

# Bazy danych — Lekcja 1

## Tworzenie baz danych i aktualizacja struktury tabel

Paweł Staniszewski

Logintegra

WSB — Programista Java 2020

Paweł Staniszewski – [pawel.staniszewski@logintegra.com](mailto:pawel.staniszewski@logintegra.com)

- ▶ Skończyłem matematykę na PG
- ▶ Przez kilka lat pracowałem jako programista aplikacji webowych – `Grails`, `Ember.js` / `Vue.js`, `SQL`
- ▶ Od ponad roku zajmuję się głównie zarządzaniem zespołem oraz naszymi projektami
- ▶ Interesują mnie zagadnienia Dev-Ops – w tym automatyzacja i usprawnianie procesu wytwarzania oprogramowania
- ▶ Czasem wykonuję analizy logistyczne, najczęściej w `R`
- ▶ Będę opiekunem projektu końcowego

Wiem, że to nie jest przyjemne, ale chętnie poznam odpowiedzi na następujące pytania.

- ▶ Jakiego mam doświadczenie w programowaniu?
- ▶ Jakiego mam doświadczenie w pracy z bazami danych?
- ▶ W jakim sektorze pracuję?
- ▶ Jakiego systemu operacyjnego używam (Windows, macOS, Linux)?
- ▶ Jakie są moje oczekiwania co do tego przedmiotu?

# Czego chciałbym nauczyć

- ▶ Jak zainstalować oraz skonfigurować bazę w nowym projekcie
- ▶ Jak pracować z kopiami zapasowymi baz
- ▶ Podstaw SQL (zapytania, indeksy, procedury)
- ▶ Na co zwracać uwagę podczas modelowania danych;
- ▶ Myślenia o wydajności zapytań – także podczas pracy z kodem Java;
- ▶ Przywiązywania wagi do odpowiedniego nazewnictwa, stylu itp.

- ▶ Musimy robić *regularne* przerwy, żeby wietrzyć salę
- ▶ Możemy zrobić jedną dłuższą – np. 30-minutową – przerwę, jeśli jest taka potrzeba
- ▶ Wszelkie pytania mile widziane
- ▶ Proszę mówić, jeśli zacznie być zbyt nudno lub trudno
- ▶ Jako materiały/narzędzia będę starał się podawać głównie te darmowe
- ▶ Zaliczenie
  - ▶ Nie ma egzaminu z przedmiotu – ale pytania z niego znajdą się na egzaminie semestralnym
  - ▶ Będzie do zrobienia projekt, ale szczegóły omówimy na kolejnym spotkaniu

# Co to jest baza danych

## Baza danych – ogólna definicja

Baza danych – zbiór danych zapisanych zgodnie z określonymi regułami

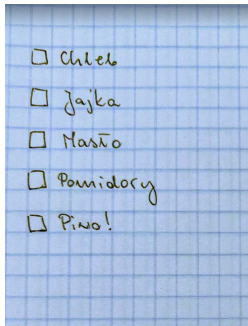
[https://pl.wikipedia.org/wiki/Baza\\_danych](https://pl.wikipedia.org/wiki/Baza_danych)

## Baza danych – bardzo specyficzna definicja

Baza danych oznacza 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

Obwieszczenie Marszałka Sejmu Rzeczypospolitej Polskiej z dnia 18 października 2019 r. w sprawie ogłoszenia jednolitego tekstu ustawy o ochronie baz danych (<http://isap.sejm.gov.pl/isap.nsf/DocDetails.xsp?id=WDU20190002134>)

# Bazy danych – przykłady

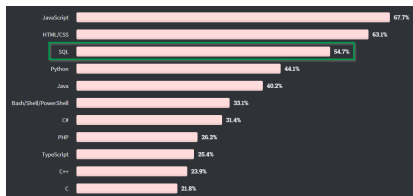


	A	B	C	D
1	<b>Tytuł</b>	<b>Autor</b>	<b>Rok wydania</b>	<b>Język</b>
2	Harry Potter i Kamień Filozoficzny	J.K. Rowling	1997	Angielski
3	Duma i uprzedzenie	J. Austen	1813	Angielski
4	Zbrodnia i kara	F. Dostojewski	1867	Rosyjski
5	Bracia Karamazow	F. Dostojewski	1880	Rosyjski
6	Dziady	A. Mickiewicz	1823	Polski
7	Morderstwo w Orient Expressie	A. Christie	1934	Angielski

Rysunek: Baza danych w formie elektronicznej

Rysunek: Technicznie  
jest to baza danych

# Dlaczego warto uczyć się baz danych – popularność SQL



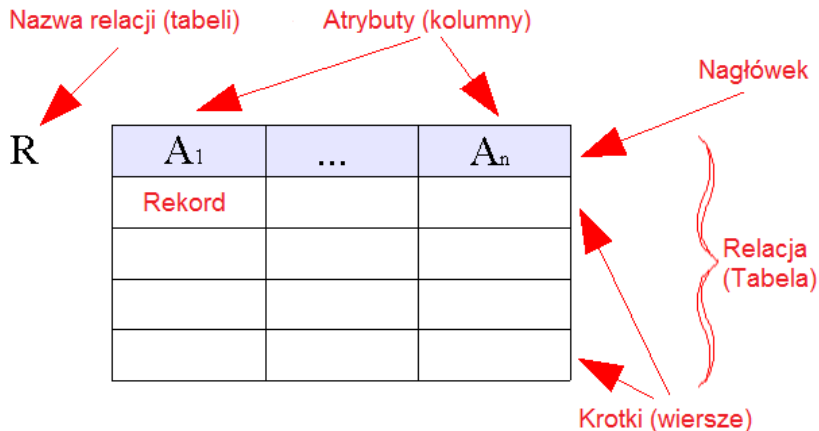
Rysunek: Stack Overflow – najpopularniejsze technologie



Rysunek: Stack Overflow – najlepiej opłacane technologie

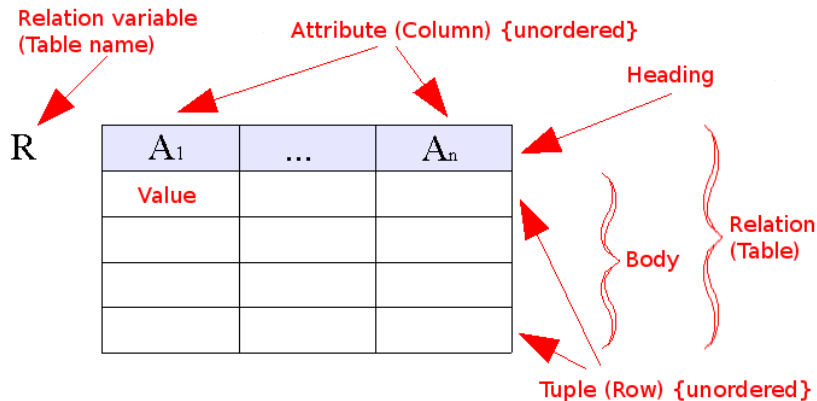


# Relacyjne bazy danych



Rysunek: Model relacyjny – pojęcia

# Tabele w relacyjnych bazach danych



Rysunek: Model relacyjny

## SQL

SQL (Structured Query Language) – język programowania służący do manipulacji danymi oraz zarządzania relacyjnymi bazami danych

```
SELECT id,  
       username,  
       email  
FROM person  
WHERE username = 'pawel';
```

# Systemy zarządzania bazami danych (DBMS)

## DBMS

System zarządzania bazami danych (*Database management system, DBMS*) – system oprogramowania, który umożliwia użytkownikom definiowanie, tworzenie, utrzymywanie i kontrolowanie dostępu do bazy danych

*RDBMS* – system zarządzania bazami relacyjnymi

Najpopularniejsze systemy zarządzania relacyjnymi bazami danych:

- ▶ MySQL (open-source)
- ▶ PostgreSQL (open-source) ← tego będziemy używać
- ▶ MSSQL (Microsoft)
- ▶ Oracle



# Instalacja PostgreSQL

PostgreSQL zainstalować można na wszystkich najpopularniejszych systemach

## Packages and Installers

Select your operating system family:



Select your Linux distribution:



Rysunek: <https://www.postgresql.org/download/>

# Instalacja PostgreSQL – Linux

Instalację wykonujemy zgodnie z [oficjalną instrukcją](#)

```
# Dodajemy oficjalne repozytorium PostgreSQL — aby mieć możliwość instalacji najnowszej wersji
sudo sh -c 'echo "deb http://apt.postgresql.org/pub/repos/apt $(lsb_release -cs)-pgdg main" >
/etc/apt/sources.list.d/pgdg.list'

# Pobieramy i instalujemy klucz dla repozytorium, aby Ubuntu uznało je za zaufane
wget --quiet -O - https://www.postgresql.org/media/keys/ACCC4CF8.asc | sudo apt-key add -

# Odswiezamy baze danych bibliotek
sudo apt-get update

# W koncu instalujemy najnowsza wersje PostgreSQL
sudo apt-get install postgresql-13
```

Na koniec możemy sprawdzić, czy na pewno mamy dobrą wersję

```
$ psql --version

# Chcemy tu dostać "13.0"
psql (PostgreSQL) 13.0 (Ubuntu 13.0-1.pgdg20.04+1)
```

# Konfiguracja PostgreSQL – Linux

Tak od razu nie będziemy mogli za wiele zrobić

---

```
$ psql
```

```
psql: error: could not connect to server: FATAL: role  
"pawel_staniszewski" does not exist
```

---



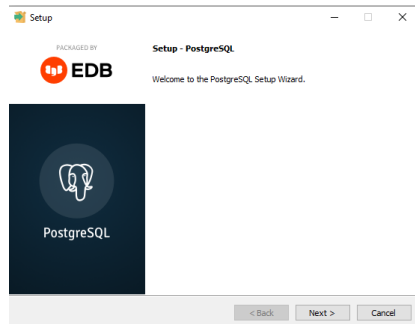


# Instalacja PostgreSQL – Windows/macOS – Graficzny instalator

Dla systemów Windows i Mac dostępny jest graficzny instalator, który znacznie ułatwia konfigurację PostgreSQL

## Wskazówka

Dobra instrukcja obsługi instalatora znajduje się na stronie [enterprisedb.com](https://enterprisedb.com)



Rysunek: Graficzny instalator dla Windows/macOS

# Instalacja PostgreSQL – Windows/macOS

[postgresql.org/download](https://postgresql.org/download)

Windows installers 

Interactive installer by EDB

Pobieramy aplikację,  
która zainstaluje nam  
PostgreSQL

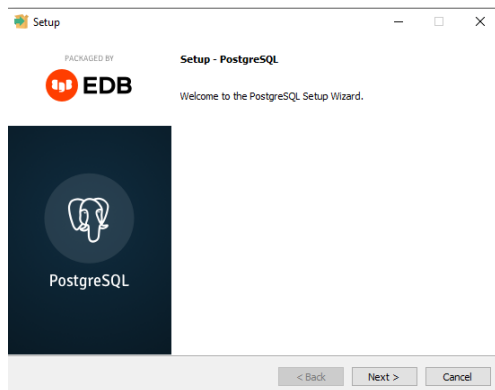
Download the **installer** certified by EDB for all supported PostgreSQL versions.

Rysunek: <https://www.postgresql.org/download/windows/>



# Instalacja PostgreSQL – Windows/macOS – Pierwsze kroki

Uruchamiamy pobrany plik i klikamy *Next*

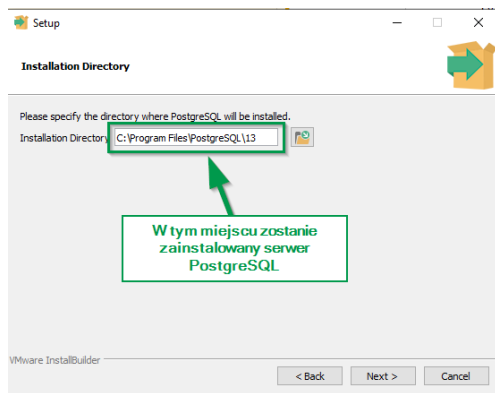


**Rysunek:** Po uruchomieniu instalki nie mamy zbyt wielu opcji

# Instalacja PostgreSQL – Windows/macOS – Lokalizacja instalacji

Ustawiamy ścieżkę, w której zostanie zainstalowany serwer PostgreSQL oraz wszystkie biblioteki i komponenty.

Możemy zostawić domyślny wybór.

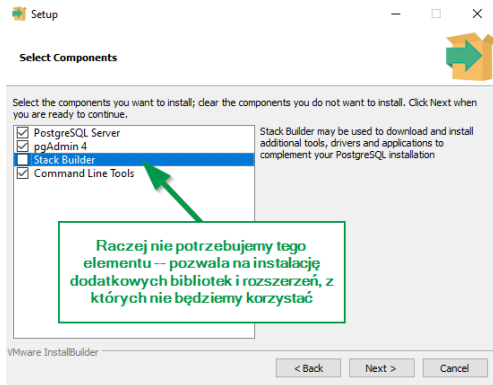


Rysunek: Ustalamy lokalizację instalacji

# Instalacja PostgreSQL – Windows/macOS – Wybór komponentów

Wybieramy komponenty, które chcemy zainstalować.

Polecam zaznaczyć wszystko oprócz *Stack Builder*



**Rysunek:** Wybieramy elementy, które chcemy zainstalować

# Instalacja PostgreSQL – Windows/macOS – Opis komponentów

Opis komponentów z poprzedniego kroku

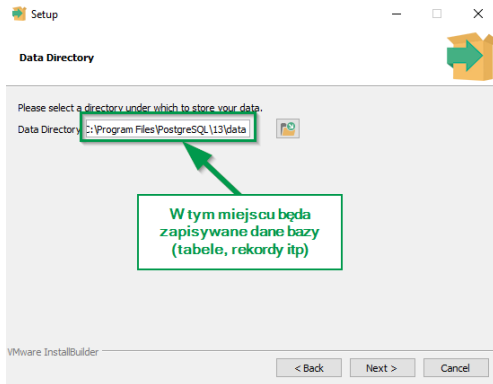
- ▶ **PostgreSQL Server** – serwer bazy, niezbędny
- ▶ **pgAdmin 4** – graficzne narzędzie do zarządzania bazą, przydatne (opowiemy o nim później)
- ▶ **Stack Builder** – program do instalacji rozszerzeń PostgreSQL, nam się nie przyda
- ▶ **Command Line Tools** – narzędzia wiersza poleceń jak `psql` czy `createdb` – przydatne



# Instalacja PostgreSQL – Windows/macOS – Lokalizacja danych

Ustawiamy ścieżkę, w której będą zapisywane dane bazy – wszystko, co zapiszemy w tabelach.

Możemy zostawić domyślny wybór.

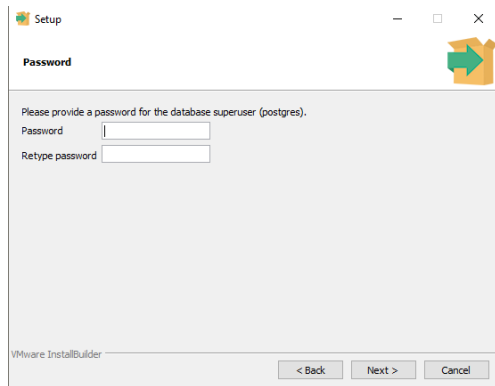


Rysunek: Ustalamy lokalizację plików bazy

# Instalacja PostgreSQL – Windows/macOS – Hasło dla użytkownika postgres

Ustawiamy hasło dla użytkownika **postgres**, posiadającego uprawnienia superadmina.

Zapamiętajmy to hasło!  
Będzie potrzebne do administracji bazą.

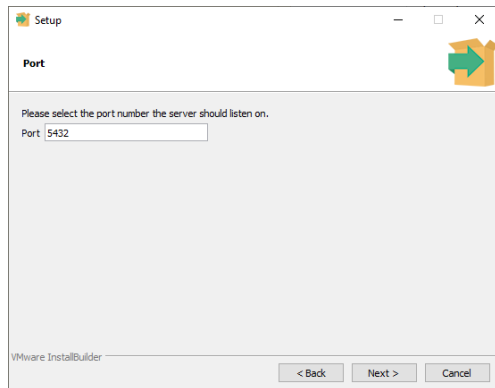


**Rysunek:** Ustawiamy hasło dla użytkownika **postgres**

# Instalacja PostgreSQL – Windows/macOS – Wybór portu

Wybieramy *port*, na którym nasłuchiwać będzie nasz serwer bazy.

Jeżeli instalujemy PostgreSQL za pierwszym razem, możemy zostawić domyślny wybór – **5432**.

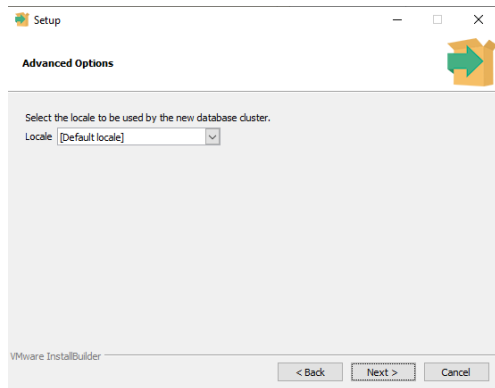


Rysunek: Ustawiamy port

# Instalacja PostgreSQL – Windows/macOS – Język

Ustawiamy język (*locale*)

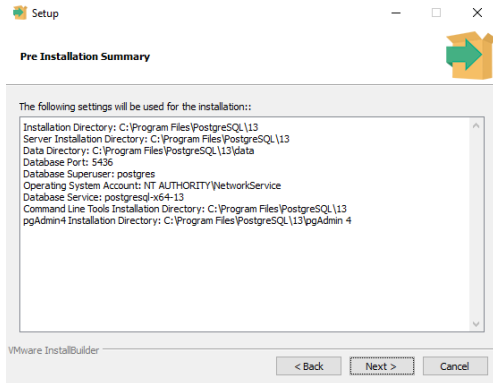
Możemy zostawić domyślny wybór



Rysunek: Wybieramy język

# Instalacja PostgreSQL – Windows/macOS – Podsumowanie

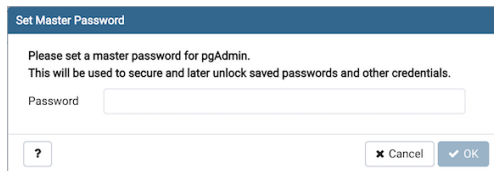
W kolejnym kroku zobaczymy podsumowanie wszystkich naszych wyborów



Rysunek: Podsumowanie

**pgAdmin** jest graficznym narzędziem do zarządzania i pracy z PostgreSQL

Przy pierwszym uruchomieniu zostaniemy poproszeni o ustawienie hasła



Set Master Password

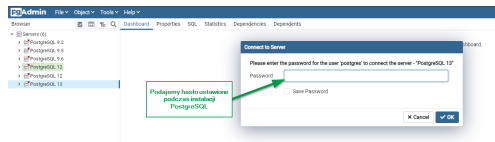
Please set a master password for pgAdmin.  
This will be used to secure and later unlock saved passwords and other credentials.

Password

? Cancel OK

Rysunek: Pierwsze uruchomienie pgAdmin

Żeby połączyć się z serwerem PostgreSQL rozwijamy odpowiednią gałąź w drzewku po lewej i podajemy hasło ustawione podczas instalacji PostgreSQL – czyli hasło użytkownika *postgres*

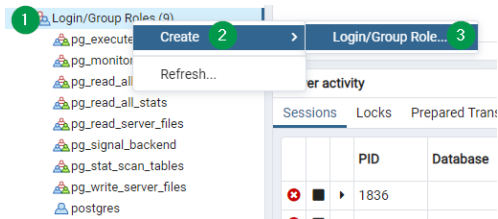


Rysunek: Łączenie z serwerem

# pgAdmin – Tworzenie użytkowników

Żeby utworzyć użytkownika

- 1 Klikamy prawym przyciskiem myszy na *Login/Group Roles*
- 2 Następnie wybieramy *Create*
- 3 Ostatecznie wybieramy *Login/Group Role ...*



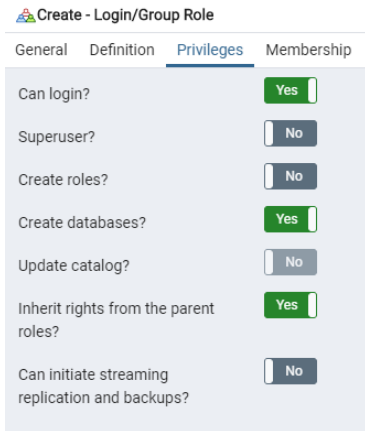
Rysunek: Tworzenie użytkownika



# pgAdmin – Tworzenie użytkowników

Podczas tworzenia użytkownika powinniśmy w zakładce *Privileges* wybrać kilka opcji:

- ▶ *Can login?* – odznaczenie tej opcji sprawi, że utworzymy *rolę* a nie *użytkownika*
- ▶ *Create databases?* – chcemy móc tworzyć nowe bazy
- ▶ *Inherit rights from the parent roles?* – to jest domyślnie, więc zostawiamy



The screenshot shows the 'Create - Login/Group Role' dialog box in pgAdmin, with the 'Privileges' tab selected. The dialog has four tabs: 'General', 'Definition', 'Privileges', and 'Membership'. The 'Privileges' tab contains a list of permissions with corresponding 'Yes' or 'No' buttons. The 'Can login?' button is highlighted in green, indicating it is selected. The 'Superuser?' button is highlighted in grey, indicating it is not selected. The 'Create roles?' button is highlighted in grey, indicating it is not selected. The 'Create databases?' button is highlighted in green, indicating it is selected. The 'Update catalog?' button is highlighted in grey, indicating it is not selected. The 'Inherit rights from the parent roles?' button is highlighted in green, indicating it is selected. The 'Can initiate streaming replication and backups?' button is highlighted in grey, indicating it is not selected.

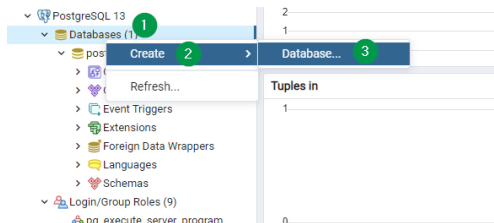
Permission	Selected
Can login?	Yes
Superuser?	No
Create roles?	No
Create databases?	Yes
Update catalog?	No
Inherit rights from the parent roles?	Yes
Can initiate streaming replication and backups?	No

Rysunek: Uprawnienia użytkownika

# pgAdmin – Tworzenie bazy

Żeby utworzyć bazę

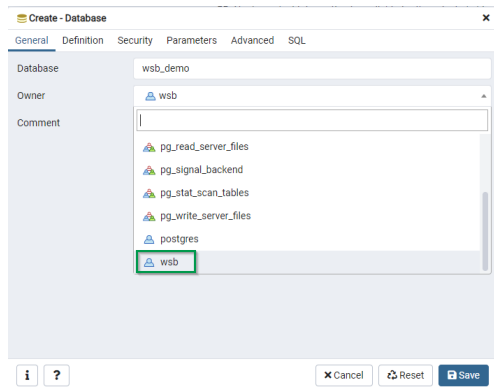
- 1 Klikamy prawym przyciskiem myszy na *Databases*
- 2 Następnie wybieramy *Create*
- 3 Ostatecznie wybieramy *Database ...*



Rysunek: Tworzenie bazy

# pgAdmin – Tworzenie bazy c.d.

Podczas tworzenia bazy musimy tylko pamiętać o nadaniu czytelnej nazwy oraz wyborze właściciela – np. utworzonego przez nas użytkownika

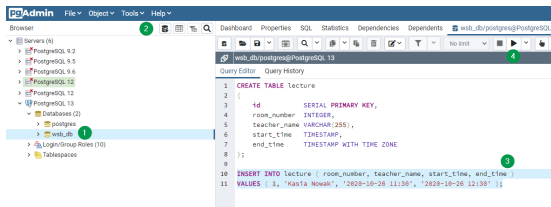


Rysunek: Tworzenie bazy

# pgAdmin – Wykonywanie zapytań SQL

**pgAdmin** pozwala nam także wykonywać zapytania SQL

- 1 Zaznaczamy w drzewku naszą bazę
- 2 Klikamy w ikonę *Query Tool*
- 3 W okienku *Query Editor* piszemy nasze zapytania
- 4 Zaznaczamy kod, który chcemy wykonać, i klikamy trójkącik lub naciskamy klawisz *F5*



Rysunek: Wykonywanie zapytań

# Integracja z Visual Studio Code

Podczas pracy nad większymi projektami – zwłaszcza takimi, które bazują na wielu technologiach – wygodnie jest użyć IDE.

Do pracy z PostgreSQL możemy użyć

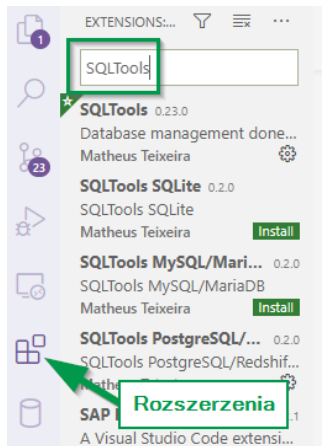
- ▶ [Visual Studio Code](#) – darmowy
- ▶ [IntelliJ Ultimate](#)
- ▶ [DataGrip](#)
- ▶ Dowolnego narzędzia JetBrains z dodatkiem [database-navigator](#) – darmowy

# Integracja z Visual Studio Code

Integrację z PostgreSQL pokażemy na przykładzie darmowego **Visual Studio Code**.

Pierwszym krokiem jest instalacja rozszerzeń

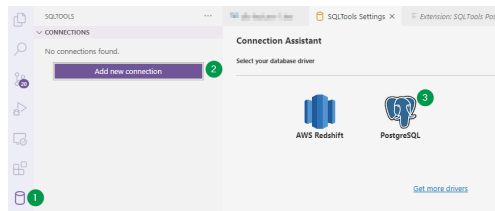
- ▶ *SQLTools*
- ▶ *SQLTools PostgreSQL/Redshift Driver*



Rysunek: Rozszerzenie *SQLTools*

# Integracja z Visual Studio Code c.d.

- 1 Wybieramy rozszerzenie *SQLTools*
- 2 *Add new connection*
- 3 Wybieramy *PostgreSQL*



Rysunek: Rozszerzenie *SQLTools*

# Integracja z Visual Studio Code c.d.

W ostatnim kroku podajemy dane ustawione podczas instalacji PostgreSQL oraz tworzenia użytkownika i bazy.

Nie powinno tu być problemów – o ile tylko wszystko zapamiętamy.

Connection Assistant < Step 2/3

Connection Settings

Connection name\* wsb-example

Connection group

Connect using\* Server and Port

Server Address\* localhost

Port\* 5436

Database\* wsb-db

Username\* wsb

Use password Save password

Password\* .....

node-pg driver specific options

You can get more info here: <https://node-postgres.com/api/pool>

SSL Disabled

statement\_timeout

Number of milliseconds before a statement in query will time out. Default is no timeout

Tej sekcji możemy nie dotykać

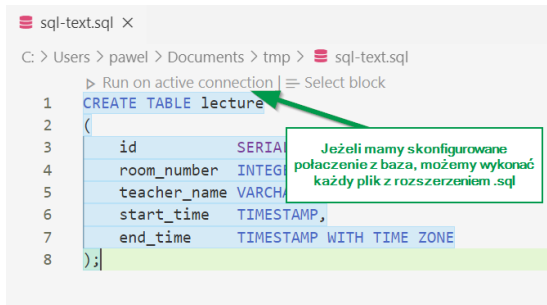
Rysunek: Rozszerzenie *SQLTools*

logintegra



# Integracja z Visual Studio Code c.d.

Teraz możemy otworzyć (lub stworzyć) dowolny plik z rozszerzeniem `.sql` – i wykonać zapytania na połączonej bazie za pomocą **Run on active connection**



Rysunek: Wykonywanie zapytań

- 1 Zainstaluj **PostgreSQL 13**
- 2 Utwórz użytkownika o dowolnej nazwie
- 3 Utwórz bazę danych o nazwie `wsb_demo`
- 4 Utwórz w bazie `wsb_demo` tabelę o nazwie `table_demo`
- 5 Skonfiguruj połączenie z `wsb_demo` w dowolnym narzędziu innym niż *pgAdmin* – np. IntelliJ, Visual Studio Code – używając użytkownika innego niż `postgres`; w ramach testu wykonaj:

---

```
SELECT *  
FROM table_demo;
```

---

Oczekiwany wynik to `0 rows retrieved`.

# Przykład – Zajęcia na uczelni

Założmy, że musimy przygotować bazę danych zajęć na uczelni. Dla każdej lekcji potrzebujemy pewnie m.in. takich informacji:

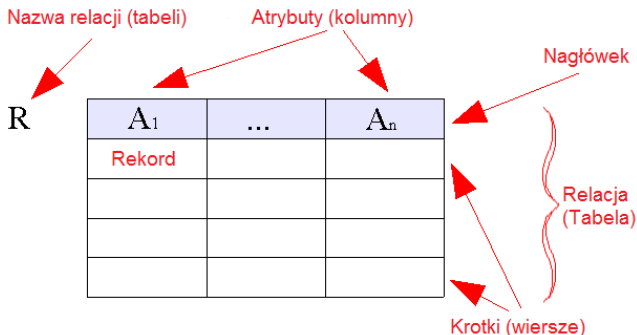
- ▶ Numer sali
- ▶ Nazwisko prowadzącego
- ▶ Data i godzina rozpoczęcia
- ▶ Data i godzina zakończenia
- ▶ Nazwa przedmiotu
- ▶ Liczba wolnych miejsc
- ▶ Czy jest to laboratorium?
- ▶ Adres budynku

# Projektowanie tabel – zaczniemy od zwykłego arkusza

Numer sali	Nazwisko prowadzącego	Data i godzina rozpoczęcia	Data i godzina zakończenia
1	Kasia Nowak	26.10.2020 11:30	26.10.2020 12:30
2	Jan Kowalski	27.10.2020 11:30	27.10.2020 13:30
3	Emilia Lewandowska	27.10.2020 14:00	27.10.2020 16:30
4	Jan Kowalski	29.10.2020 07:00	29.10.2020 11:00
1	Kasia Nowak	26.10.2020 11:00	25.10.2020 11:30

# Tabele w relacyjnych bazach danych

Wróćmy do modelu relacyjnego – co w naszym przykładzie będzie *tabelą*, *kolumną*, *wierszem*, *rekordem*?



Rysunek: Model relacyjny

# Tworzenie tabel w SQL

Utworzymy tabelę z poprzedniego slajdu za pomocą SQL

W naszym arkuszu mieliśmy następujące kolumny:

- ▶ Numer sali
- ▶ Nazwisko prowadzącego
- ▶ Data i godzina rozpoczęcia
- ▶ Data i godzina zakończenia

```
CREATE TABLE lecture
(
    room_number INTEGER,
    teacher_name VARCHAR(255),
    start_time   TIMESTAMP,
    end_time     TIMESTAMP
);
```

→ [Dokumentacja](#)

# Typy danych

Typów danych w PostgreSQL jest bardzo dużo – por. z [dokumentacją](#) – nie trzeba znać wszystkich, omówimy te najważniejsze

Name	Aliases	Description
bigint	int8	signed eight-byte integer
bigserial	serial8	autoincrementing eight-byte integer
bit [ ( n ) ]		fixed-length bit string
bit varying [ ( n ) ]	varbit [ ( n ) ]	variable-length bit string
boolean	bool	logical Boolean (true/false)
box		rectangular box on a plane
bytea		binary data ("byte array")
character [ ( n ) ]	char [ ( n ) ]	fixed-length character string
character varying [ ( n ) ]	varchar [ ( n ) ]	variable-length character string
cidr		IPv4 or IPv6 network address
circle		circle on a plane
date		calendar date (year, month, day)
double precision	float8	double precision floating-point number (8 bytes)
inet		IPv4 or IPv6 host address
integer	int, int4	signed four-byte integer
interval [ fields ] [ ( p ) ]		time span
json		textual JSON data
jsonb		binary JSON data, decomposed
line		infinite line on a plane
lseg		line segment on a plane
macaddr		MAC (Media Access Control) address
macaddr8		MAC (Media Access Control) address (EUI-64 format)
money		currency amount
numeric [ ( p, s ) ]	decimal [ ( p, s ) ]	exact numeric of selectable precision

path		geometric path on a plane
pg_log		PostgreSQL Log Sequence Number
pg_snapshot		user-level transaction ID snapshot
point		geometric point on a plane
polygon		closed geometric path on a plane
real	float4	single precision floating-point number (4 bytes)
smallint	int2	signed two-byte integer
smallserial	serial2	autoincrementing two-byte integer
serial	serial4	autoincrementing four-byte integer
text		variable-length character string
time [ ( p ) ] [ without time zone ]		time of day (no time zone)
time [ ( p ) ] with time zone	timetz	time of day, including time zone
timestamp [ ( p ) ] [ without time zone ]		date and time (no time zone)
timestamp [ ( p ) ] with time zone	timestampz	date and time, including time zone
tsquery		text search query
tsvector		text search document
txid_snapshot		user-level transaction ID snapshot (deprecated: see pg_snapshot)
uuid		universally unique identifier
xml		XML data

Rysunek: Typy danych c.d.

Rysunek: Typy danych

# Najważniejsze typy danych

- ▶ `CHAR(n)` – ciąg znaków o długości `n` znaków
- ▶ `VARCHAR(n)` – ciąg znaków o długości maksymalnie `n` znaków
- ▶ `TEXT` – ciąg znaków o nieograniczonej długości
- ▶ `BOOLEAN` – wartość logiczna  
`TRUE` / `FALSE`
- ▶ `SERIAL` – automatycznie rosnąca liczba
- ▶ `INTEGER` – liczby z zakresu -2147483648 do +2147483647
- ▶ `BIGINT` – liczby z ogromnego zakresu
- ▶ `REAL` – liczby zmiennoprzecinkowe
- ▶ `TIMESTAMP` –  
2020-10-25 12:23:54
- ▶ `TIMESTAMP WITH TIME ZONE`  
– 2020-10-25 12:23:54+02



# Dodawanie danych – INSERT INTO

Dodamy dane za pomocą SQL

---

-- Dodanie wiersza do tabeli wykonujemy za pomocą polecenia INSERT INTO

```
INSERT INTO lecture ( room_number, teacher_name, start_time, end_time )  
VALUES ( 1, 'Kasia Nowak', '2020-10-26 11:30', '2020-10-26 12:30' );
```

-- Możemy dodać kilka wierszy na raz

```
INSERT INTO lecture ( room_number, teacher_name, start_time, end_time )  
VALUES ( 2, 'Jan Kowalski', '2020-10-27 11:30', '2020-10-27 13:30' ),  
      ( 3, 'Emilia Lewandowska', '2020-10-27 14:00', '2020-10-27 16:30' ),  
      ( 4, 'Jan Kowalski', '2020-10-29 07:00', '2020-10-29 11:00' ),  
      ( 1, 'Kasia Nowak', '2020-10-26 11:00', '2020-10-26 11:30' );
```

---

→ [Dokumentacja](#)

# Zmiana struktury bazy – ADD COLUMN

Dodać nowe kolumny możemy za pomocą

```
ALTER TABLE ... ADD COLUMN
```

---

**ALTER TABLE** lecture  
**ADD COLUMN** name TEXT;

**ALTER TABLE** lecture  
**ADD COLUMN** is\_lab BOOLEAN **DEFAULT FALSE**;

---

→ [Dokumentacja](#)

# Zmiana struktury bazy – DROP COLUMN

Usuwanie kolumny za pomocą `ALTER TABLE ... DROP COLUMN`

---

```
ALTER TABLE lecture  
DROP COLUMN IF EXISTS is_lab;
```

---

`IF EXISTS` sprawi, że PostgreSQL nie zwróci błędu, jeśli kolumna, którą chcemy usunąć, nie będzie istniała w bazie.

→ [Dokumentacja](#)

# Klucz główny

**Klucz główny** jest to kolumna (bądź kilka kolumn) jednoznacznie identyfikujących wiersz w tabeli.

Jak na razie nasza tabela nie ma takiego klucza

---

```
INSERT INTO lecture ( room_number, teacher_name, start_time, end_time )  
VALUES ( 1, 'Kasia Nowak', '2020-10-26 11:30', '2020-10-26 12:30' );
```


```
INSERT INTO lecture ( room_number, teacher_name, start_time, end_time )  
VALUES ( 1, 'Kasia Nowak', '2020-10-26 11:30', '2020-10-26 12:30' );
```

---

```
wsb_demo-> SELECT *  
wsb_demo-> FROM lecture;
```

room_number	teacher_name	start_time	end_time
1	Kasia Nowak	2020-10-26 11:30:00	2020-10-26 12:30:00+00
1	Kasia Nowak	2020-10-26 11:30:00	2020-10-26 12:30:00+00

(2 rows)



Nie możemy jednoznacznie zidentyfikować tych wierszy

Rysunek: Tabela bez klucza głównego

# Tworzenie tabeli z kluczem głównym

```
CREATE TABLE lecture
(  
    id          SERIAL PRIMARY KEY,  
    room_number INTEGER,  
    teacher_name VARCHAR(255),  
    start_time  TIMESTAMP,  
    end_time    TIMESTAMP WITH TIME ZONE  
);
```

- ▶ **PRIMARY KEY** – Ustawia kolumnę jako **Klucz główny**
- ▶ **SERIAL** – Każdy nowy wiersz będzie w kolumnie **id** zawierał kolejną liczbę (1, 2, 3 itd.)

# Dodanie klucza głównego do istniejącej tabeli

Możemy dodać kolumnę `id` do naszej tabeli za pomocą `ADD COLUMN` – określając, że jest to `PRIMARY KEY`

---

```
ALTER TABLE lecture  
  ADD COLUMN id SERIAL PRIMARY KEY;
```

---