Tworzenie aplikacji bazodanowych

wykład

Podstawy baz danych przypomnienie

Joanna Kołodziejczyk

2015

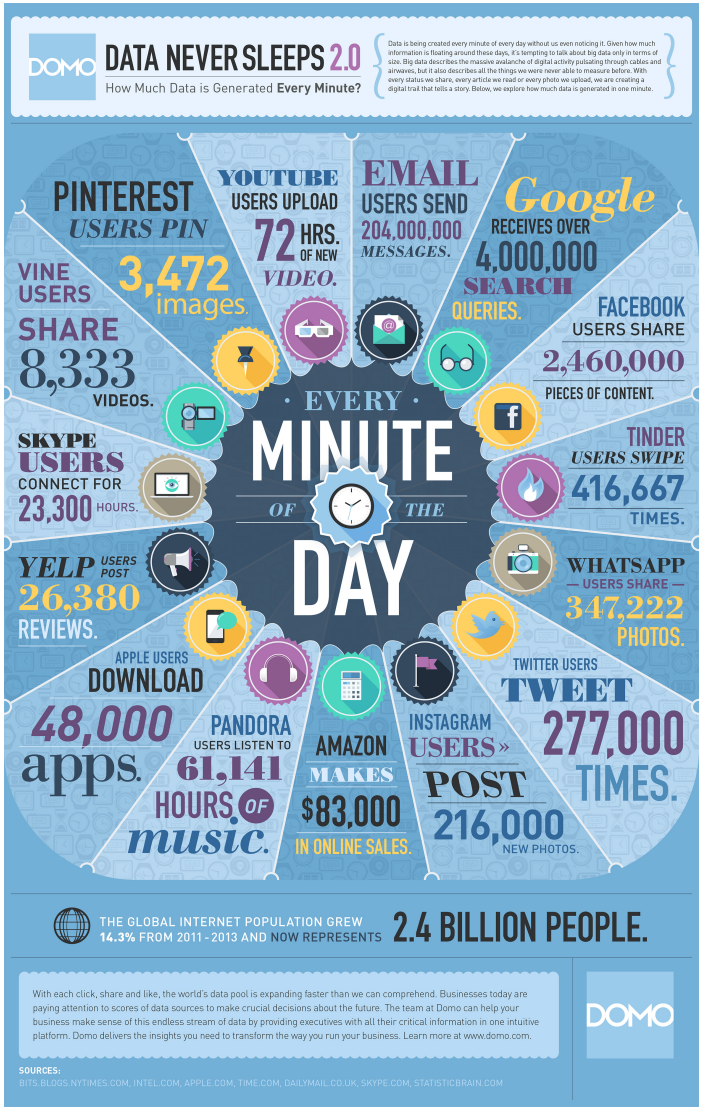
DBMS

Database Management System (DBMS) zapewnia

1. wydajne
2. niezawodne
3. wygodne
4. bezpieczne
5. wielo-użytkownikowe
6. trwałe

przechowywanie i dostęp do dużych ilości danych.

Terabajty danych każdego dnia



Pamięć a dane

- Gdzie przechowuje się dane?
- Czy wystarczy pamięci operacyjnej?

Dane w bazach przechowywane są poza pamięcią operacyjną.

Trwałość

- Do żyje dłużej: Dane czy software?
- Jaka jest różnica w danych programu, a danych w bazie danych?

Bezpieczeństwo

- Dane w bankach?
- Dane wrażliwe?
- Czy może się zdarzyć utrata, włamanie, zmiana?

Wielu użytkowników

- Czy wiele aplikacji może użytkować dane z tej samej bazy danych?
- Jak zapewnia się wielodostęp?

Wygoda

- DBMS są tak konstruowany by ułatwić dostęp, obsługę danych w dużych ilościach.
- Physical Data Independence - dane są przechowywane na dysku niezależnie od programu i sposobu operowania danymi w programie. Np. w programie dane z bazy mogą być przechowywane i wykorzystywane w dowolny sposób.
- Z niezależności fizycznej wynika struktura języków zapytań do baz danych. Języki są deklaratywne, mówisz co chcesz uzyskać z bazy, a nie w jaki sposób.

Efektywność

Powiedzenie

Three most important things in a database system is first performance, second performance and again performance.

Bazy wykonują tysiące zapytań na sekundę.

Niezawodność

DBMS zapewniają 99,9999% niezawodność.

Obsługa

1. Aplikacje bazodanowe mogą być programowane poprzez "frameworks": Django, Ruby on Rails.
2. DBMS może działać w połączeniu z "middleware": application servers, web servers, pomaga w interakcji z bazą danych.
3. Aplikacji na olbrzymich danych mogą w ogóle nie używać DBMS. Dane mogą być przechowywane w plikach: Hadoop, MongoDB.

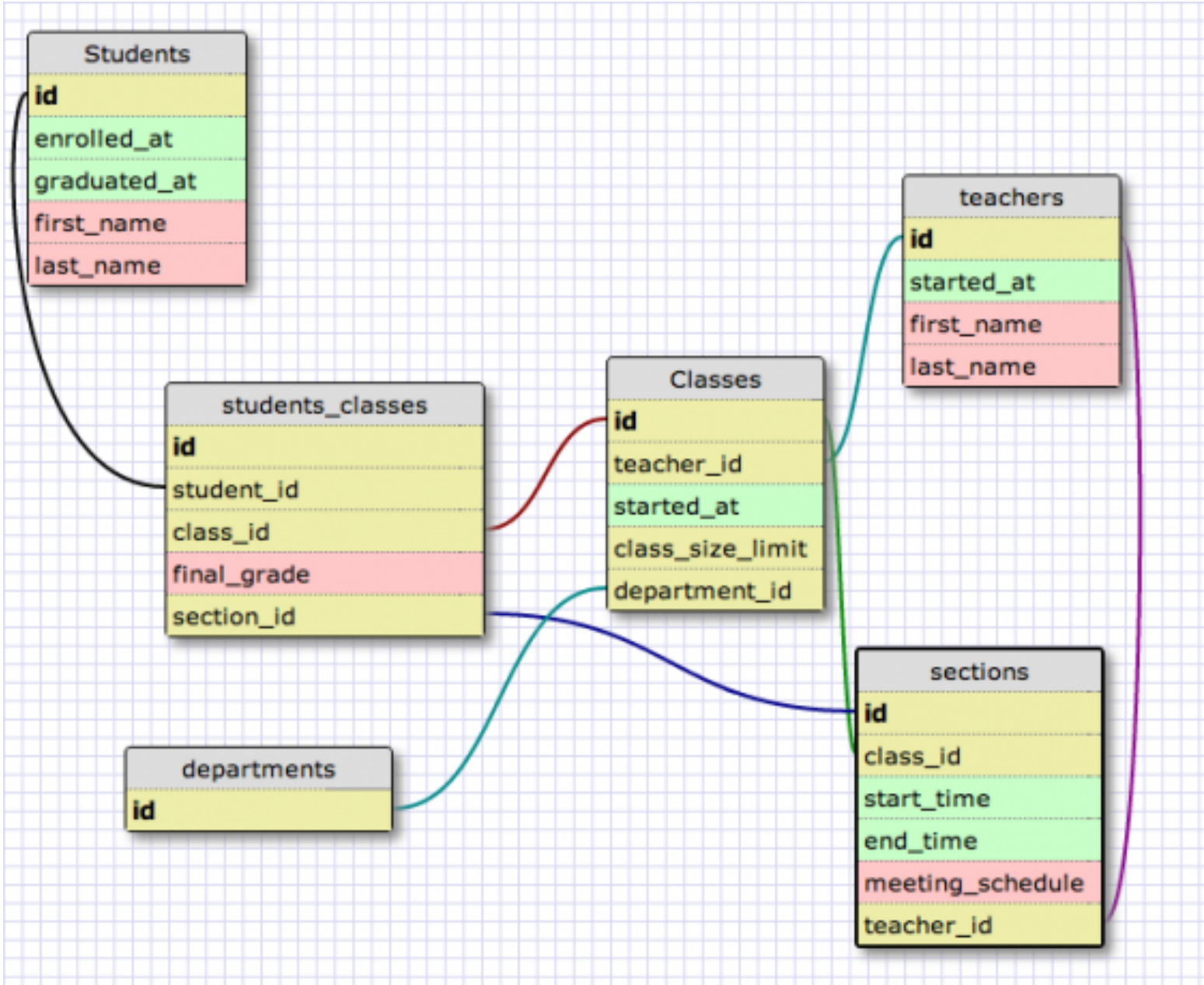
Kluczowe zagadnienia

1. Model danych — opis struktury danych.
2. Schemat vs dane.
3. DLL - data definition language
4. DML - Data manipulation or query language

Model danych

- Model relacyjny — jeden z popularniejszych. W modelu tym o danych myśli się jak o zbiorze rekordów.
- Dokumenty XML — o danych myśli się jak o hierarchicznej strukturze danych etykietowanych.
- Model graficzny — dane zaprezentowane w postaci węzłów i krawędzi.

Model danych

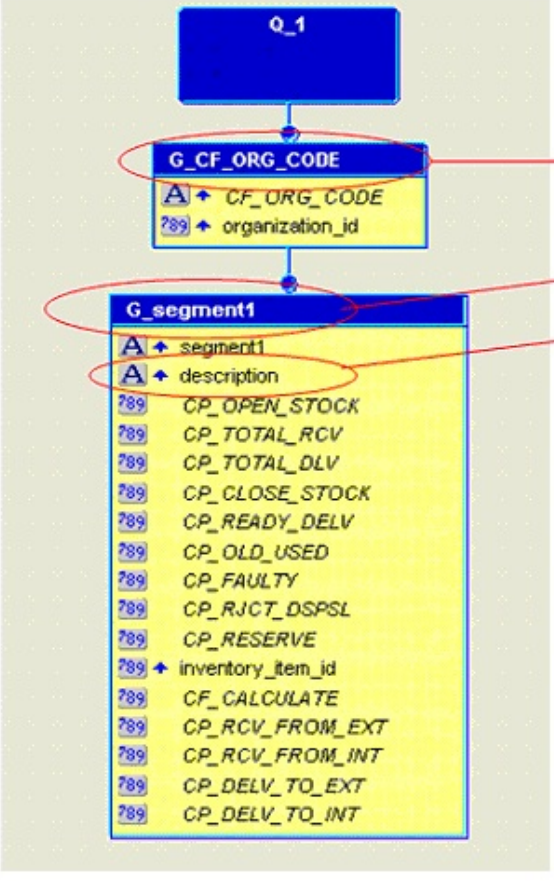


Model danych

```
<?xml version="1.0"?>
<contact-info>
    <contact1>
        <name>Tanmay Patil</name>
        <company>TutorialsPoint</company>
        <phone>(011) 123-4567</phone>
    </contact1>
    <contact2>
        <name>Manisha Patil</name>
        <company>TutorialsPoint</company>
        <phone>(011) 789-4567</phone>
    </contact2>
</contact-info>
```

Model danych

Mapping of Data Model and XML Output



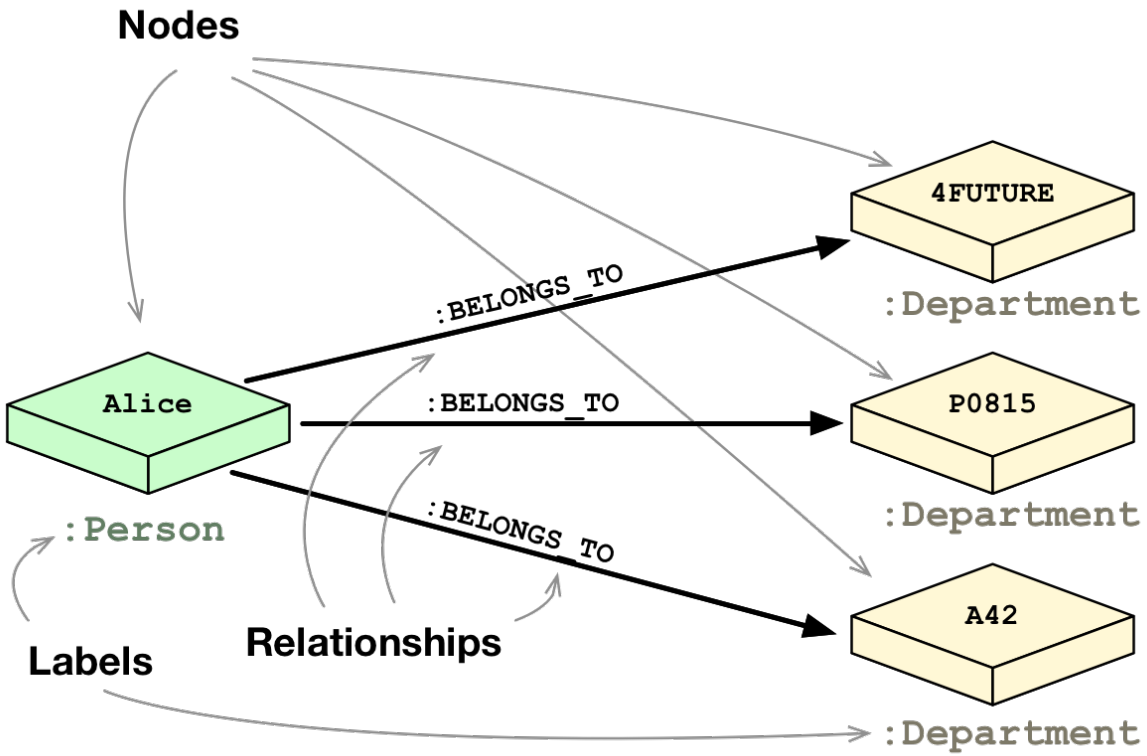
Data Model

```
<?xml version="1.0" ?>
<!-- Generated by Oracle Reports version 6.0.8.24.0 -->
- <GP_INV_RLCCLC_STK>
  - <LIST_G_CF_ORG_CODE>
    - <G_CF_ORG_CODE>
      <ORGANIZATION_ID>85</ORGANIZATION_ID>
      - <LIST_G_SEGMENT1>
        - <G_SEGMENT1>
          <SEGMENT1>1000496</SEGMENT1>
          <DESCRIPTION>Siemens A 62</DESCRIPTION>
          <INVENTORY_ITEM_ID>2001</INVENTORY_ITEM_ID>
          <CP_OPEN_STOCK>1</CP_OPEN_STOCK>
          <CP_TOTAL_RCV>0</CP_TOTAL_RCV>
          <CP_TOTAL_DLV>0</CP_TOTAL_DLV>
          <CP_CLOSE_STOCK>1</CP_CLOSE_STOCK>
          <CP_READY_DELV>1</CP_READY_DELV>
          <CP_OLD_USED>0</CP_OLD_USED>
          <CP_FAULTY>0</CP_FAULTY>
          <CP_RJCT_DSPSL>0</CP_RJCT_DSPSL>
          <CP_RESERVE>0</CP_RESERVE>
          <CF_CALCULATE>0</CF_CALCULATE>
          <CP_RCV_FROM_EXT>0</CP_RCV_FROM_EXT>
          <CP_RCV_FROM_INT>0</CP_RCV_FROM_INT>
          <CP_DELV_TO_EXT>0</CP_DELV_TO_EXT>
          <CP_DELV_TO_INT>0</CP_DELV_TO_INT>
        </G_SEGMENT1>
      </LIST_G_SEGMENT1>
      <CF_ORG_CODE>DHL</CF_ORG_CODE>
    </G_CF_ORG_CODE>
  </LIST_G_CF_ORG_CODE>
</GP_INV_RLCCLC_STK>
```

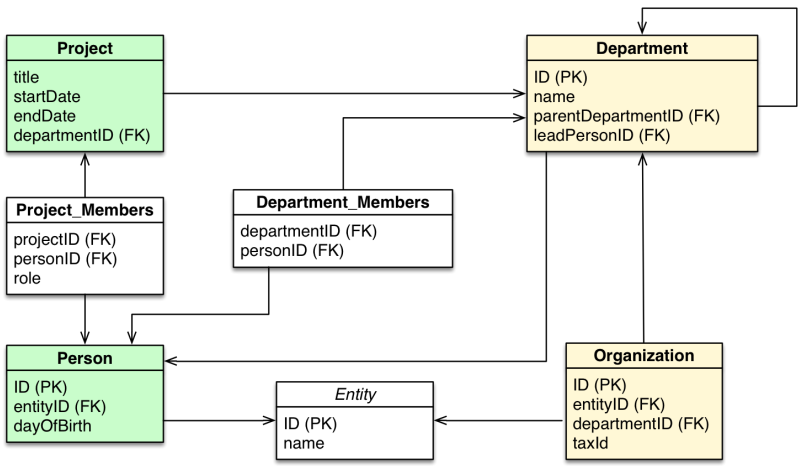
XML Output Generated after the report is run

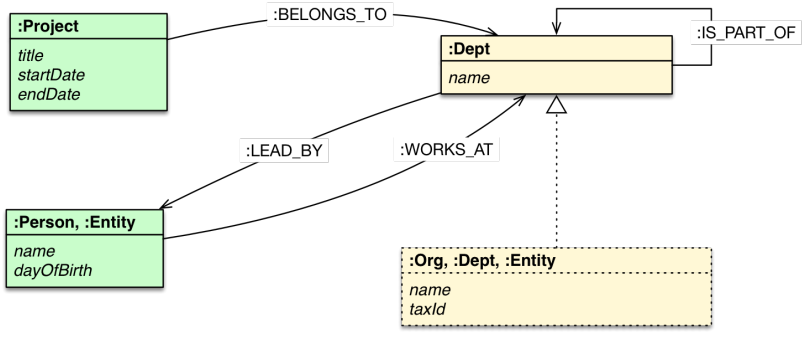
The XML output must be saved on the Local Machine with .xml extension

Model danych



Model danych





Schemat vs dane

Schemat to struktura bazy danych. Dane to rekordy.
Co się częściej zmienia?
Co jest ustalane na początku procesu projektowania?

DDL - data definition language

Używany, by stworzyć strukturę bazy danych.
`https://en.wikipedia.org/wiki/Data_definition_language`

DML - Data manipulation or query language

Po utworzeniu struktury bazy danych i załadowaniu jej danymi można zacząć posługiwać się językiem zapytań.

Kluczowe postaci

1. Specjalista implementujący DBMS
2. Projektant bazy danych
3. Programista aplikacji bazodanowej
4. Administrator bazy danych

Zalety modelu relacyjnego

1. Używany przez większość systemów komercyjnych
2. Bardzo prosty model
3. Możliwość odpytywania w językach wyższego rzędu: prosty ale ekspresywny.
4. Wydajna implementacja

Podstawowe konstrukcje w modelu relacyjnym

Baza danych składa się ze zbioru relacji, zwanych dalej *tabelą*, z których każda ma nazwę.
Fikcyjny przykład będzie dotyczył studentów aplikujących na uczelnie.
Potrzebne są dwie tabele

1. Student
2. Uczelnia

Podstawowe konstrukcje w modelu relacyjnym

Następnie potrzebne są atrybuty (cechy).

Tabela *student*

ID	Name	AvG	Photo

Tabela *uczelnia*

Nazwa	Miejsce	IleM

Podstawowe konstrukcje w modelu relacyjnym

Rzeczywiste dane przechowywane są w wierszach tabeli zwanych krotkami.

Tabela *student*

ID	Name	AvG	Photo
123	Anna	50%	anna.jpg
234	Marek	25%	Null
345	Paweł	Null	pawel.jpg

Podstawowe konstrukcje w modelu relacyjnym

Tabela *uczelnia*

Nazwa	Miejsce	IleM
PWSZ	Gorzów Wlk.	2 000
ZUT	Szczecin	5 000
US	Szczecin	6 000

Podstawowe konstrukcje w modelu relacyjnym

- Każdy atrybut ma typ, np,. ID to może być liczba całkowita, zdjęcie plik w formacie jpg itp, itd.
- Można stosować też typ wyliczeniowy np. dla województw (np 16 skrótów dla każdego województwa).
- Typy mogą być atomowe (jak w przykładzie lub złożone)

Podstawowe konstrukcje w modelu relacyjnym

Schemat

Opis strukturalny relacji w bazie danych. Zawiera nazwę relacji i atrybutów oraz typy tych atrybutów.

Instancja, krotka

rzeczywista zawartość w danym momencie. Wiersze zmieniają się w czasie.

Podstawowe konstrukcje w modelu relacyjnym

Wartości NULL — wartość jest nieznana lub nieokreślona.

Mając w tabeli wartości NULL trzeba być ostrożnym z zapytaniami:

np. Zapytaniem chcemy uzyskać listę wszystkich studentów ze średnim wynikiem z matury podstawowej wyższym niż 30%.
W wyniku uzyska się: Anna, nie Marek i nie Paweł.
Jeżeli zapytamy o wszystkich studentów z wynikiem mniejszym równym 30% uzyskamy: nie Anna, Marek i nie Paweł.

Podstawowe konstrukcje w modelu relacyjnym

Klucz

jest atrybutem lub zbiorem atrybutów w tabeli, gdzie każda wartość tego atrybutu (zbioru) jest unikalna.

Np. w tabeli Student zakładamy, że ID będzie niepowtarzalnym numerem i będzie kluczem.

Np. w tabeli Uczelnia nazwa nie musi być niepowtarzalna. W takim przypadku kluczem staje się zbiór atrybutów, np. nazwa i miejsce.

Podstawowe konstrukcje w modelu relacyjnym

Cele stosowania klucza:

- by zidentyfikować konkretne krotki — DBMS by zwiększyć wydajność przechowuje dane w określonej strukturze, która pozwala na szybkie wyszukiwanie po kluczu.
- by odnieść się do krotki w innej tabeli. Czyli jedna tabela odnosi się do krotki w innej tabeli za pomocą klucza.

Podstawowe konstrukcje w modelu relacyjnym

Tworzenie tabeli w SQL:
Create Table student (ID, Name, AvG, photo)

Create Table uczelnia (Nazwa string, Miejsce string, IleM integer)

Kroki w tworzeniu bazy danych

1. Twórz schemat: używa języka DDL
2. Załaduj dane początkowe (mogą być z innego źródła)
3. Wykonywanie zapytań i i modyfikacji.

Odpytywanie bazy ad hoc

Pytania można wymyślać „na bieżąco”. Nie trzeba ich programować. Przykładowe pytania do bazy:

1. Wszyscy studenci z AvG>40% tylko z ZUT i PWSZ.
2. Wszystkie wydziały w Polsce z mniejszą niż 500 liczbą aplikantów.
3. Liceum z najlepszą średnią.

Odpytywanie bazy ad hoc

Niektóre zapytania łatwe do napisani, inne trudne.
Niektóre zapytania łatwe (wykonują się efektywnie) dla DBMS inne trudne.
Modyfikacje też wykonuje się językiem zapytań.

Algebra relacyjna

Formalizmem dla baz danych jest algebra relacyjna. `http://mst.mimuw.edu.pl/lecture.php?lecture=bad&part=Ch2`

SQL — implementacja algebry relacyjnej

Select student.ID From student, Apply Where Student.ID=Apply.ID And AvG>40% and uczelnia='PWSZ'

Do opracowania tej części wykładu wykorzystano

1. Wiadomości z kursu „Databases: DB1 Introduction and Relational Databases" Stanform University
2. Schemat relacyjny: `http://www.paulzaich.com/2012/07/03/blog/ruby-rails/dev-bootcamp-day-17-relational-databases-deconstructing`
3. Oracle Apps Tutorials `https://iamlegand.wordpress.com/page/88/`
4. Grafy `http://neo4j.com/developer/graph-db-vs-rdbms/`