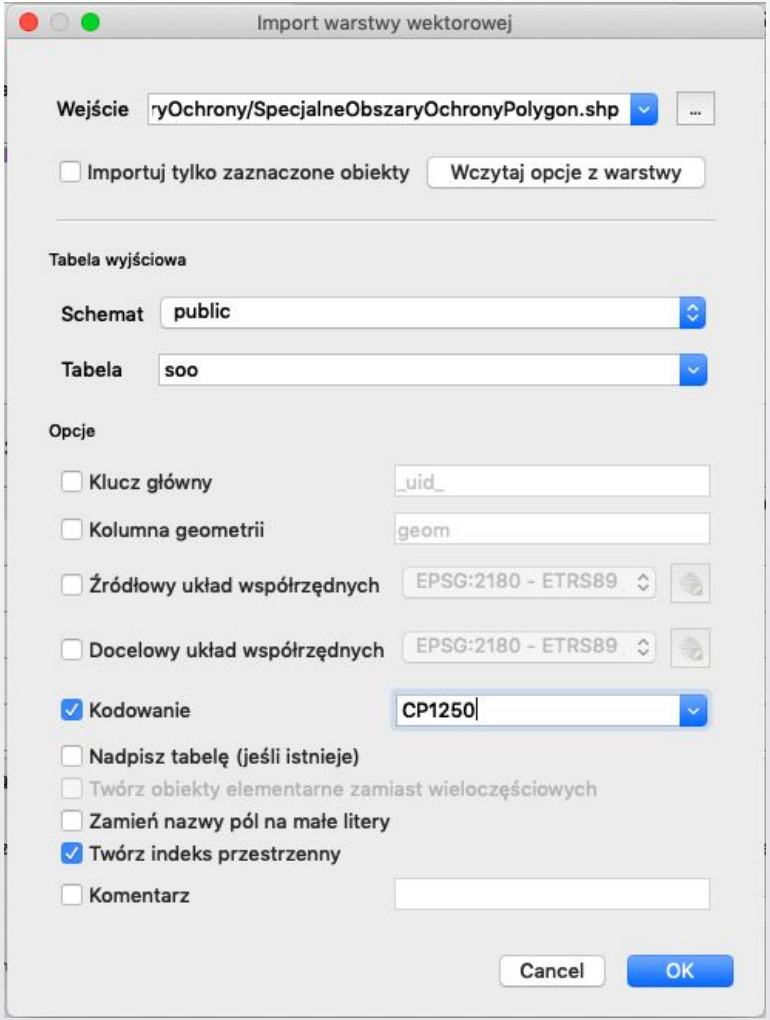
Zasilenie bazy z użyciem QGIS

- **Bazy danych -> Zarządzanie bazami danych**



Zasilenie bazy z użyciem QGIS

- Przy pomocy QGIS można zasilić bazę dowolnymi danymi wektorowymi obsługiwanymi przez QGIS: z pliku SHP, GML, CSV, warstwy tymczasowej…
- Wadą narzędzia do ładowania w QGIS jest powolne działanie dla serwerów zdalnych i danych ponad 1000 obiektów.



Praca z klientem bazy danych (psql, pgAdmin)

Praca z psql

psql jest narzędziem dostępnym z linii komend.

Uruchomienie:

po otwarciu menu Start wpisać "cmd"

odnaleźć w systemie plik psql.exe (np. C:\PostgreSQL\11\bin) i przeciągnąć go do okna konsoli

dopisać parametry:

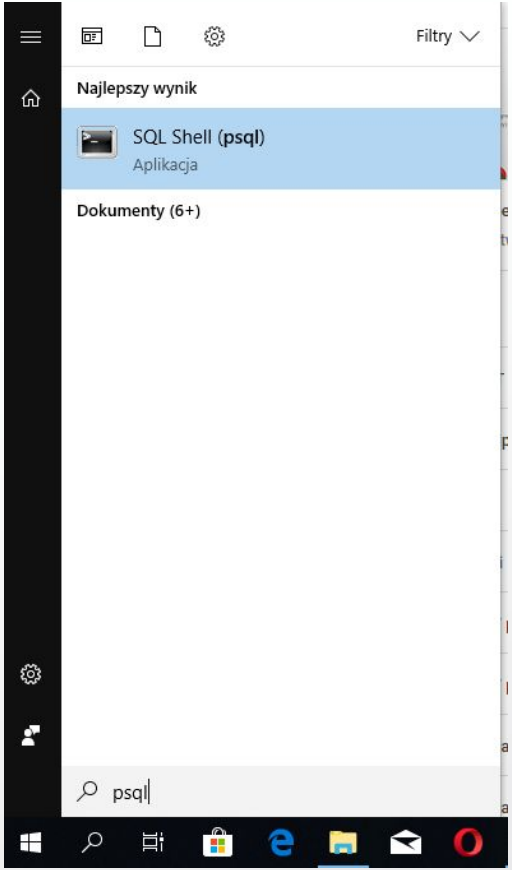
-h localhost

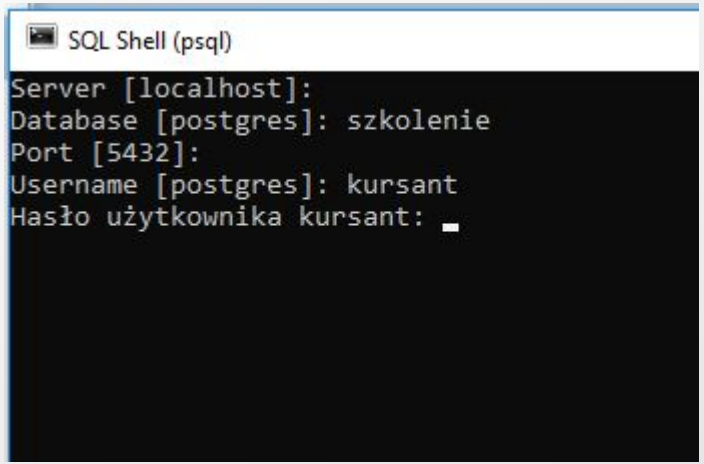
-d postgres

-U postgres

podczas wpisywania hasła nie pokazują się żadne znaki - jest to normalne.

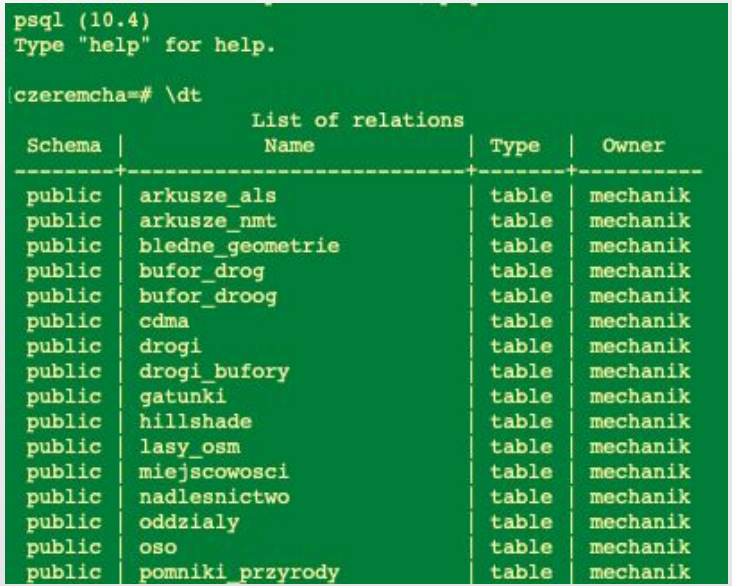
Praca z psql


Filtry
Najlepszy wynik
SQL Shell (psql)
Aplikacja
Dokumenty (6+)
psql


SQL Shell (psql)
Server [localhost]:
Database [postgres]: szkolenie
Port [5432]:
Username [postgres]: kursant
Hasło użytkownika kursant:

Komendy psql - zatwierdzane przez wciśnięcie Enter:

\l - wyświetlenie listy baz danych w systemie
\l+ - jak wyżej, ale z podaniem rozmiaru na dysku
\connect gis - przełączenie się na bazę gis
\dt - wyświetlenie listy tabel w bazie
\dt+ - jak wyżej, ale z podaniem rozmiaru na dysku
\du - wyświetlenie listy użytkowników
\dv - wyświetlenie listy widoków
\di - wyświetlenie listy indeksów



```
psql (10.4)
Type "help" for help.

czeremcha=# \dt
                List of relations
 Schema |          Name          | Type  |  Owner
--------+------------------------+-------+----------
 public | arkusze_als            | table | mechanik
 public | arkusze_nmt            | table | mechanik
 public | bledne_geometrie       | table | mechanik
 public | bufor_drog             | table | mechanik
 public | bufor_droog            | table | mechanik
 public | cdma                   | table | mechanik
 public | drogi                  | table | mechanik
 public | drogi_bufory           | table | mechanik
 public | gatunki                | table | mechanik
 public | hillshade              | table | mechanik
 public | lasy_osm               | table | mechanik
 public | miejscowosci           | table | mechanik
 public | nadlesnictwo           | table | mechanik
 public | oddzialy               | table | mechanik
 public | oso                    | table | mechanik
 public | pomniki_przyrody       | table | mechanik
```

\timing - pokazuje czas wykonania zapytania

\d <nazwa tabeli> - pokazuje dostępne kolumny w tabeli

\a - wyłącza / włącza justowanie tabeli

\x - pokazuje kolumny pionowo zamiast poziomo

\q - koniec sesji

Zapytanie może być wpisane w wielu wierszach - wiersz kończy się poprzez wciśnięcie Enter.

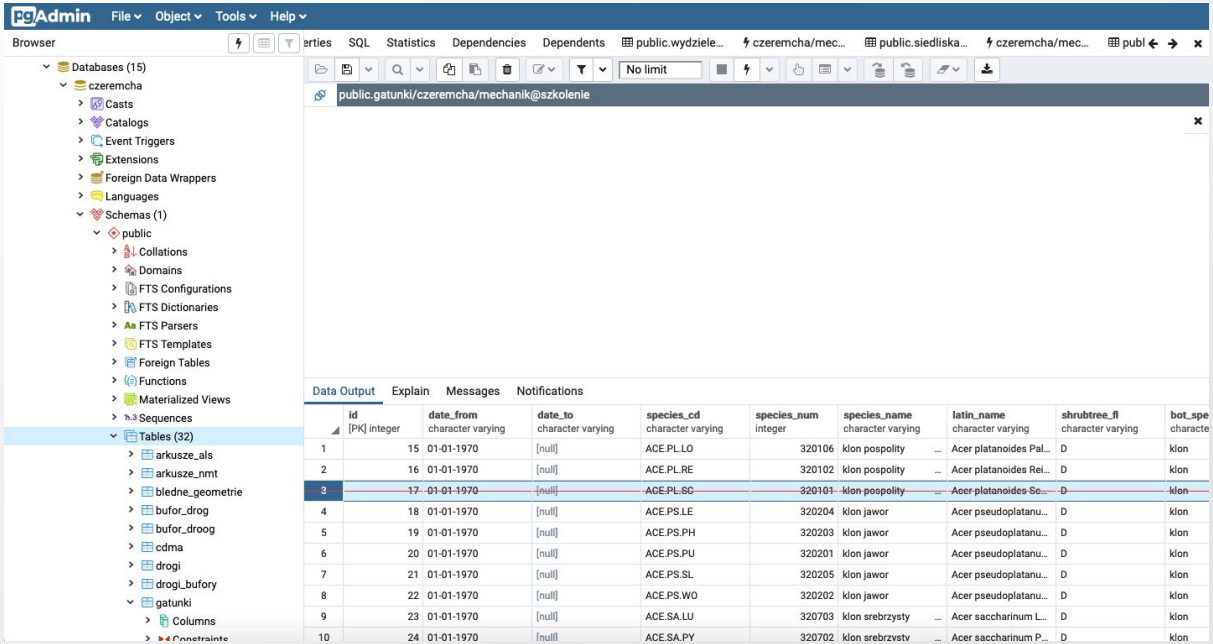
Zapytanie kończy się znakiem ; i wciśnięcie Enter.

Np.

```
szkolenie=# select *
szkolenie-# from geometry_columns;
```

Praca z pgAdmin

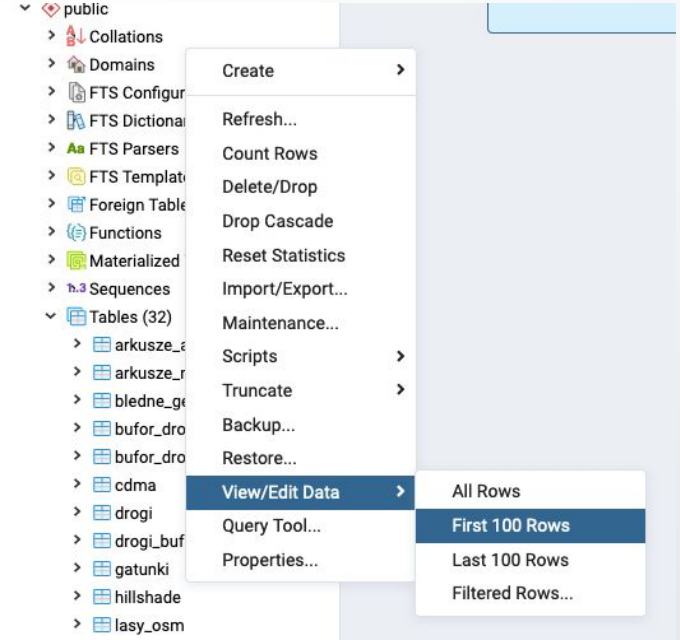
W programie pgAdmin definiuje się połączenia podobnie jak w QGIS, z tym, że wykorzystując jedną deklarację połączenia w pgAdmin można się przełączać pomiędzy różnymi bazami danych na tym samym serwerze - a w QGIS nie.



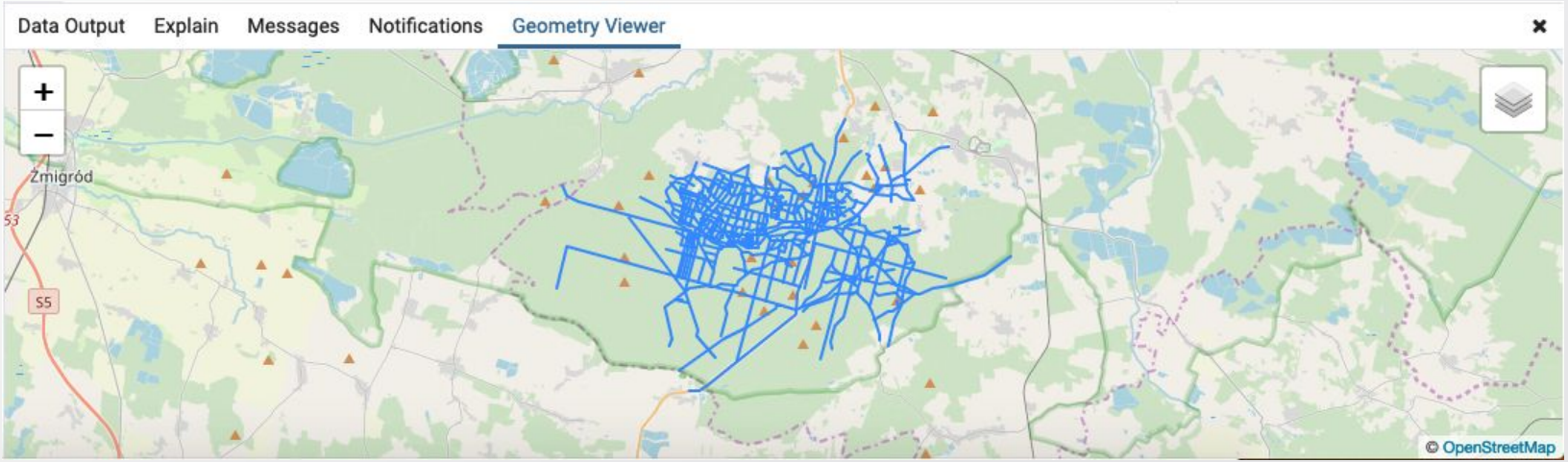
	id [PK] integer	date_from character varying	date_to character varying	species_cd character varying	species_num integer	species_name character varying	latin_name character varying	shrubtree_fl character varying	bot_spe
1	15	01-01-1970	[null]	ACE.PL.LO	320106	klon pospolity	Acer platanoides Pal...	D	klon
2	16	01-01-1970	[null]	ACE.PL.RE	320102	klon pospolity	Acer platanoides Rei...	D	klon
3	17	01-01-1970	[null]	ACE.PL.SC	320101	klon pospolity	Acer platanoides Sc...	D	klon
4	18	01-01-1970	[null]	ACE.PS.LE	320204	klon jawor	Acer pseudoplatanu...	D	klon
5	19	01-01-1970	[null]	ACE.PS.PH	320203	klon jawor	Acer pseudoplatanu...	D	klon
6	20	01-01-1970	[null]	ACE.PS.PU	320201	klon jawor	Acer pseudoplatanu...	D	klon
7	21	01-01-1970	[null]	ACE.PS.SL	320205	klon jawor	Acer pseudoplatanu...	D	klon
8	22	01-01-1970	[null]	ACE.PS.WO	320202	klon jawor	Acer pseudoplatanu...	D	klon
9	23	01-01-1970	[null]	ACE.SA.LU	320703	klon srebrzysty	Acer saccharinum L...	D	klon
10	24	01-01-1970	[null]	ACE.SA.PY	320702	klon srebrzysty	Acer saccharinum P...	D	klon

Przeglądanie danych tabeli - prawy klik - View Data - First 100 Rows

Wykonanie zapytania - ikona "błyskawicy"



Przeglądanie danych przestrzennych - ikona "oka" 
Dane w układzie współrzędnych WGS84 są pokazywane na podkładzie mapowym OpenStreetMap Standard, w pozostałych układach - bez podkładu mapowego.



Typy danych, konwersja typów

Kolumny i typy

- Różnica pomiędzy arkuszem kalkulacyjnym a bazą danych
- Kolumna może przechowywać tylko dane określonego typu, np. liczba całkowita, tekst
- Podobieństwo do tabeli atrybutów w GIS
- PostgreSQL posiada 56 typów + warianty tablicowe (lista wartości w pojedynczym polu)
- Silne typowanie: nie można wpisać do kolumny typu INTEGER wartości tekstowej itp. - w niektórych innych systemach baz danych można to zrobić

Typy tekstowe

- Typ **CHAR**: tekst stałej długości (np. suma kontrolna)
- Typ **VARCHAR**: tekst zmiennej długości, opcjonalny typmod dla ograniczenia maksymalnej długości (alias - **CHARACTER VARYING)**
- Typ **TEXT**: tekst dowolnej długości
- Typ **XML**: dokument XML - musi być well-formed
- Typ **TSVECTOR**: dla wyszukiwania pełnotekstowego

Typy liczbowe

- Typ **INTEGER**: liczba całkowita 32 bit (alias - **INT4**)
- Typ **BIGINT**: liczba całkowita 64 bit (alias - **INT8**)
- Typ **REAL**: liczba zmiennoprzecinkowa pojedynczej precyzji
- Typ **DOUBLE PRECISION**: liczba zmiennoprzecinkowa podwójnej precyzji (alias - **FLOAT**)
- Typ **NUMERIC**: liczba dziesiętna o zdefiniowanej precyzji (alias - **DECIMAL**)

Typy daty i czasu

- Typ **DATE**: tylko data
- Typ **TIME**: tylko czas (może być WITH TIME ZONE i WITHOUT TIME ZONE)
- Typ **TIMESTAMP**: data i czas - w wariantach WITH TIME ZONE i WITHOUT TIME ZONE

Inne typy

- Typ **BYTEA**: dane binarne
- Typ **BOOLEAN**: wartość prawda/fałsz
- Typ **HSTORE**: pary klucz-wartość
- Typy **JSON** i **JSONB**: obiekty JSON (można użyć do wpisania wielu wartości w jedno pole)
- Typ **OID**: identyfikator dla obiektów systemowych
- Pseudo-typ **SERIAL**: kolumna **INTEGER**+sekwencja+wartość domyślna, używany do nadawania identyfikatorów
- Typy **POINT**, **PATH**, **POLYGON**: typy grafiki wektorowej 2D
    - - **nie używać do danych GIS!**

Typy PostGIS

- Typ **GEOMETRY**: najczęściej używany. Dowolny układ współrzędnych, wszystkie obliczenia są wykonywane na współrzędnych płaskich, wyniki zwracane w jednostkach układu. Opcjonalny typmod na typ geometrii (POINT, LINESTRING, POLYGON…) oraz układ współrzędnych
- Typ **GEOGRAPHY**: dla współrzędnych geograficznych długość-szerokość, obliczenia wykonywane metodami geodezji wyższej, wyniki i parametry podawane są w metrach.

Konwersja typów

- Konwersja typów nazywa się rzutowaniem (CAST)
- Rzutowanie w standardzie SQL: CAST ('2019-09-05' AS date); - słowo kluczowe CAST, następnie w nawiasie wartość, słowo kluczowe AS i nazwa typu.
- Dodatkowo w PostgreSQL jest dostępny operator ::, np. '2010-09-05'::date

Uwaga: słowo kluczowe AS ma w SQL także inną rolę - może służyć do nadawania aliasów dla kolumn - szczegóły w części poświęconej zapytaniom SELECT.

Wstęp do projektowania relacyjnej bazy danych

Pojęcia:

Encja - *"reprezentacja wyobrażonego lub rzeczywistego obiektu (grupy obiektów)"*

np. rezerwat, decyzja, uchwała, wydzielenie

Atrybut encji - *"dowolne szczegóły (cechy) mające znaczenie dla danej encji*"

np. typ rezerwatu, data wejścia w życie, numer uchwały, typ siedliskowy lasu

Związki encji:

Opcjonalność - czy encja może lub musi wystąpić w związku z innymi encjami? (np. wydzielenie musi należeć do obrębu)

Krotność - czy encja wchodzi w związki 1:1, 1:n, m:n?

Model logiczny

Reprezentacja bazy w postaci czytelnej dla człowieka, np. diagramów
Stosowane typy diagramów:

ER (Entity-Relationship) - posługuje się pojęciem encji, opisuje związki pomiędzy nimi, teoretycznie nie musi zawierać informacji o atrybutach. Ta sama encja może występować wielokrotnie w jednym diagramie.

UML (Universal Modelling Language) - opis klas obiektów, ich atrybutów i powiązań, każda klasa obiektu występuje tylko raz.

Model logiczny jest tworzony przez projektanta bazy. Istnieje oprogramowanie do wspomagania tworzenia modeli logicznych.

Model fizyczny

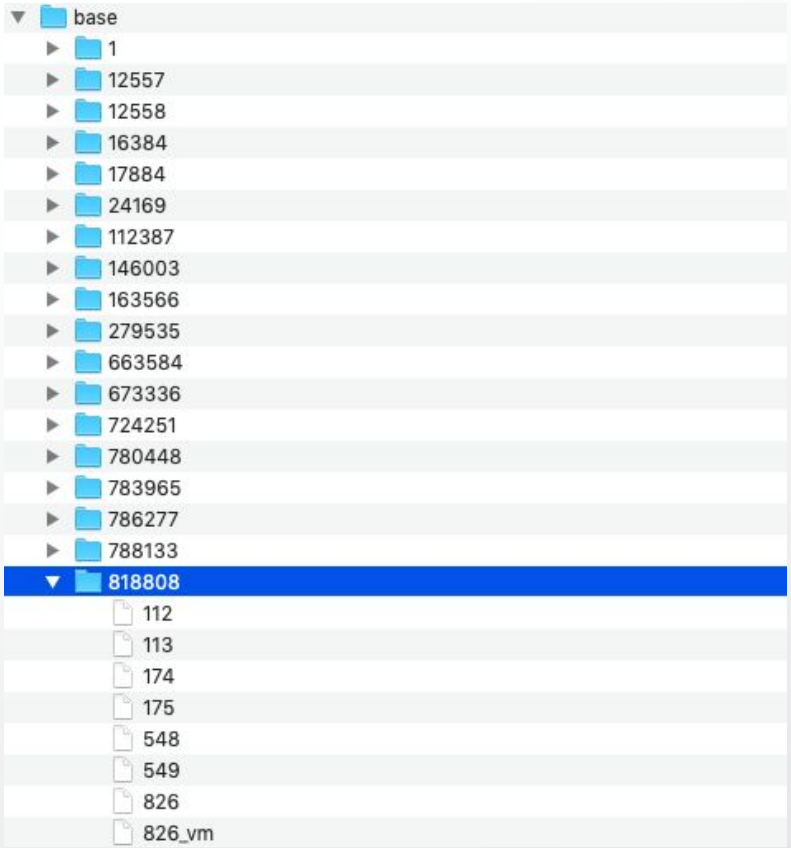
Implementacja konkretnego modelu logicznego w konkretnym systemie zarządzania bazą danych i konkretnym środowisku sprzętowo-programowym.

Podział na:

- pliki (*files*)
- strony (*pages*)
- krotki (*tuples*)

Model fizyczny zwykle nie jest tworzony przez człowieka, administrator bazy jedynie ustawia pewne parametry mające wpływ na budowę modelu fizycznego.

Model fizyczny



Dane są składowane w katalogu określonym przez parametr data_directory (np. C:\PostgreSQL\11\data) \ base

Każda baza danych ma swój katalog, który ma nazwę równą jej identyfikatorowi systemowemu (OID) - nazwę można zmienić, a OID nie

Wewnątrz katalogu bazy znajdują się pliki, które nie przekraczają 1 GB - jeśli tabela jest większa, to zawiera się w wielu plikach.

Narzędzia

Oprogramowanie desktop - komercyjne:

Enterprise Architect
Toad
SQL Maestro
StarUML

Oprogramowanie desktop - open source:
pgModeler
Moskitt
Modelio
White Star UML

Online:

Pony ORM Editor
Vertabelo

W nazwach kolumn i tabel (nazywanych identyfikatorami, *identifiers*) używanych bez cudzysłowu mogą występować następujące znaki:

- małe litery (także litery spoza alfabetu łacińskiego - **polskie znaki są dozwolone**)
- znaki specjalne _ oraz $
- cyfry

muszą rozpoczynać się literą lub znakiem _, oraz nie mogą być słowami kluczowymi SQL (np. select, insert, update, delete, drop, table, join, natural, right, left, lateral, truncate, user, role…)

Identyfikatory które będą zawierane w "podwójny cudzysłów" mogą zawierać słowa i dowolne znaki - wszystkie znaki specjalne oraz spacje.

Maksymalna długość identyfikatora to 63 bajty. 1 znak alfabetu łacińskiego to 1 bajt. Polskie znaki zajmują 2 bajty. (kalkulator bajtów: https://mothereff.in/byte-counter)

Podstawy języka SQL

- **S**tructured **Q**uery **L**anguage
- *Strukturalny język zapytań używany do tworzenia, modyfikowania baz danych oraz do umieszczania i pobierania danych z baz danych.* (Wikipedia)
- Jest językiem deklaratywnym, co oznacza że użytkownik definiuje **co** chce uzyskać, natomiast decyzję o tym **jak** to zrobić podejmuje maszyna
- Jest zdefiniowany standard (ANSI-SQL) oraz implementacje (dialekty) mniej lub bardziej zgodne ze standardem
- Najważniejsze instrukcje i konwencje są wspólne dla wszystkich baz, ale…
- Różne sposoby formatowania zapytań, różne nazwy funkcji, różne podejście do typów danych

Elementy języka SQL

- **DML** - Data Manipulation Language

  Polecenia wyszukiwania, dodawania, edycji i usuwania danych

- **DDL** - Data Definition Language

  Polecenia tworzenia, zmiany i usuwania struktur danych (tabel, kolumn…)

- **DCL** - Data Control Language

  Polecenia kontroli dostępu do danych (tylko bazy klient-serwer)

Wybieranie danych, instrukcja SELECT i filtrowanie wyników

Instrukcja SELECT

- Zapytania SELECT służą przede wszystkim do pobierania danych
- W PostgreSQL **nie** jest gwarantowane, że SELECT nie zmieni danych, np. SELECT AddGeometryColumn...
- Zapytanie SELECT może operować na "0 lub więcej relacji", co oznacza, że niekoniecznie musi odnosić się do danych zapisanych w bazie - może również sprawdzać ustawienia systemowe, lub… posłużyć za kalkulator
- Przykłady SELECT na 0 relacji:
  - SELECT 1; -- zapytanie zwróci 1, służy np. do sprawdzenia czy system działa
  - SELECT PostGIS_full_version(); -- sprawdzenie wersji PostGIS
  - SELECT now(); -- sprawdzenie aktualnej daty i godziny w systemie
  - SELECT 50000::real / 365::real; -- użycie PostgreSQL jako kalkulatora

Przykłady zapytań SELECT

```
1  SELECT * FROM wydzielenia;
```

Data Output

	id integer	geom geometry	a_i_num bigint	adr_for character varying (25)	area_type character varying (11)	site_type character varying (9)	silvicult character varying (5)	forest_fun character varying (
1	1	0106000020840…	223010205	02-23-1-06-194 -a -00	PAS GRAN	[null]	[null]	[null]
2	2	0106000020840…	223020333	02-23-1-04-149 -b -00	D-STAN	LWYŻŚW	O	OCHR
3	3	0106000020840…	223005550	02-23-2-10-21 -a -00	D-STAN	LŚW	O	OCHR
4	4	0106000020840…	223012227	02-23-2-13-188 -j -00	PS	[null]	[null]	[null]
5	5	0106000020840…	223020812	02-23-2-14-215 -d -00	D-STAN	LŚW	O	OCHR
6	6	0106000020840…	223001181	02-23-1-03-125 -g -00	D-STAN	LWYŻŚW	O	OCHR
7	7	0106000020840…	223020164	02-23-1-07-84 -b -00	D-STAN	LGŚW	O	OCHR
8	8	0106000020840…	223016867	02-23-2-10-24 -b -00	D-STAN	LMŚW	O	OCHR
9	9	0106000020840…	223000778	02-23-1-02-42A -a -00	D-STAN	LŚW	O	OCHR
10	10	0106000020840…	223014011	02-23-1-07-91 -b -00	D-STAN	LMGŚW	O	OCHR

Przykłady zapytań SELECT

```
SELECT adr_for, site_type, sub_area FROM wydzielenia;
```

Data Output

	adr_for character varying (25)	site_type character varying (9)	sub_area double precision
1	02-23-1-06-194 -a -00	[null]	1.32
2	02-23-1-04-149 -b -00	LWYŻŚW	1.13
3	02-23-2-10-21 -a -00	LŚW	1.94
4	02-23-2-13-188 -j -00	[null]	0.23
5	02-23-2-14-215 -d -00	LŚW	14.49
6	02-23-1-03-125 -g -00	LWYŻŚW	1.69
7	02-23-1-07-84 -b -00	LGŚW	19.46
8	02-23-2-10-24 -b -00	LMŚW	5.54
9	02-23-1-02-42A -a -00	LŚW	1.95
10	02-23-1-07-91 -b -00	LMGŚW	2.33

Przykłady zapytań SELECT

Query Editor | Query History

```
1  SELECT adr_for AS "adres leśny", site_type AS "typ siedliskowy", sub_area AS powierzchnia FROM wydzielenia;
```

Data Output

	adres leśny character varying (25)	typ siedliskowy character varying (9)	powierzchnia double precision
1	02-23-1-06-194 -a -00	[null]	1.32
2	02-23-1-04-149 -b -00	LWYŻŚW	1.13
3	02-23-2-10-21 -a -00	LŚW	1.94
4	02-23-2-13-188 -j -00	[null]	0.23
5	02-23-2-14-215 -d -00	LŚW	14.49
6	02-23-1-03-125 -g -00	LWYŻŚW	1.69
7	02-23-1-07-84 -b -00	LGŚW	19.46
8	02-23-2-10-24 -b -00	LMŚW	5.54
9	02-23-1-02-42A -a -00	LŚW	1.95
10	02-23-1-07-91 -b -00	LMGŚW	2.33

Klauzula AS - pozwala na nadanie aliasu dla nazwy kolumny, dzięki czemu nazwa w wyniku zapytania może być inna niż zapisana w systemie. Czasem jest to koniecznością, np. jeśli łączy się dane z dwóch tabel o powtarzających się nazwach kolumn.

Reguły

- Słowa kluczowe mogą być pisane zarówno wielkimi, jak i małymi literami. Konwencją jest pisanie wielkimi, ale nie jest to wymagane
- Kolumny mogą mieć dowolnie długie nazwy i zawierać dowolne znaki, jeśli jednak:
    - zawierają spacje, znaki specjalne
    - zawierają słowa kluczowe SQL (np. select, table, insert, order)
    - zaczynają się od cyfry
    - zawierają wielkie litery
- **ich nazwy muszą być ujęte w "podwójny cudzysłów".**

Dla osób dobrze znających QGIS: zasady formułowania wyrażeń w QGIS zostały zaczerpnięte z SQL, dlatego składnia zapytań w bazie danych będzie podobna do składni wyrażeń w QGIS.

Ćwiczenia

- Wybierz wszystkie wiersze i wszystkie kolumny z tabeli "gatunki"
- Wybierz kolumny: "site_type_cd", "site_type_name" z tabeli "siedliska"

- Wybierz wszystkie wiersze i wszystkie kolumny z tabeli "gatunki"

SELECT * FROM gatunki;

	id integer	date_from character varying	date_to character varying	species_cd character varying	species_num integer	species_name character varying	latin_name character varying	shrubtree_fl character varying	bot_ char
1	15	01-01-1970	[null]	ACE.PL.LO	320106	klon pospolity ...	Acer platanoides Pal...	D	klon
2	16	01-01-1970	[null]	ACE.PL.RE	320102	klon pospolity ...	Acer platanoides Rei...	D	klon
3	17	01-01-1970	[null]	ACE.PL.SC	320101	klon pospolity ...	Acer platanoides Sc...	D	klon
4	18	01-01-1970	[null]	ACE.PS.LE	320204	klon jawor	Acer pseudoplatanu...	D	klon
5	19	01-01-1970	[null]	ACE.PS.PH	320203	klon jawor	Acer pseudoplatanu...	D	klon

- Wybierz kolumny: "site_type_cd", "site_type_name" z tabeli "siedliska"

SELECT site_type_cd, site_type_name FROM siedliska;

	site_type_cd character varying	site_type_name character varying
1	BB	Bór bagienny ...
2	BG	Bór górski
3	BGW	Bór górski wilgotny ...
4	BMB	Bór mieszany bagien...
5	BMG	Bór mieszany górski ...
6	BMW	Bór mieszany wilgot...

Filtrowanie

- Do filtrowania służy klauzula WHERE
- W klauzuli WHERE można używać operatorów: =, !=, >, >=, <, <=, IN (lista wartości), LIKE (wyszukiwanie przybliżone)
- Wymienione operatory nie działają dla wartości NULL - zawsze zwrócą fałsz. Do porównania z wartością NULL trzeba stosować specjalne operatory IS NULL oraz IS NOT NULL.
- Warunki można łączyć wykorzystując operatory logiczne: AND, OR, NOT

Filtrowanie

- W klauzuli WHERE można podać jeden lub więcej warunków. Warunki są łączone operatorami logicznymi: AND (i), OR (lub)
- Nazwy kolumn podajemy bez cudzysłowu lub w podwójnym cudzysłowie.
- Wartości liczbowe podajemy bez cudzysłowu, separatorem dziesiętnym jest zawsze kropka.
- Wartości tekstowe podajemy zawsze w pojedynczym cudzysłowie.

SELECT * FROM wydzielenia WHERE rotat_age > 100;

site_type character varying (9)	silvicult character varying (5)	forest_fun character varying (6)	stand_stru character varying (8)	rotat_age bigint	sub_area double precision	prot_categ character varying (9)	spe cha
[null]	[null]	[null]	[null]	0	1.32	[null]	BRZ
LŚW	O	OCHR	DRZEW	90	1.94	OCH USZK	SO
[null]	[null]	[null]	[null]	0	0.23	[null]	[null
LGŚW	O	OCHR	DRZEW	80	19.46	OCH USZK	ŚW

SELECT * FROM wydzielenia WHERE rotat_age < 100;

silvicult character varying (5)	forest_fun character varying (6)	stand_stru character varying (8)	rotat_age bigint	sub_area double precision	prot_categ character varying (9)	species_cd character varying (9)	part_cd character
O	OCHR	DRZEW	120	1.13	OCH USZK	BK	6
O	OCHR	DRZEW	120	14.49	OCH USZK	BK	2
O	OCHR	DRZEW	130	1.69	OCH USZK	DB	2

SELECT * FROM wydzielenia WHERE site_type = 'LŚW';

	id integer	geom geometry	a_i_num bigint	adr_for character varying (25)	area_type character varying (11)	site_type character varying (9)	silvicult character varying (5)	forest_fun character varying (
1	3	0106000020840...	223005550	02-23-2-10-21 -a -00	D-STAN	LŚW	O	OCHR
2	5	0106000020840...	223020812	02-23-2-14-215 -d -00	D-STAN	LŚW	O	OCHR

SELECT * FROM wydzielenia WHERE rotat_age < 100 AND species_cd = 'BRZ';

ying (6)	stand_stru character varying (8)	rotat_age bigint	sub_area double precision	prot_categ character varying (9)	species_cd character varying (9)	part_cd character varying (3)	spec_age bigint	a_year bigint
	[null]	0	1.32	[null]	BRZ	[null]	50	2018
	DRZEW	60	5.54	OCH USZK	BRZ	4	60	2018
	DRZEW	60	1.19	OCH USZK	BRZ	8	55	2018

SELECT * FROM wydzielenia WHERE site_type = 'LŚW' OR gat_gl = 'BK';

	site_type character varying (9)	silvicult character varying (5)	forest_fun character varying (6)	stand_stru character varying (8)	rotat_age bigint	sub_area double precision	prot_categ character varying (9)	spe cha
	LWYŻŚW	O	OCHR	DRZEW	120	1.13	OCH USZK	BK
	LŚW	O	OCHR	DRZEW	90	1.94	OCH USZK	SO
	LŚW	O	OCHR	DRZEW	120	14.49	OCH USZK	BK
	LŚW	O	OCHR	DRZEW	130	1.95	OCH USZK	DB

Filtrowanie

SELECT * FROM wydzielenia WHERE species_cd LIKE 'B%';

	silvicult character varying (5)	forest_fun character varying (6)	stand_stru character varying (8)	rotat_age bigint	sub_area double precision	prot_categ character varying (9)	species_cd character varying (9)	part_cd charact
	[null]	[null]	[null]	0	1.32	[null]	BRZ	[null]
	O	OCHR	DRZEW	120	1.13	OCH USZK	BK	6
	O	OCHR	DRZEW	120	14.49	OCH USZK	BK	2

SELECT * FROM wydzielenia WHERE site_type ILIKE 'bm%';

	id integer	geom geometry	a_i_num bigint	adr_for character varying (25)	area_type character varying (11)	site_type character varying (9)	silvicult character varying (5)	forest_fun character varying (6)
	14	0106000020840...	223016946	02-23-2-10-46 -b -00	D-STAN	BMŚW	O	OCHR
	36	0106000020840...	223006700	02-23-2-09-114 -a -00	D-STAN	BMŚW	O	OCHR
	52	0106000020840...	223003087	02-23-1-06-218 -c -00	D-STAN	BMGŚW	O	OCHR

Funkcje skalarne i agregujące

Funkcja jest fragmentem kodu - w języku SQL bądź innym obsługiwanym przez bazę - nadającym się do wielokrotnego użytku.

W PostgreSQL funkcje nie muszą być "czyste", tj. przekształcać zbioru argumentów w zbiór wartości bez pozostawiania trwałych zmian w bazie.

Funkcja w PostgreSQL jest definiowana przez nazwę oraz liczbę i typy argumentów - w razie pomyłki w argumentach, komunikat o błędzie będzie brzmiał *function does not exist*.

Wyróżniamy funkcje skalarne (operujące na pojedynczych wierszach) i agregujące (operujące na grupach wierszy).

Oprócz wbudowanych funkcji, użytkownik może tworzyć własne.

SELECT lower(area_type) FROM wydzielenia;

Funkcja lower zmienia wielkość liter w podanym tekście na małe.

SELECT upper(naz_glowna) FROM miejscowosci;

Funkcja upper zmienia wielkość liter w podanym tekście na wielkie.

SELECT replace(prot_categ,'OCH','OCHRONNE') FROM wydzielenia;

Funkcja replace przyjmuje 3 argumenty: tekst do zmiany, fragment podlegający zmianie oraz fragment, który ma zostać wstawiony.

SELECT split_part(adr_for, '-', 5) FROM wydzielenia;

Funkcja split_part dzieli tekst na fragmenty według podanego separatora, oraz zwraca żądany fragment.

SELECT trim(trailing ' ' from height_grp), length(height_grp), length(trim(trailing ' ' from height_grp)) FROM gatunki;

Funkcja trim obcina żądane znaki na początku, końcu lub z obu stron tekstu (najczęstsze zastosowanie: spacje na końcu). Funkcja length oblicza długość ciągu znaków.

SELECT substring(adr_for from 1 for 5) FROM wydzielenia;

Funkcja substring ekstrahuje fragment tekstu według położenia pierwszego i ostatniego znaku.

```
SELECT ceil(sub_area) FROM wydzielenia;
SELECT floor(sub_area) FROM wydzielenia;
SELECT round(sub_area) FROM wydzielenia;
```

Funkcja ceil zaokrągla liczbę dziesiętną w górę, floor - w dół, round - do końcówki 5 w dół, powyżej w górę.

```
SELECT round(sub_area::numeric,2) FROM wydzielenia;
```

Funkcja round na typie numeric z podaniem drugiego argumentu - precyzji umożliwia zaokrąglenie do żądanej precyzji.

```
SELECT log(spec_age) FROM wydzielenia;
SELECT ln(spec_age) FROM wydzielenia;
```

Funkcja log oblicza logarytm przy podstawie 10, ln - przy podstawie e.

```
SELECT sqrt(sub_area) FROM wydzielenia WHERE sub_area >= 0;
```

Funkcja sqrt oblicza pierwiastek kwadratowy. Dla liczb ujemnych zwraca błąd (ERROR: cannot take square root of a negative number) stąd konieczność ograniczenia do liczb nieujemnych.

```
SELECT abs(-3);
SELECT abs(3);
SELECT sign(-3);
```

Funkcja abs oblicza wartość bezwzględną, sign - znak liczby.

```
SELECT radians(90);
SELECT degrees(pi()/2);
```

Funkcja radians przelicza stopnie na radiany, degrees - radiany na stopnie.

SELECT max(spec_age) FROM wydzielenia;

Funkcja max zwraca największą wartość ze zbioru.

SELECT min(spec_age) FROM wydzielenia;

Funkcja min zwraca najmniejszą wartość ze zbioru.

SELECT sum(spec_age) FROM wydzielenia;

Funkcja sum oblicza sumę wartości ze zbioru.

SELECT count(*) FROM wydzielenia;

Funkcja count zwraca liczbę wartości w zbiorze.

Sortowanie wyników zapytania

Limitowanie wyników

Sortowanie i limitowanie

- Do sortowania służy klauzula ORDER BY
- Domyślny kierunek sortowania - rosnąco, zmiana poprzez klauzulę DESC
- Domyślnie wartości puste są na początku, zmiana przez klauzulę NULLS LAST
- Możliwe jest sortowanie po więcej niż 1 kolumnie
- Ograniczenie liczby zwracanych wierszy - klauzula LIMIT

```
SELECT adr_for AS adres, species_cd AS gatunek, rotat_age AS wiek_rebny FROM wydzielenia ORDER BY gatunek;

SELECT adr_for AS adres, species_cd AS gatunek, rotat_age AS wiek_rebny FROM wydzielenia ORDER BY gatunek ASC;

SELECT adr_for AS adres, species_cd AS gatunek, rotat_age AS wiek_rebny FROM wydzielenia ORDER BY gatunek ASC NULLS LAST;

SELECT adr_for AS adres, species_cd AS gatunek, rotat_age AS wiek_rebny FROM wydzielenia ORDER BY gatunek ASC NULLS LAST LIMIT 5;
```

- Wybierz wszystkie wiersze z tabeli "gatunki" sortując rosnąco po kolumnie "bot_species"
- Wybierz 10 wierszy z tabeli "gatunki" o najwyższych wartościach kolumny "species_num"

Uwzględnij fakt, że w kolumnie "species_num" występują wartości NULL.

Wybierz wszystkie wiersze z tabeli "gatunki" sortując rosnąco po kolumnie "bot_species"

```
SELECT * FROM gatunki ORDER BY bot_species ASC;
```

Data Output

	date_from character varying	date_to character varying	species_cd character varying	species_num integer	species_name character varying	latin_name character varying	shrubtree_fl character varying	bot_species character varying
0	01-01-1970	[null]	AGR	590100	agrest pospolity ...	Ribes grossularia ...	K	agrest
2	01-01-1970	[null]	AKT.P	1160200	aktinidia pstrolistna ...	Actinidia kolomikta ...	K	aktinidia
1	01-01-1970	[null]	AKT.O	1160100	aktinidia ostrolistna ...	Actinidia arguta ...	K	aktinidia
3	01-01-1970	[null]	AMB.A	620100	ambrowiec ameryka...	Liquidambar styracif...	D	ambrowiec ...

Wybierz 10 wierszy z tabeli "gatunki" o najwyższych wartościach kolumny "species_num"

```
SELECT * FROM gatunki ORDER BY species_num DESC NULLS LAST LIMIT 10;
```

Data Output

	id integer	date_from character varying	date_to character varying	species_cd character varying	species_num integer	species_name character varying	latin_name character varying	shrubtree_fl character varying	bot_s... charac...
1	847	01-01-1970	[null]	DZW.BRO	2008800	dzwonek brodaty ...	Campanula barbata ...	K	dzwon
2	907	01-01-1970	[null]	SKA.ZWO	2008700	skalnica zwodnicza ...	Saxifraga sponhemi...	K	skalni
3	881	01-01-1970	[null]	PRZ.WCZ	2008600	przetacznik wczesny...	Veronica praecox All...	K	przeta

Dzień 2

Zaawansowane zapytania i przetwarzanie wyników

Grupowanie i operacje na zestawach danych

Grupowanie

Możliwości funkcji agregujących są znacznie większe, gdy zastosuje się klauzulę GROUP BY. Możliwe jest wówczas obliczanie statystyk nie tylko dla całej tabeli, ale też podziału tabeli na kategorie.

Reguła: wszystkie kolumny, które mają być zwrócone przez zapytanie, muszą być:

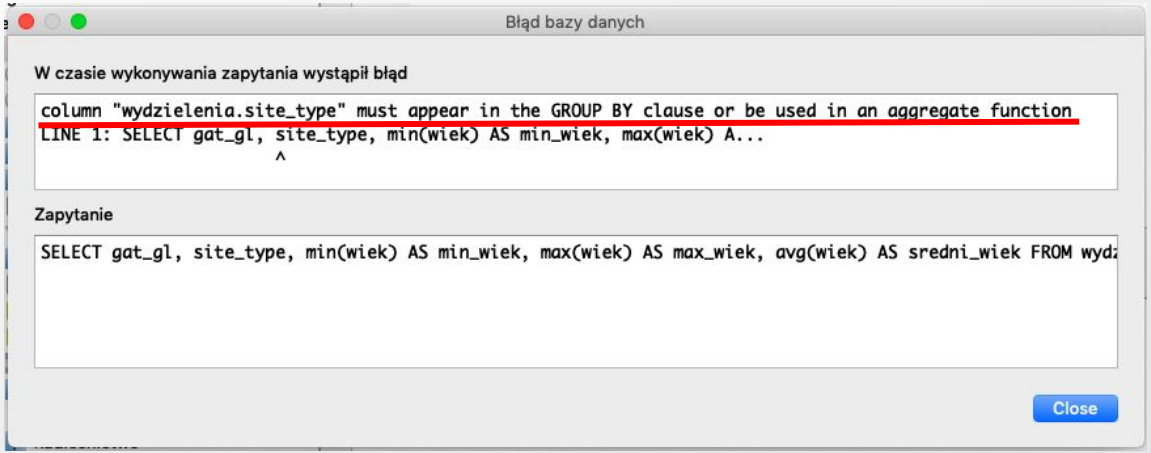
- użyte w funkcji agregującej lub
- użyte w klauzuli GROUP BY.

Przykłady **poprawnego** użycia:

```
SELECT species_cd, min(spec_age) AS min_wiek, max(spec_age) AS max_wiek, avg(spec_age) AS sredni_wiek
FROM wydzielenia
GROUP BY spec_agel;

SELECT count(*), fomatree_grp, wood_kind_fl FROM gatunki
GROUP BY fomatree_grp, wood_kind_fl;
```

Przykład **błędnego** zapytania:

```
SELECT gat_gl, site_type, min(spec_age) AS min_wiek, max(spec_age) AS max_wiek,
avg(spec_age) AS sredni_wiek
FROM wydzielenia
GROUP BY gat_gl;
```



Możliwe rozwiązania:

- wprowadzenie dodatkowego grupowania:

```
SELECT species_cd, site_type, min(spec_age) AS min_wiek, max(spec_age) AS max_wiek, avg(spec_age) AS sredni_wiek
FROM wydzielenia
GROUP BY gat_gl, site_type;
```

- wprowadzenie dodatkowej funkcji agregującej:

```
SELECT gat_gl, string_agg(site_type,','), min(spec_age) AS min_wiek, max(spec_age) AS max_wiek, avg(spec_age) AS sredni_wiek
FROM wydzielenia
GROUP BY gat_gl, site_type;
```

Jaka jest łączna powierzchnia (sub_area) wydzieleń dla każdego typu siedliskowego lasu (site_type)?

Ile jest wydzieleń w poszczególnych typach drzewostanu (area_type)?

Jaka jest łączna powierzchnia (sub_area) wydzieleń dla każdego typu siedliskowego lasu (site_type)?
SELECT site_type, sum(sub_area) FROM wydzielenia GROUP BY site_type;

	site_type character varying (9)	sum double precision
1	[null]	305.89
2	LMŚW	1848.34
3	LMWYŻW	13.65

Ile jest wydzieleń w poszczególnych typach powierzchni (area_type)?
SELECT area_type, count(*) FROM wydzielenia GROUP BY area_type;

	area_type character varying (11)	count bigint
1	PARKING L	1
2	B-R	3
3	L-CTWO	15

Operacje na zbiorach

PostgreSQL umożliwia wykonywanie operacji z zakresu algebry zbiorów: sumy (UNION), iloczynu (INTERSECT), różnicy (EXCEPT).

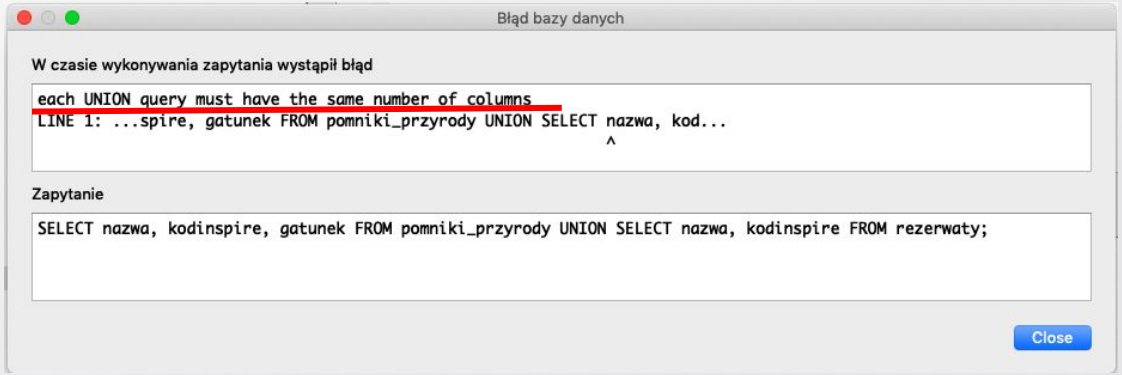
Do łączenia zbiorów służy operator UNION.

Suma zbiorów

UNION eliminuje powtarzające się wartości. UNION ALL zwraca wszystkie wyniki włącznie z powtarzającymi się.

Każde z zapytań wchodzących w skład UNION musi mieć taką samą liczbę i typ kolumn (nazwy mogą się różnić - do wyniku brana jest pod uwagę nazwa z pierwszego zbioru) - przykład **błędnego** zapytania:

```
SELECT nazwa, kodinspire, gatunek FROM pomniki_przyrody
UNION SELECT nazwa, kodinspire FROM rezerwaty;
```



Jeśli jedna z tabel nie zawiera jakiejś kolumny do budowy prawidłowego UNION, można podstawić w jej miejsce wartość stałą:

```
SELECT nazwa, kodinspire, gatunek FROM pomniki_przyrody
UNION SELECT nazwa, kodinspire, 'nie dotyczy' AS gatunek FROM rezerwaty;
```

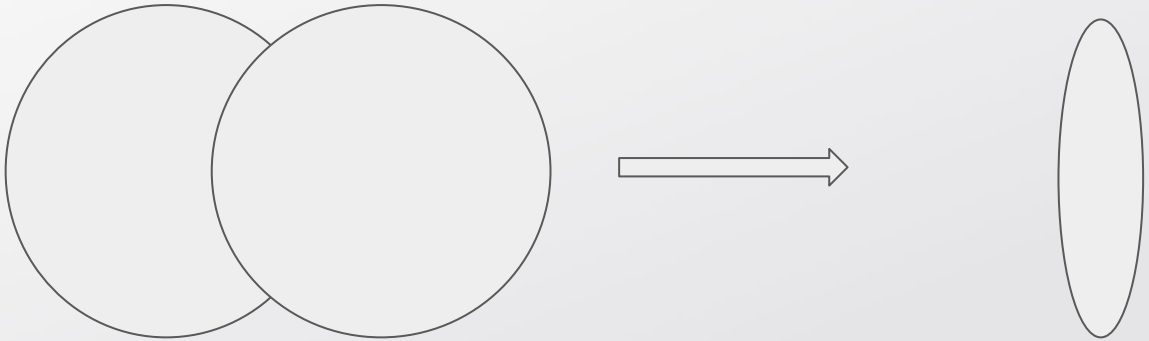
	nazwa character varying (254)	kodinspire character varying (254)	gatunek character varying
1	Anna i Arkadiusz	PL.ZIPOP.1393.PP.1609073...	Platan klonolistny - Platanus xacerifolia (Pl...
2	Annabrzeskie Wąwozy	PL.ZIPOP.1393.RP.1265	nie dotyczy
3	Antoniuk	PL.ZIPOP.1393.RP.1214	nie dotyczy

Iloczyn zbiorów

Do wyliczenia iloczynu zbiorów służy operator INTERSECT.

```
SELECT nazwa FROM soo
INTERSECT
SELECT nazwa FROM oso;
```

zwraca wartości występujące w obu zbiorach.

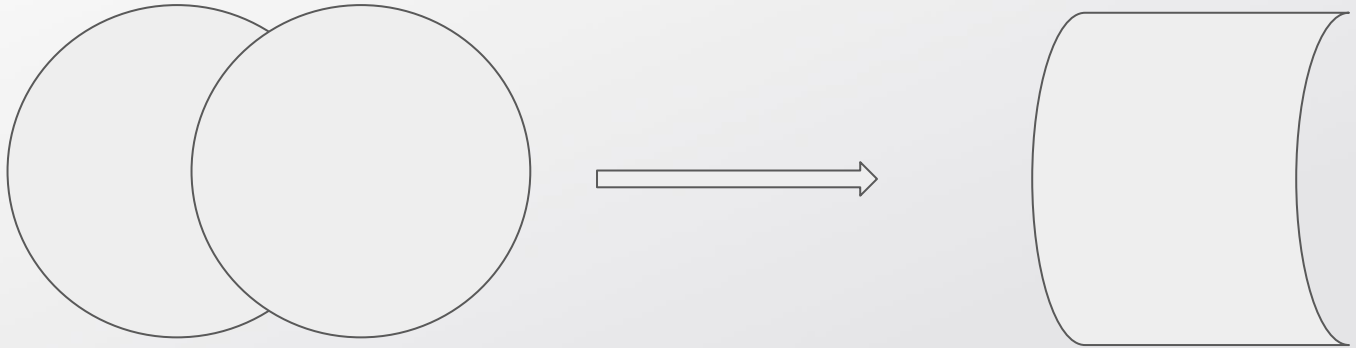


Różnica zbiorów

Do wyliczenia różnicy zbiorów służy operator EXCEPT.

```
SELECT nazwa FROM soo
EXCEPT
SELECT nazwa FROM oso;
```

zwraca wartości występujące w zbiorze pierwszym, ale nie w zbiorze drugim.



Jak otrzymać listę wszystkich kodów INSPIRE dla obszarów Natura 2000 i rezerwatów - tabele "soo", "oso" i "rezerwaty"?

Czy są wśród nich wartości powtarzające się?

Jakie kody typów siedliskowych ("site_type_cd") występują w tabeli "siedliska", a nie występują w kolumnie "site_type" tabeli "wydzielenia"

Odpowiedzi

Jak otrzymać listę wszystkich kodów INSPIRE dla obszarów Natura 2000 i rezerwatów - tabele "soo", "oso" i "rezerwaty"?

SELECT kodinspire FROM soo UNION SELECT kodinspire FROM oso UNION SELECT kodinspire FROM rezerwaty;

	kodinspire character varying (254)
1	PL.ZIPOP.1393.RP.1048
2	PL.ZIPOP.1393.RP.41
3	PL.ZIPOP.1393.RP.1408
4	PL.ZIPOP.1393.RP.25
5	PL.ZIPOP.1393.N2K.PLB22...

Czy są wśród nich wartości powtarzające się?

Nie występują. UNION i UNION ALL zwracają tyle samo wierszy.

Jakie kody typów siedliskowych ("site_type_cd") występują w tabeli "siedliska", a nie występują w kolumnie "site_type" tabeli "wydzielenia"
SELECT site_type_cd FROM siedliska EXCEPT SELECT site_type FROM wydzielenia;

	site_type_cd character varying
1	LWYŻ
2	LMB

Złączenia wewnętrzne i zewnętrzne

Złączenia tabel pozwalają na pobranie informacji z wielu tabel relacyjnej bazy danych, jeśli tylko posiadają one wspólną wartość na podstawie której można je połączyć.

Nie jest wymagane, aby relacja była utworzona na etapie projektowania bazy danych ani skonfigurowana przez administratora. Wystarczą identyczne wartości w kolumnach.

Złączenie domyślne

Złączenie domyślne wykorzystuje tylko klauzulę WHERE, bez użycia słowa kluczowego JOIN.

Przykład:

```
SELECT a.adr_for, a.species_cd, b.site_type_name
FROM wydzielenia a, siedliska b
WHERE a.site_type = b.site_type_cd;
```

	adr_for character varying (25)	species_cd character varying (9)	site_type_name character varying
1	02-23-1-04-149 -b -00	BK	Las wyżynny świeży
2	02-23-2-10-21 -a -00	SO	Las świeży
3	02-23-2-14-215 -d -00	BK	Las świeży
4	02-23-1-03-125 -g -00	DB	Las wyżynny świeży
5	02-23-1-07-84 -b -00	ŚW	Las górski świeży

Typ złączenia realizowany przez złączenie domyślne jest **złączeniem wewnętrznym**, oznacza to, że zwracane są tylko takie wyniki, które mają pasującą parę wartości w tabelach "a" i "b". Zapytanie można zapisać w następujący sposób:

```
SELECT a.adr_for, a.species_cs, b.site_type_name
FROM wydzielenia a INNER JOIN siedliska b
ON a.site_type = b.site_type_cd;
```

Warunek złączenia zapisywany jest za słowem kluczowym ON.

Złączenie zewnętrzne

Złączenie zewnętrzne pozwala na połączenie wielu tabel także wtedy, gdy nie zawsze wystąpią pasujące wiersze. Wówczas dla wiersza bez "pary" zostanie dopasowana wartość pusta (NULL).

Złączenie LEFT

W złączeniu zewnętrznym typu LEFT zwracane są wszystkie wiersze z pierwszej tabeli, jeśli nie ma odpowiedników w drugiej tabeli, zwracana jest wartość NULL; jeśli zostanie użyta funkcja agregująca np. count lub sum, wówczas zwróci zero.

```
SELECT a.site_type_cd, a.site_type_name, count(b.*)
FROM siedliska a LEFT OUTER JOIN wydzielenia b
ON a.site_type_cd = b.site_type
GROUP BY a.site_type_cd, a.site_type_name;
```

```
SELECT a.adr_for, a.species_cd, b.species_name FROM wydzielenia a
LEFT JOIN gatunki b
ON a.species_cd = b.species_cd;
```

Jaki będzie wynik, gdy słowo LEFT zamienimy na RIGHT?

Złączenie RIGHT

W złączeniu zewnętrznym typu RIGHT zwracane są wszystkie wiersze z pierwszej tabeli, jeśli nie ma odpowiedników w drugiej tabeli, zwracana jest wartość NULL.

```
SELECT a.adr_for, a.species_cd, b.species_name FROM wydzielenia a
RIGHT JOIN gatunki b
ON a.species_cd = b.species_cd;
```

	adr_for character varying (25)	species_cd character varying (9)	species_name character varying
1	02-23-1-06-194 -a -00	BRZ	brzoza brodawkowata
2	02-23-1-04-149 -b -00	BK	buk pospolity
3	02-23-2-10-21 -a -00	SO	sosna zwyczajna
4	02-23-2-14-215 -d -00	BK	buk pospolity
5	02-23-1-03-125 -g -00	DB	dąb nieokreślony

Podzapytania

Operatory ANY, ALL, DISTINCT

Zapytania SQL w systemie PostgreSQL mogą być zagnieżdżone.

Przykład:

Które wydzielenia mają powierzchnię powyżej średniej?

```
SELECT adr_for, sub_area
FROM wydzielenia
WHERE sub_area > (SELECT avg(sub_area) FROM wydzielenia)
ORDER BY sub_area ASC;
```

Operator IN

Operator IN sprawdza, czy wartość znajduje się w pewnym zbiorze.
Zbiór może być zadany przez użytkownika jako konkretne wartości lub jako podzapytanie.

```
SELECT adr_for, sub_area FROM wydzielenia
WHERE site_type IN ('BMŚW', 'BMW', 'BMB');

SELECT * FROM gatunki
WHERE species_cd IN (SELECT species_cd FROM wydzielenia);
```

Operator ANY

Alternatywną formą porównania do pewnego zbioru wartości jest zastosowanie operatora ANY.

```
SELECT * FROM gatunki
WHERE species_cd = ANY (SELECT species_cd FROM wydzielenia);

SELECT adr_for, sub_area FROM wydzielenia
WHERE site_type = ANY (ARRAY['BMŚW', 'BMW', 'BMB']);
```

Operator ALL

Operator ALL dokonuje porównania z wszystkimi wartościami z zadanego zbioru.

Składnia:

wartość <operator porównania> ALL (podzapytanie)

Przykład:

```
SELECT adr_for, sub_area, spec_age, species_cd FROM wydzielenia
WHERE spec_age >=  ALL (SELECT spec_age FROM wydzielenia WHERE
species_cd = 'BK');
```

Modyfikator DISTINCT

Modyfikator DISTINCT służy do zwrócenia tylko niepowtarzalnych wartości.

Przykład:

```
SELECT species_cd FROM wydzielenia;

SELECT DISTINCT species_cd FROM wydzielenia;
```

	species_cd character varying (9)
1	OL.S
2	ŚL.T
3	SO.WE
4	[null]
5	JD
6	DG
7	TP
8	BRZ
9	SO

Modyfikator DISTINCT może zostać zastosowany również wspólnie z funkcjami agregującymi, np.

SELECT count(DISTINCT species_cd), count(species_cd) FROM wydzielenia;

	count bigint	count bigint
1	34	3638

DISTINCT ON

Baza PostgreSQL dysponuje również modyfikatorem DISTINCT ON, który nie należy do standardu SQL.

Przykład zastosowania: zwrócenie pełnych informacji o największym/najmniejszym wierszu w każdej kategorii

```
SELECT DISTINCT ON (site_type) * FROM wydzielenia
ORDER BY site_type, spec_age DESC;
```

Aby DISTINCT ON miał sens, powinien być użyty z klauzulą ORDER BY. Klauzula musi zawierać 2 kolumny - jako pierwsza kolumna w której znajdują się kategorie, jako druga kolumna w której znajdują się wartości.

Modyfikacja i usuwanie danych

Wstawianie danych do tabel i widoków

Do modyfikacji danych służy komenda UPDATE.

Podstawowa składnia zapytania UPDATE jest następująca:

UPDATE <tabela> SET <kolumna> = <wartość>

Przykład:

UPDATE gatunki SET height_grp = trim(trailing ' ' from height_grp);

- usunięcie zbędnych spacji na końcu tekstu. Zabieg obowiązkowy dla danych z Banku Danych o Lasach

UPDATE w tej formie zmienia wartość we WSZYSTKICH wierszach tabeli!

Możliwa jest modyfikacja danych w wielu kolumnach na raz, np.

```
UPDATE gatunki SET fomatree_grp = trim(trailing ' ' from fomatree_grp),
volume_grp = trim(trailing ' ' from volume_grp),
worth_grp = trim(trailing ' ' from worth_grp);
```

Oprócz UPDATE dla całej tabeli, za pomocą SQL można modyfikować pojedyncze wiersze:

```
UPDATE gatunki SET bot_name = 'Agrest pospolity' WHERE id = 30;
```

lub ich grupy według dowolnej klauzuli WHERE:

```
UPDATE gatunki SET height_grp = gat_grp WHERE height_grp IS NULL;
```

Aby móc edytować dane za pomocą narzędzi graficznych takich jak pgAdmin lub QGIS, tabela musi mieć zdefiniowany **klucz główny**.

Dane importowane z zewnętrznych źródeł za pomocą QGIS czy ogr2ogr zwykle spełniają ten warunek.

Jeśli tabela nie posiada klucza głównego, ale posiada kolumnę z niepowtarzalnym identyfikatorem, należy wykonać:

```
ALTER TABLE tabela ADD CONSTRAINT tabela_pk PRIMARY KEY(identyfikator);
```

Jeśli nie ma takiej kolumny, można wygenerować identyfikatory automatycznie:

```
ALTER TABLE tabela ADD COLUMN identyfikator SERIAL PRIMARY KEY;
```

Do usuwania danych służy komenda DELETE.

Komenda DELETE bez podania klauzuli WHERE usunie **WSZYSTKIE** wiersze w tabeli!

```
DELETE FROM cdma;
```

zwykle jednak usuwa się pojedyncze wiersze:

```
DELETE FROM wydzielenia WHERE adr_for = '02-23-2-13-188    -j    -00';
```

lub ich grupy:

```
DELETE FROM siedliska WHERE date_to::date > now();
```

W praktyce do czyszczenia całej tabeli, wykorzystuje się komendę TRUNCATE TABLE:

```
TRUNCATE TABLE punkty_pomiarowe;
```

gdyż jest szybsza od DELETE bez WHERE.

Do usuwania danych w programach pgAdmin i QGIS stosuje się te same zasady, co do modyfikacji: wymagany jest zadeklarowany klucz główny.

Wstawianie danych

Do wstawiania danych służy komenda INSERT.

Podstawowa składnia:

INSERT INTO tabela VALUES ('wartosc1','wartosc2','wartosc3');

w takiej sytuacji muszą być podane wartości dla wszystkich kolumn w tabeli.

Możliwe jest też podanie podzbioru kolumn:

```
INSERT INTO tabela(kolumna1, kolumna2)
VALUES('wartosc1','wartosc1');
```

Przykłady:

```
INSERT INTO siedliska VALUES(-1,current_date, NULL, 'BM', 'Bór mokry', 99, 5, 'BB');

INSERT INTO gatunki(date_from, species_cd, species_name, latin_name)
VALUES(current_date, 'CHI', 'Choinka zwyczajna', 'Choinkus vulgaris');
```

Tworzenie tabel i import danych

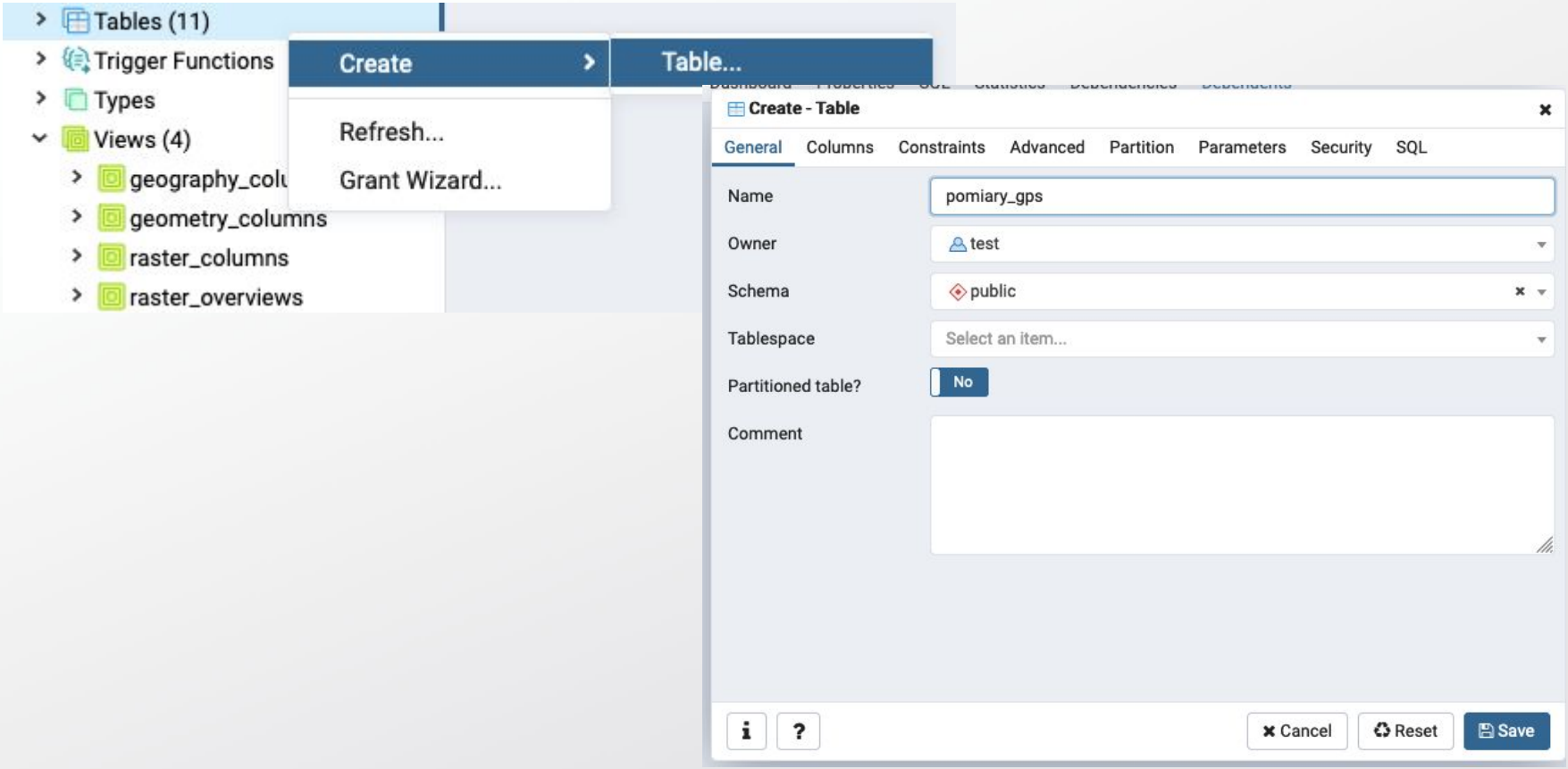
Tworzenie nowych tabel

SQL umożliwia tworzenie tabel z poziomu zapytania. Przykładowa definicja tabeli:

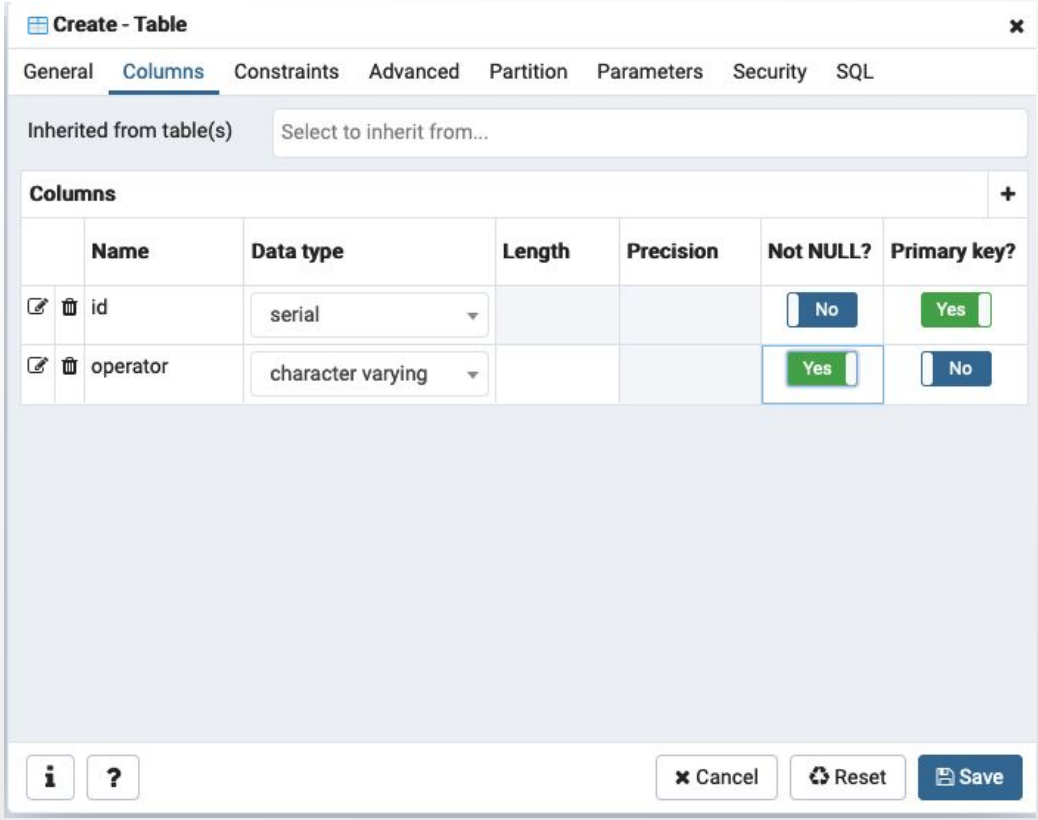
```
CREATE TABLE powierzchnie_kontrolne(

id SERIAL PRIMARY KEY,
numer INTEGER,
nazwa VARCHAR,
uwagi VARCHAR,
geom GEOMETRY(Polygon,2180)

);
```

Tworzenie nowych tabel

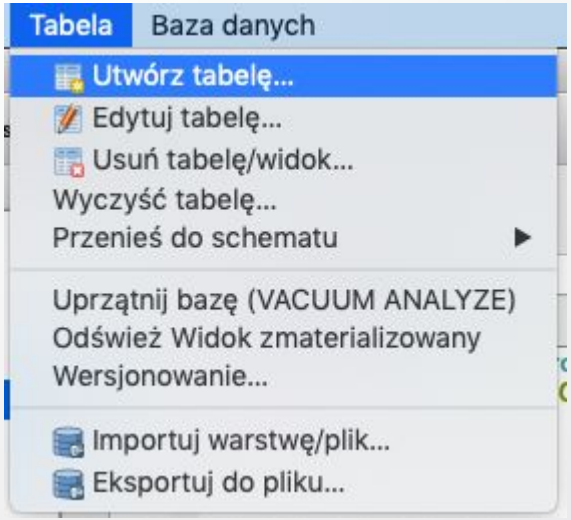
Tabelę można również utworzyć z pomocą narzędzi graficznych. Przykład - pgAdmin:

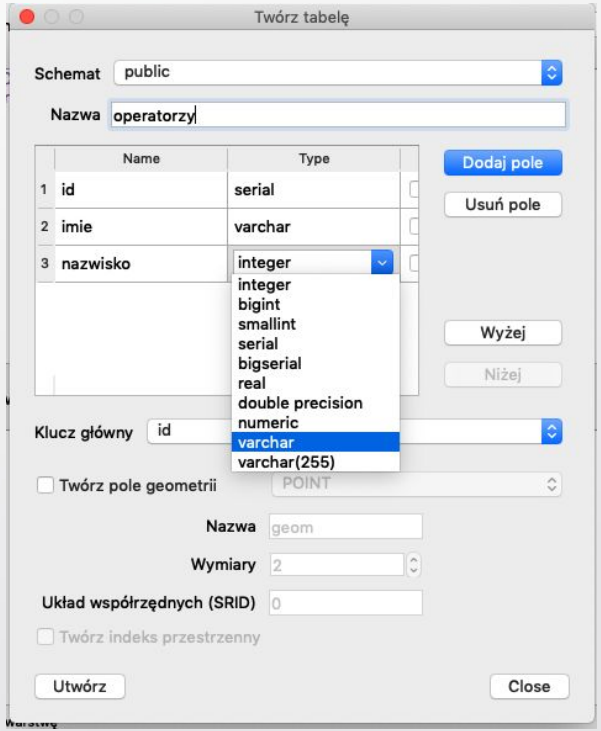


Tworzenie nowych tabel



Można również utworzyć tabelę z pomocą QGIS:





W początkowej części szkolenia został zaprezentowany import danych do bazy z użyciem QGIS. Oprócz tego, dostępne są jeszcze 2 narzędzia do importu istniejących danych do bazy PostGIS, uruchamiane z linii komend:

shp2pgsql, narzędzie przeznaczone wyłącznie do plików SHP, instalowane razem z PostGIS

ogr2ogr, narzędzie przeznaczone do różnego typu danych przestrzennych (SHP, GML, GPKG, GeoJSON, KML…) instalowane przez OSGeo4W - pakiet gdal-bin

Import shp2pgsql

```
shp2pgsql -D -s 2180 -I gminy.shp public.gminy > gminy.sql
psql -d szkolenie -f gminy.sql
```

Opcje:

-D używa masowego importu zamiast INSERT
-s 2180 ustawia układ współrzędnych
-I tworzy indeks przestrzenny

```
ogr2ogr -f PostgreSQL -lco GEOMETRY_NAME=geom -lco FID=id
  "PG:dbname=szkolenie host=localhost user=kursant password=postgis"
PL.PZGIK.201.16__OT_ADMS_P.xml
```

Domyślna nazwa klucza głównego to "ogc_fid" - zmiana przez -lco FID=id

Domyślna nazwa kolumny geometrii to "wkb_geometry" - zmiana przez -lco GEOMETRY_NAME=geom

```
ogr2ogr -f PostgreSQL -lco GEOMETRY_NAME=geom -lco FID=id
  "PG:dbname=szkolenie host=localhost user=kursant password=postgis"
ParkiNarodowe.shp
```

Błędy:

Występowanie geometrii typu MULTI:
ERROR: Geometry type (Polygon) does not match column type (MultiPolygon)

-nlt PROMOTE_TO_MULTI

Występowanie pól numerycznych:
DETAIL: A field with precision 24, scale 15 must round to an absolute value less than 10^9.

-lco precision=NO

Polskie znaki w WIN1250:

ERROR: Invalid byte sequence for encoding UTF-8

Windows: SET PGCLIENTENCODING=WIN1250
Linux/Mac: export PGCLIENTENCODING=WIN1250

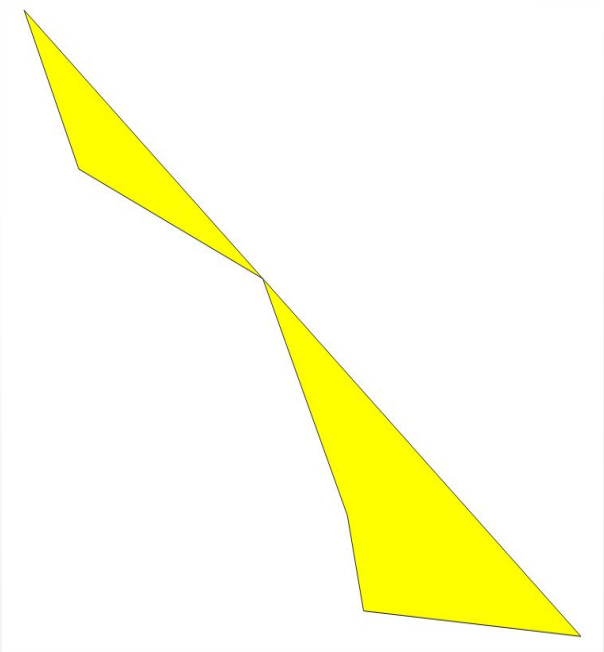
Kryteria poprawności geometrii

Ocena poprawności geometrii i metody naprawy niepoprawnych obiektów

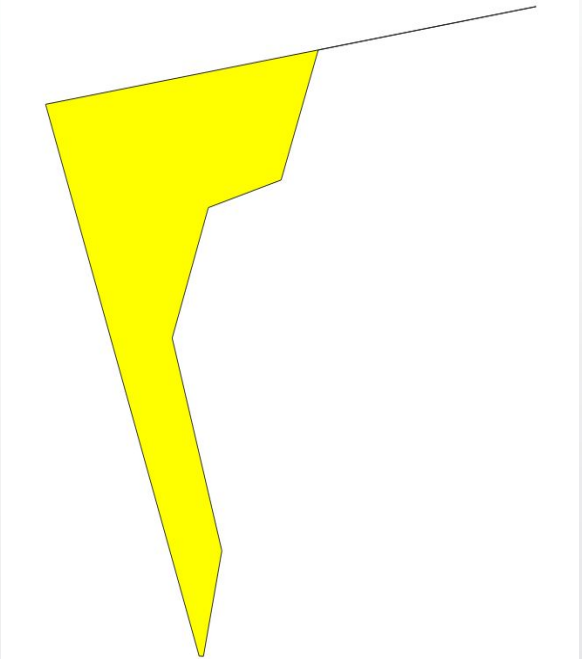
- Geometrie biorące udział w analizach przestrzennych muszą spełniać kryteria poprawności
- Próba wykonania analizy na niepoprawnej geometrii kończy się błędem:

```
NOTICE:  TopologyException: found non-noded intersection between LINESTRING (coords
edited) and LINESTRING (coords edited) at (position edited)
ERROR:  GEOS Intersection() threw an error!

********** Error **********

ERROR: GEOS Intersection() threw an error!
SQL state: XX000
```

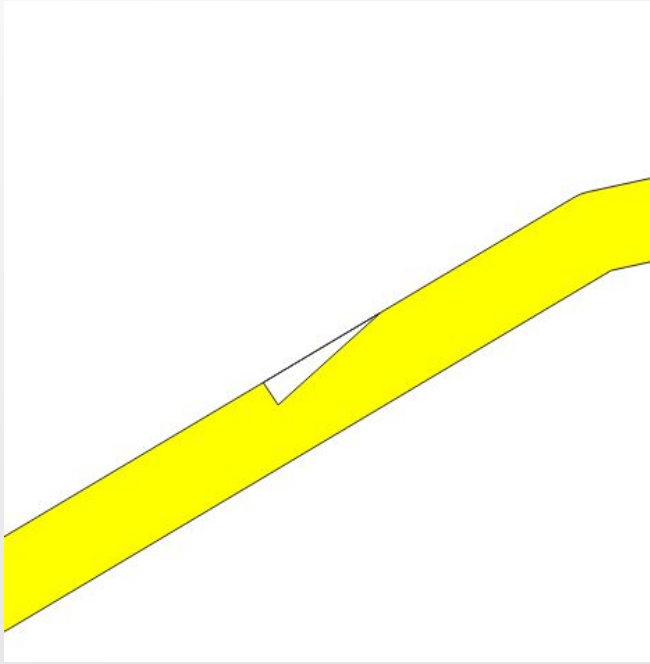
Kryteria poprawności



Samoprzecięcie



Bagnet



Enklawa (inner ring)
w granicy

ST_MakeValid - stara się nie zmieniać kształtu geometrii (np. zamiana Polygon na 2-częściowy MultiPolygon)

Może tworzyć geometrie typu GeometryCollection, które nie są obsługiwane przez QGIS!

ST_Buffer - zmienia kształt geometrii, tworzy zawsze geometrie typu Polygon

Do sprawdzania geometrii można także użyć narzędzia "Sprawdź poprawność" w QGIS.

```
SELECT *, ST_IsValidReason(geom) FROM oso
WHERE ST_IsValid(geom) = FALSE;
```

	id	geom	gid	nazwa	kod	kodinspire	st_isvalidreason
1	123	010600002...	45560	Grądy Odrza...	PLB020002	PL.ZIPOP.13...	Self-intersection[369087.243042136 359393.061863236]

```
SELECT id, ST_MakeValid(geom) FROM oso
WHERE ST_IsValid(geom) = FALSE;
```

```
SELECT id, ST_Buffer(geometry, 0) FROM oso
WHERE ST_IsValid(geom) = FALSE;
```

Układy współrzędnych w bazie danych

Układy współrzędnych

- **SRID** - identyfikator liczbowy - kod EPSG
- Definicje w tabeli **spatial_ref_sys**
- odczyt SRID geometrii: **SELECT** ST_SRID(geom)
- przypisanie SRID do geometrii, jeśli go brak lub jest błędny: ST_SetSRID
- przeliczenie (transformacja): ST_Transform(geom,SRID)

- Przykład sprawdzenia układu współrzędnych - SELECT DISTINCT ST_SRID(geom) FROM oso;

- Przykład transformacji - SELECT nazwa,gatunek, ST_Transform(geom,4326) FROM pomniki_przyrody;
-
- Transformacja geometrii do układu 4326 jest wymagana w pgAdmin do wyświetlenia danych na podkładzie OpenStreetMap.

Dziękuję za uwagę