SQL - wprowadzenie

Korepetycje.intro23wertyk@gmail.com Bartosz Pawliczak

2021-01-23

Materiały ogólnodostępne przygotowane przeze mnie:
https://github.com/bpawliczak/intro23wertyk_public

Pełne zestawienie materiałów (tylko dla kursantów):
https://github.com/bpawliczak/intro23wertyk

Więcej informacji i inspiracji:
https://www.facebook.com/intro23wertyk

SQL

SQL (ang. Structured Query Language) – strukturalny język zapytań używany do tworzenia, modyfikowania baz danych oraz do umieszczania i pobierania danych z baz danych.
Język SQL jest językiem deklaratywnym. Decyzję o sposobie przechowywania i pobrania danych pozostawia się systemowi zarządzania bazą danych (DBMS).

SQL

SQL (ang. Structured Query Language) – strukturalny język zapytań używany do tworzenia, modyfikowania baz danych oraz do umieszczania i pobierania danych z baz danych.
Język SQL jest językiem deklaratywnym. Decyzję o sposobie przechowywania i pobrania danych pozostawia się systemowi zarządzania bazą danych (DBMS).

Co to oznacza tak naprawdę oznacza? SQL to język zapytań, który może być używany na wielu systemach bazodanowych (np. Firebird/Firebase, MS SQL, MySQL, Oracle Database, SQLite).

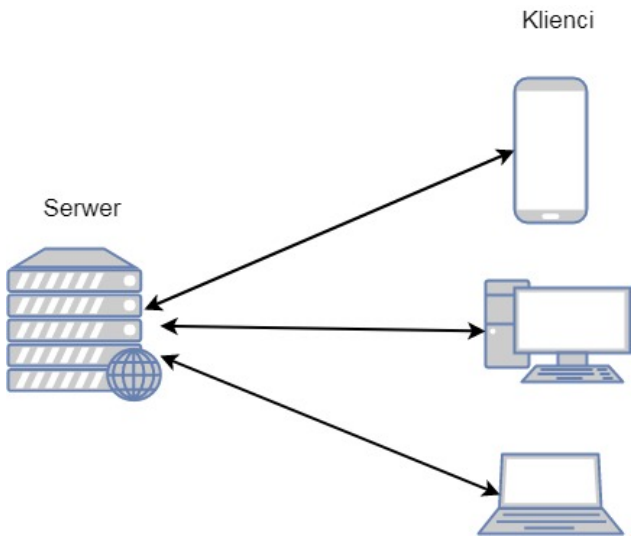
SQL

SQL (ang. Structured Query Language) – strukturalny język zapytań używany do tworzenia, modyfikowania baz danych oraz do umieszczania i pobierania danych z baz danych.
Język SQL jest językiem deklaratywnym. Decyzję o sposobie przechowywania i pobrania danych pozostawia się systemowi zarządzania bazą danych (DBMS).

Co to oznacza tak naprawdę oznacza? SQL to język zapytań, który może być używany na wielu systemach bazodanowych (np. Firebird/Firebase, MS SQL, MySQL, Oracle Database, SQLite). Użycie SQL, zgodnie z jego nazwą, polega na zadawaniu zapytań do bazy danych i na tym się skupimy.

System zarządzania bazą danych

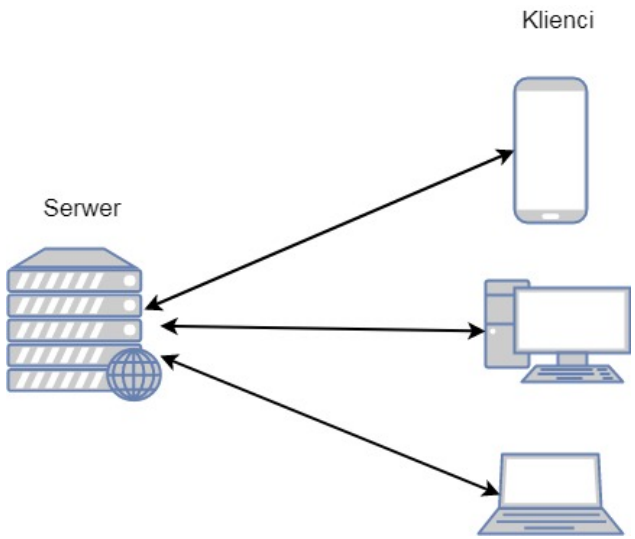
Większość obecnie spotykanych systemów działa w trybie klient-serwer, gdzie baza danych jest udostępniana klientom przez SZBD będący serwerem. Serwer bazy danych może udostępniać dane klientom bezpośrednio lub przez inny serwer, np. poprzez serwer WWW lub serwer aplikacji.



Rysunek: Mamy wiele różnych systemów zarządzania bazą danych, ale działają na tej samej zasadzie i składnia SQL używania w różnych jest zbliżona.

System zarządzania bazą danych

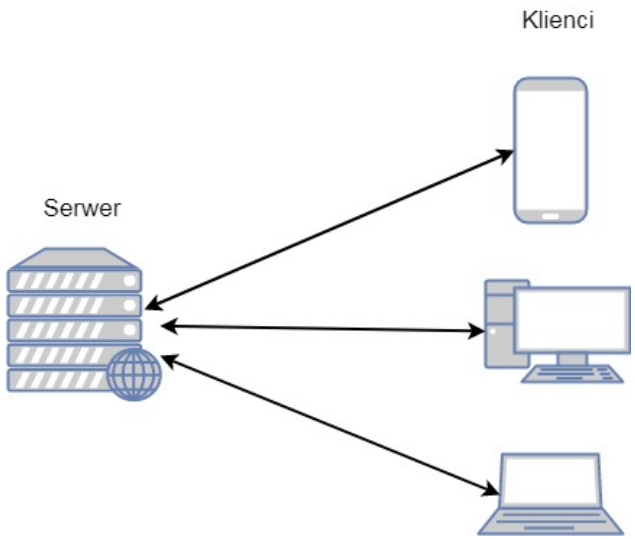
Większość obecnie spotykanych systemów działa w trybie klient-serwer, gdzie baza danych jest udostępniana klientom przez SZBD będący serwerem. Serwer bazy danych może udostępniać dane klientom bezpośrednio lub przez inny serwer, np. poprzez serwer WWW lub serwer aplikacji.



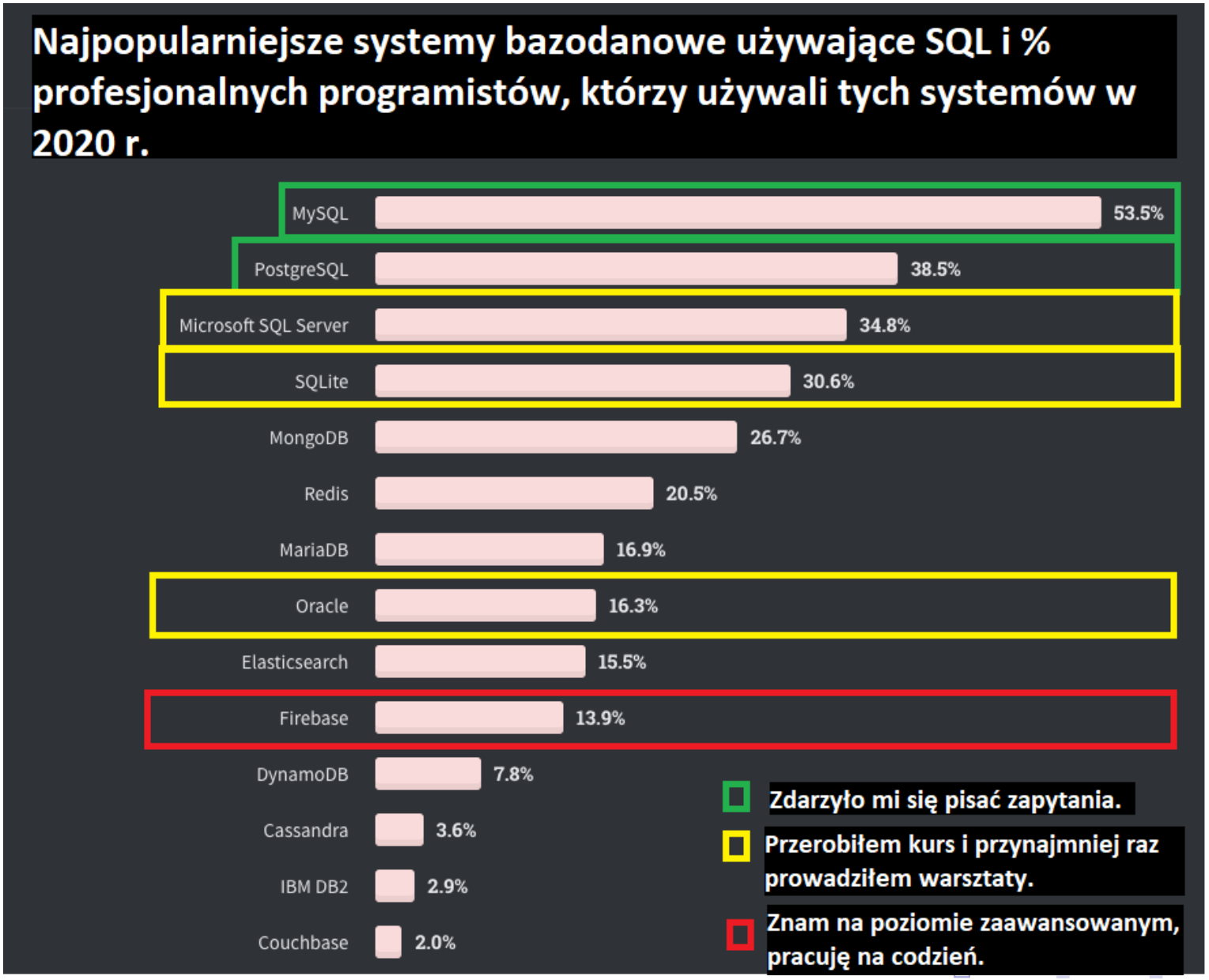
Rysunek: Mamy wiele różnych systemów zarządzania bazą danych, ale działają na tej samej zasadzie i składnia SQL używania w różnych jest zbliżona.

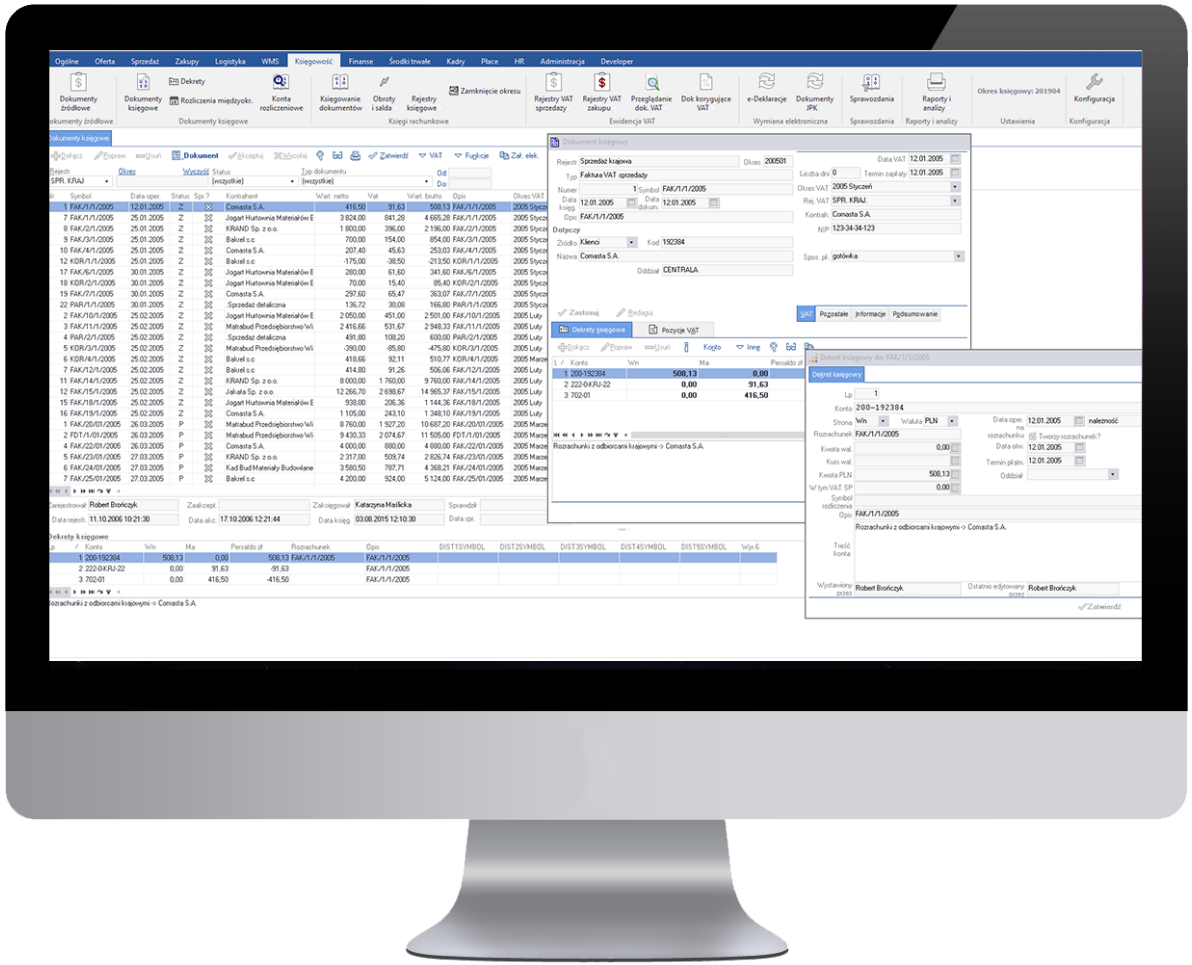
System zarządzania bazą danych

Większość obecnie spotykanych systemów działa w trybie klient-serwer, gdzie baza danych jest udostępniana klientom przez SZBD będący serwerem. Serwer bazy danych może udostępniać dane klientom bezpośrednio lub przez inny serwer, np. poprzez serwer WWW lub serwer aplikacji.

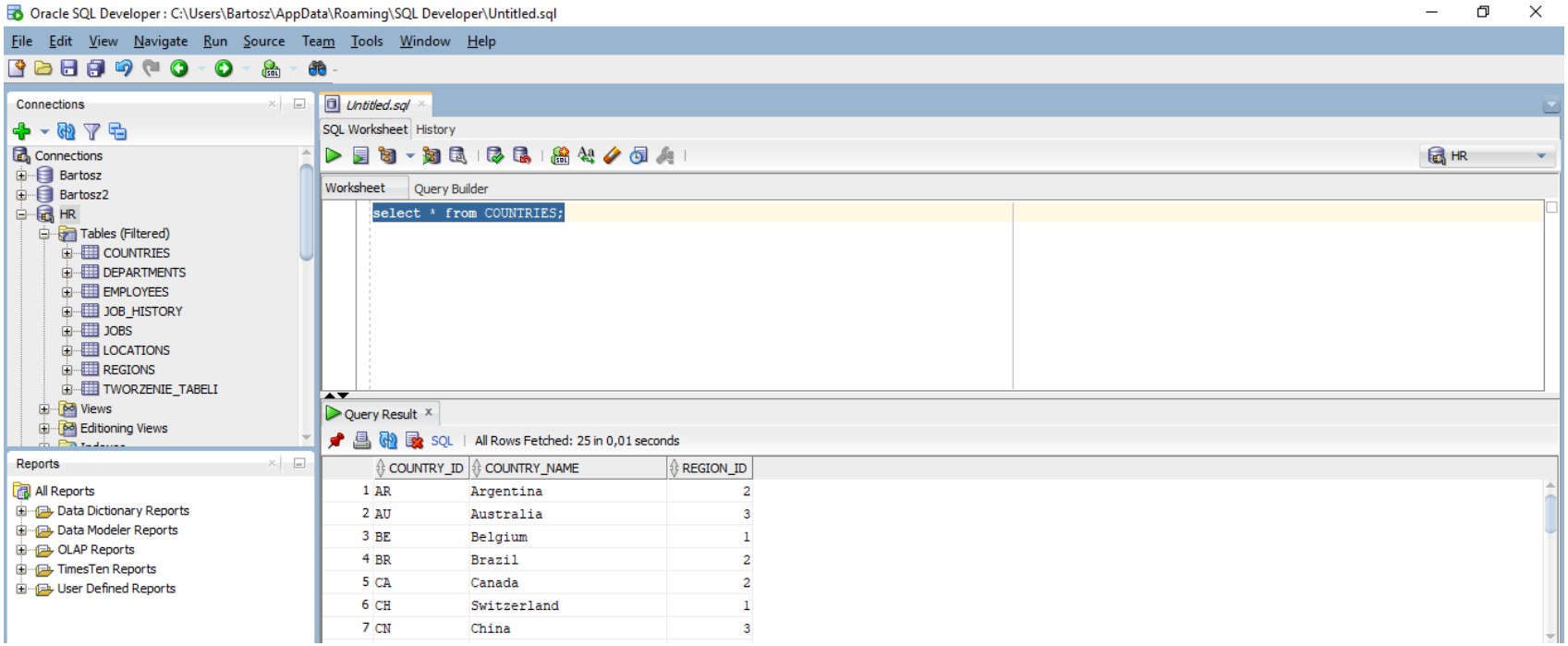


Rysunek: Mamy wiele różnych systemów zarządzania bazą danych, ale działają na tej samej zasadzie i składnia SQL używania w różnych jest zbliżona.

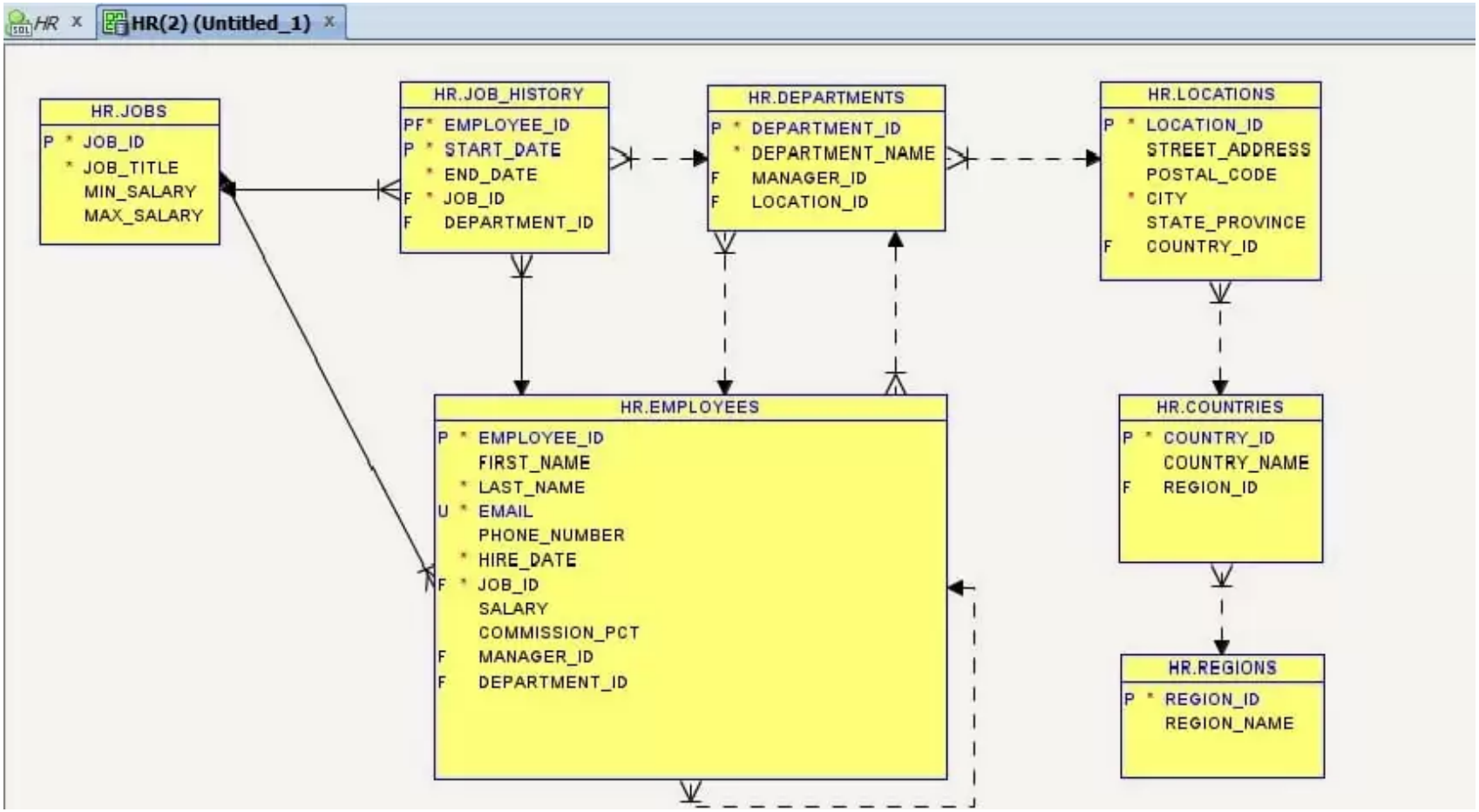

Najpopularniejsze systemy bazodanowe używające SQL i % profesjonalnych programistów, którzy używali tych systemów w 2020 r.
MySQL 53.5%
PostgreSQL 38.5%
Microsoft SQL Server 34.8%
SQLite 30.6%
MongoDB 26.7%
Redis 20.5%
MariaDB 16.9%
Oracle 16.3%
Elasticsearch 15.5%
Firebase 13.9%
DynamoDB 7.8%
Cassandra 3.6%
IBM DB2 2.9%
Couchbase 2.0%
Zdarzyło mi się pisać zapytania.
Przerobiłem kurs i przynajmniej raz prowadziłem warsztaty.
Znam na poziomie zaawansowanym, pracuję na codzień.



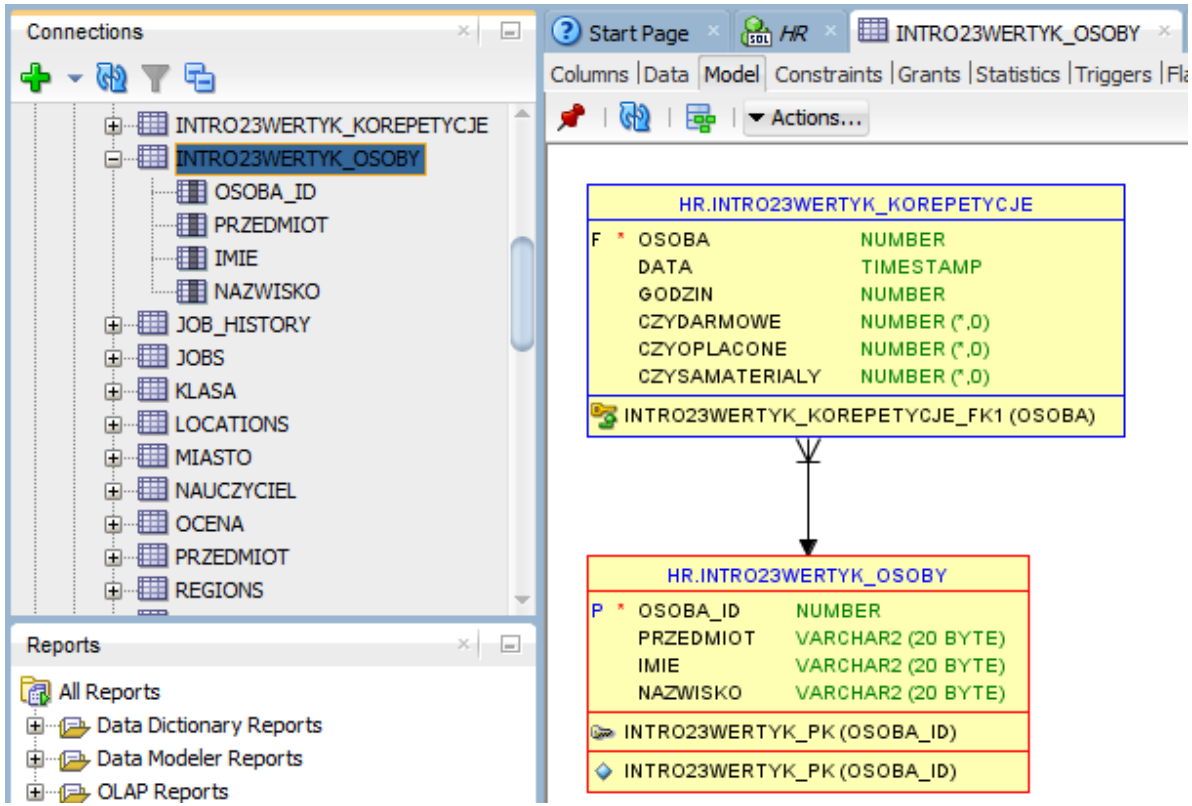
Rysunek: Przykład zastosowania systemu zarządzania bazy danych w aplikacji typu ERP. Źródło: https://sente.pl/teneum/finanse-i-ksiegowosc/funkcjonalnosc/ksiegowosc/



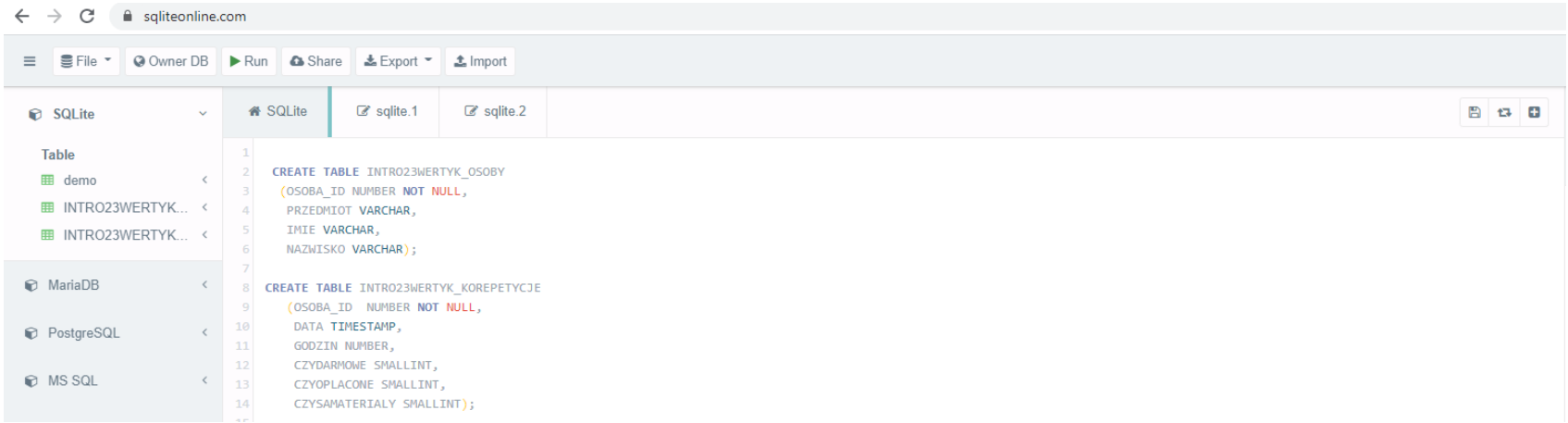
Rysunek: Podstawowe okno. Aby uzyskać taki efekt, można skonfigurować system bazodanowy pod Oracle (godny polecenia link, jest tam cały kurs do Oracle'a: http://andrzejklusiewicz.blogspot.com/2010/11/kurs-oracle-sql.html)



Rysunek: Kluczowe jest rozpoznanie danych, żebyśmy w ogóle wiedzieli o czym mówimy. Widzimy tutaj przykładowy UML.



Rysunek: Jakaś prosta struktura na początek?



Rysunek: Czas wybrać system bazodanowy na początek. Propozycja: SQLite - najczęściej używany przy integracjach, najprzyjaźniejszy w konfiguracji, można używać z poziomu przeglądarki.

```
##Przeglądanie większych zbiorów
**SQL**? Warto spróbować!
```{r, message=FALSE, warning = FALSE}
library(sqldf)
zapytanie1<-"select Age, Sex, Risk
from data2
order by Age desc
limit 5"
sqldf(zapytanie1)
```
##Wczytywanie konkretnych wyników
```{r, message=FALSE, warning = FALSE}
zapytanie0<-read.csv.sql("dane/german_credit_data.csv",
sql = "select Age, Sex, Risk
from file order by Age desc limit 5")
print(zapytanie0)
```
##Przykładowe agregacje
Średni wiek ,,złych'' mężczyzn...
```{r, message=FALSE, warning = FALSE}
zapytanie2 = "select avg(Age) as sredni_wiek
from data2 WHERE Sex='male' and Risk=1"
sqldf(zapytanie2)
```
```

Rysunek: Integracja SQL z R/Pythonem? W ramach działalności STWUR-a współorganizowałem warsztaty z tym zagadnieniem. Źródło: https://github.com/STWUR/eRementarz-06-04-2019/

Co jest uznawane za podstawe pisania zapytań? Czym się różnią różne systemy bazodanowe?

SELECT <*attribute-list and function-list*>
FROM <*table-list*>
[**WHERE** <*condition*>]
[**GROUP BY** <*grouping attribute-list*>]
[**HAVING** <*group condition*>]
[**ORDER BY** <*attribute-list*>];

Rysunek: Omówimy sobie kolejno co oznacza sam SELECT, jakie warunki warto dawać do where'a i do kolejnych części.

Zatem SQLite!

Rysunek: Na poprzednim slajdzie ze stroną dodaliśmy tabele, teraz potrzebne są jakieś wartości. Gdy one już będą, możemy zacząć pisać zapytania.

Najprostsze możliwe zapytanie?

Najprostsze możliwe zapytanie? Wyświetlanie wszystkich kolumn z tabeli, tzn.
SELECT * FROM intro23wertyk_korepetycje;
Prosty przypadek, który oznacza dokładnie to samo, co
SELECT osoba, data,
godzin,czydarmowe,czyoplacone,czysamaterialy FROM
intro23wertyk_korepetycje;

Najprostsze możliwe zapytanie? Wyświetlanie wszystkich kolumn z tabeli, tzn.
SELECT * FROM intro23wertyk_korepetycje;
Prosty przypadek, który oznacza dokładnie to samo, co
SELECT osoba, data, godzin,czydarmowe,czyoplacone,czysamaterialy FROM intro23wertyk_korepetycje;
Słowa kluczowe SELECT, FROM występują praktycznie zawsze (o wyjątkach nie będziemy na razie mówić, to coś na późniejsze korepetycje), nie jest ważna wielkość liter w tym kontekście.

Najprostsze możliwe zapytanie? Wyświetlanie wszystkich kolumn z tabeli, tzn.
SELECT * FROM intro23wertyk_korepetycje;
Prosty przypadek, który oznacza dokładnie to samo, co
SELECT osoba, data,
godzin,czydarmowe,czyoplacone,czysamaterialy FROM
intro23wertyk_korepetycje;
Słowa kluczowe SELECT, FROM występują praktycznie zawsze (o wyjątkach nie będziemy na razie mówić, to coś na późniejsze korepetycje), nie jest ważna wielkość liter w tym kontekście. Polecam testować od razu na stronie, bo zaraz przyspieszymy z tempem!

Chcemy teraz wyciągnąć tylko część informacji, np. tylko informacje o korepetycjach, które zostały opłacone. Jak to zrobić?

Chcemy teraz wyciągnąć tylko część informacji, np. tylko informacje o korepetycjach, które zostały opłacone. Jak to zrobić?
SELECT * FROM intro23wertyk_korepetycje WHERE czyoplacone=1;

Chcemy teraz wyciągnąć tylko część informacji, np. tylko informacje o korepetycjach, które zostały opłacone. Jak to zrobić?
SELECT * FROM intro23wertyk_korepetycje WHERE czyoplacone=1;
A gdybyśmy chcieli sprawdzić tylko opłacone i te, do których są materiały?

Chcemy teraz wyciągnąć tylko część informacji, np. tylko informacje o korepetycjach, które zostały opłacone. Jak to zrobić?
SELECT * FROM intro23wertyk_korepetycje WHERE czyoplacone=1;
A gdybyśmy chcieli sprawdzić tylko opłacone i te, do których są materiały?
SELECT * FROM intro23wertyk_korepetycje WHERE czysamaterialy=1 AND czyoplacone=1;

Chcemy teraz wyciągnąć tylko część informacji, np. tylko informacje o korepetycjach, które zostały opłacone. Jak to zrobić?
SELECT * FROM intro23wertyk_korepetycje WHERE czyoplacone=1;
A gdybyśmy chcieli sprawdzić tylko opłacone i te, do których są materiały?
SELECT * FROM intro23wertyk_korepetycje WHERE czysamaterialy=1 AND czyoplacone=1;
Teraz jest chwila, aby omówić w jaki sposób tworzy się warunki logiczne, jeżeli są one niejasne.

No to lećmy dalej! W jaki sposób wyświetlić tylko korki, które były po 28.01?

No to lećmy dalej! W jaki sposób wyświetlić tylko korki, które były po 28.01?
SELECT * FROM intro23wertyk_korepetycje WHERE data>='29-01-2021'

No to lećmy dalej! W jaki sposób wyświetlić tylko korki, które były po 28.01?

SELECT * FROM intro23wertyk_korepetycje WHERE data>='29-01-2021'

To, co jest ważne, to fakt, że trzeba pamiętać o odpowiednim formacie. Apostrofy są bardzo ważne w przypadku tekstu, ale też dat!

Zadania na przećwiczenie:

- Napisz zapytanie, które zwróci tylko kolumny: DATA, GODZIN.
- Napisz zapytanie, które zwróci tylko korepetycje, które trwały więcej niż jedną godzinę.
- Napisz zapytanie, które zwróci tylko korepetycje, które trwały więcej niż jedną godzinę, są opłacone i są do nich materiały.
- Napisz zapytanie, które zwróci korepetycje sprzed 29.01.2021, które były darmowe lub trwały więcej niż godzinę.

To póki co wersja wstępna pliku. Jeżeli zależy Ci na podstawach, to w kolejnym kroku na pewno warto omówić JOIN, potem GROUP BY, HAVING, ORDER BY. W momencie gdy to czytasz, w strefie dla moich kursantów na pewno są już materiały na ten temat, ale przede wszystkim ćwiczymy, a tego nie ma na prezentacji. Jeżeli zatem znasz to wszystko, a interesują Cię bardziej skomplikowane zapytania, możemy omawiać skomplikowane zapytania. Jeżeli interesuje Cię bardziej projektowanie bazy danych, możemy się skupić właśnie na tym. A może znasz już dobrze selecty, ale nigdy w praktyce nie robiłeś triggerów? Postaram się dopasować do potrzeb. Napisz na maila, powołaj się na tę prezentację.