Pomorska Fundacja
Inicjatyw Gospodarczych

Relacyjne i nierelacyjne bazy danych cz.3

Michał Szymański

Co w tej części

Dowiecie się o:

- Jak modyfikować dane w bazie
- Jak projektować struktury bazodanowe
- Jak tworzyć podstawowe struktury bazodanowe
- Typach danych w bazie

SQL – Modyfikacja danych

- INSERT – wstawianie wierszy
- DELETE - usuwanie wierszy
- UPDATE – modyfikacja wierszy

SQL – INSERT

INSERT INTO *tablica* [(*nazwa_kolumny* [, …])] VALUES (*wartosci* [, ….])

INSERT INTO klient (X0, X1, …..) VALUES (V1, V2, …..);

INSERT INTO klient (X0, X1, …..)
  SELECT Y0, Y1... FROM klient WHERE ……..

SQL – DELETE

DELETE FROM *tablica* WHERE *warunek*

DELETE FROM klient WHERE imie='Jan' AND nazwisko='Kowalski'

SQL – UPDATE

UPDATE *tablica* SET { *kolumna* = {*wyrażenie}* | { *kolumna* = {*wyrażenie}* [, ..]) }
WHERE *warunek*

```
UPDATE klient SET imie='Janusz', rok_urodzenia='2001-02-02'
 WHERE   imie='Jan'
```

Projektowanie baz danych...

Nie ma jedynie słusznego projektu schematu bazy danych

Przykład

Imie i nazwisko	Data operacji	Przez	Kwota netto	Kwota brutto	VAT	Tytuł	Autor
Michał Szymański	2014-08-01	Internet	10	12	20	Potop	Henryk Sienkiewicz
Jan Nowak	2014-05-23	Internet	100	120	20	Atlas Świata	Denmart
Michał Szymański	2013-04-02	Sklep	10	12	20	Ogniem i mieczem	Henryk Sienkiewicz
Tomasz Nowak	2012-03-22	Internet	10	12	20	Bajki robotów	Stanisław Lem
Tomasz Nowak	2014-01-05	Sklep	30	36	20	Niezwyciężony	Stanisław Lem
Michał Szymański	2014-05-02	Sklep	10	12	20	Bajki robotów	Stanisław Lem

Anomalie (bazy danych)

Anomalie powstają gdy próbujemy w jednej tabeli (relacji) umieścić zbyt wiele danych

- *Nadmiarowość* - dane powtarzają się w wielu wierszach np. imie i nazwisko
- *Anomalie modyfikacji* - jeśli jest błąd np. przez zmianę nazwiska trzeba zmienić wiele wierszy
- *Anomalie usunięć* - usunięcie jednego zamówienia może spowodować utratę informacji o osobie zamawiającej

Dekompozycja

Dekompozycja doprowadza zbiór relacje do postaci normalnej czyli takiej w której nie tracimy danych i nie ma anomalii

Trzy podstawowe postacie normalne 1NF, 2NF, 3NF

Dekompozycja - 1NF

Każda komórka powinna być elementarna i tabela powinien posiadać klucz główny. W naszym przypadku kolumny '*Autor*' i '*Imie Nazwisko*' nie są komórkami elementarnymi

Imię, Nazwisko, Data operacji , Przez, Tytuł, Autor Imię, Autor Nazwisko

Klucz główny - kolumna albo grupa kolumn pozwalająca na jednoznaczne identyfikacje wiersza w całym zbiorze

Dekompozycja - 2NF

Dane muszą być w 1NF

Każda tabela powinna przechowywać dane dotyczące tylko konkretnej klasy obiektów.

Normalizując do 2NF, wydzielić należy zbiór atrybutów (kolumn) który jest zależny tylko od klucza głównego. Wszystkie atrybuty informacyjne (nie należące do klucza), muszą zawierać informacje o elementach tej konkretnej klasy (encji, tabeli) a nie żadnej innej. Kolumny opisujące inne obiekty, powinny trafić do właściwych encji (tabel) w których te obiekty będziemy przechowywać.

Dekompozycja - 2NF

W naszym przypadku

- *Klient* – Klient_id, Imię, Nazwisko
- *Operacja* – Operacja_id, Przez, Data operacji, Kwota Netto, Kwota Brutto, Vat
- *Ksiazka* – Ksiazka_id, Tytuł
- *Autor* - Autor_id, Imię, Nazwisko

Dekompozycja - 3NF

Kolumna informacyjna nie należąca do klucza nie zależy też od innej kolumny informacyjnej, nie należącej do klucza. Czyli każdy niekluczowy argument jest bezpośrednio zależny tylko od klucza głównego a nie od innej kolumny.

W naszym przypadku:
Kwota Netto, Kwota Brutt i VAT są zależne. Jedną z kolumn można potencjalnie usunąć ale czy warto?

Dekompozycja - wynik

Klient
- Klient_id
- Imię
- Nazwisko

Książka
- Książka_id
- Tytuł
- Autor_id

Operacja
- Klient_id
- Książka_id
- Przez
- Data operacji
- Kwota Netto
- Vat

Autor
- Autor_id
- Imię
- Nazwisko

Ćwiczenie – opis problemu – „System kontrola”

Proszę o dekompozycję następującej mega-relacji przechowującej informacje o kontroli pasażerów na lotnisku.
Na razie nie uwzględniamy pół związanych z relacją miedzy tablicami.

Mega-relacja:

- Imię i nazwisko pasażera
- Data urodzenia pasażera
- Imię i nazwisko strażnika
- Stopień strażnika
- Data zatrudnienia strażnika
- Pensja strażnika
- Ilość dni przepracowanych przez strażnika od daty zatrudnienia
- Nagrody przyznane strażnikowi – to jest lista nazwa nagrody z informacją o dacie przyznania
- Nazwa portu lotniczego gdzie odbywa się kontrola
- Numer stanowiska kontrolujacego – cyfra całkowita
- Wynik kontroli – przepuszczony / zatrzymany
- Data wykonania kontroli (datą + godzina)

Należy przeprowadzić dekompozycje do 3NF przechodząc przez 1NF i 2NF
UWAGA! Wynik waszej pracy będzie przydatny w dalszych ćwiczeniach!

Ćwieczenie- wynik

Pasażer
Imię
Nazwisko

Przyznane nagrody
Nazwa
Data przyznania

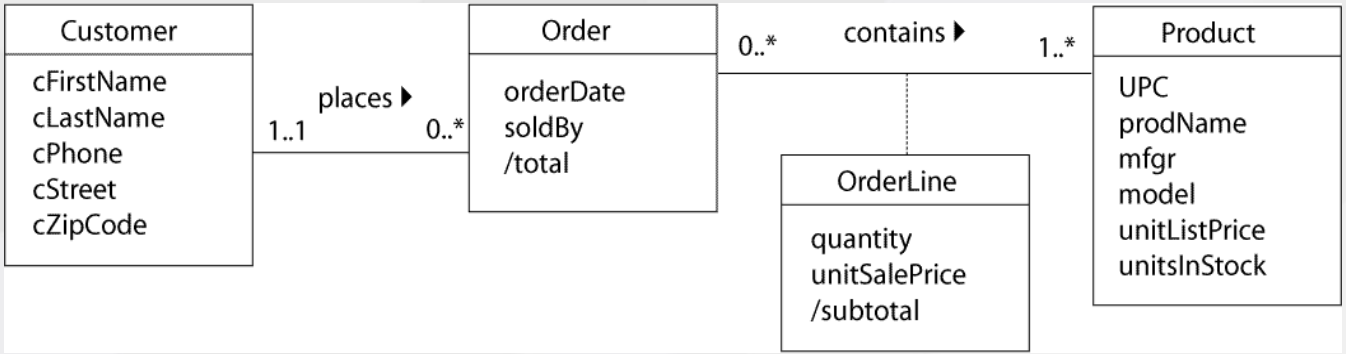
Strażnik
Imię
Nazwisko
Stopień
Data zatrudnienia
Pensja

Kontrola
Nazwa portu
Numer stanowiska
Wynik kontroli
Czas kontroli

Numer stanowiska
Numer stanowiska

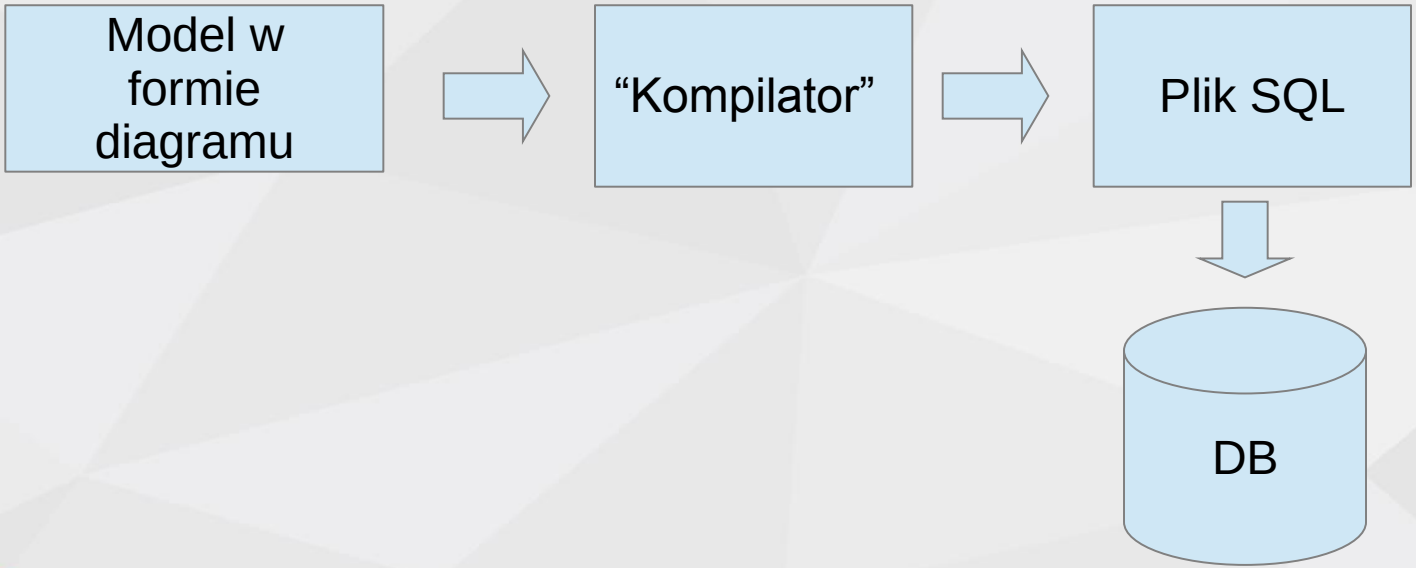
Lotnisko
Nazwa portu

UML - projektowanie



http://www.tomjewett.com/dbdesign/dbdesign.php?page=intro.html

UML – projektowanie, parę słów

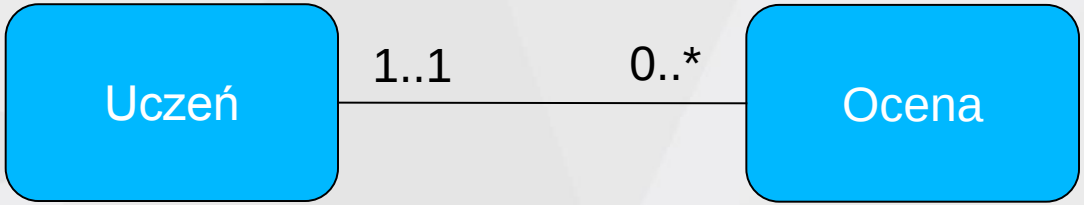


Relacje

Relacje

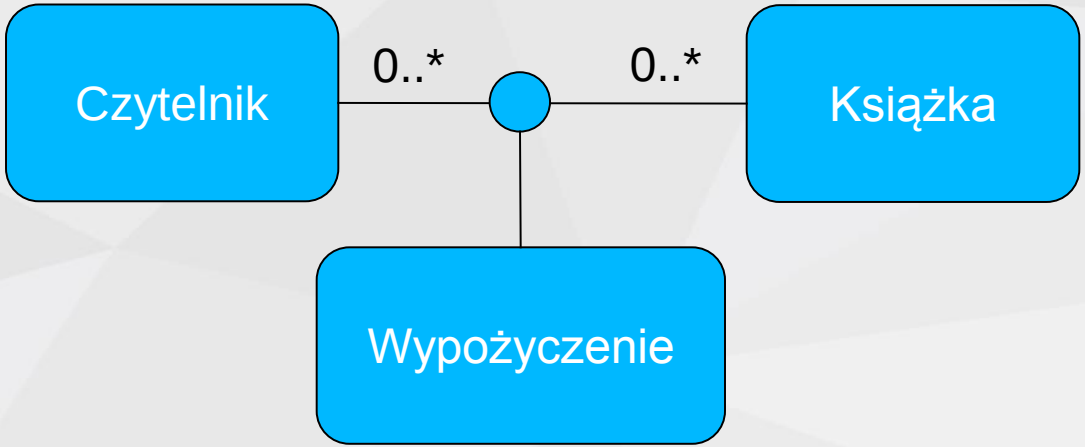
Jeden adres do budynku

Relacje



Uczeń ma wiele ocen

Relacje

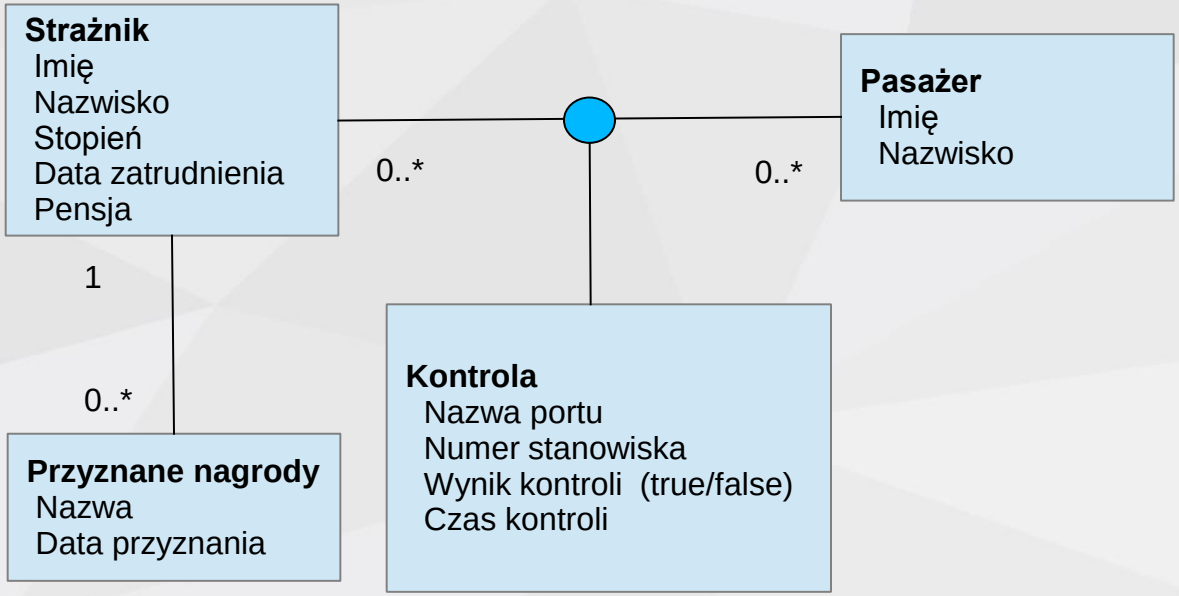


Czytelnik wypożyczył wiele książek,
książka była wypożyczona przez wielu czytelników

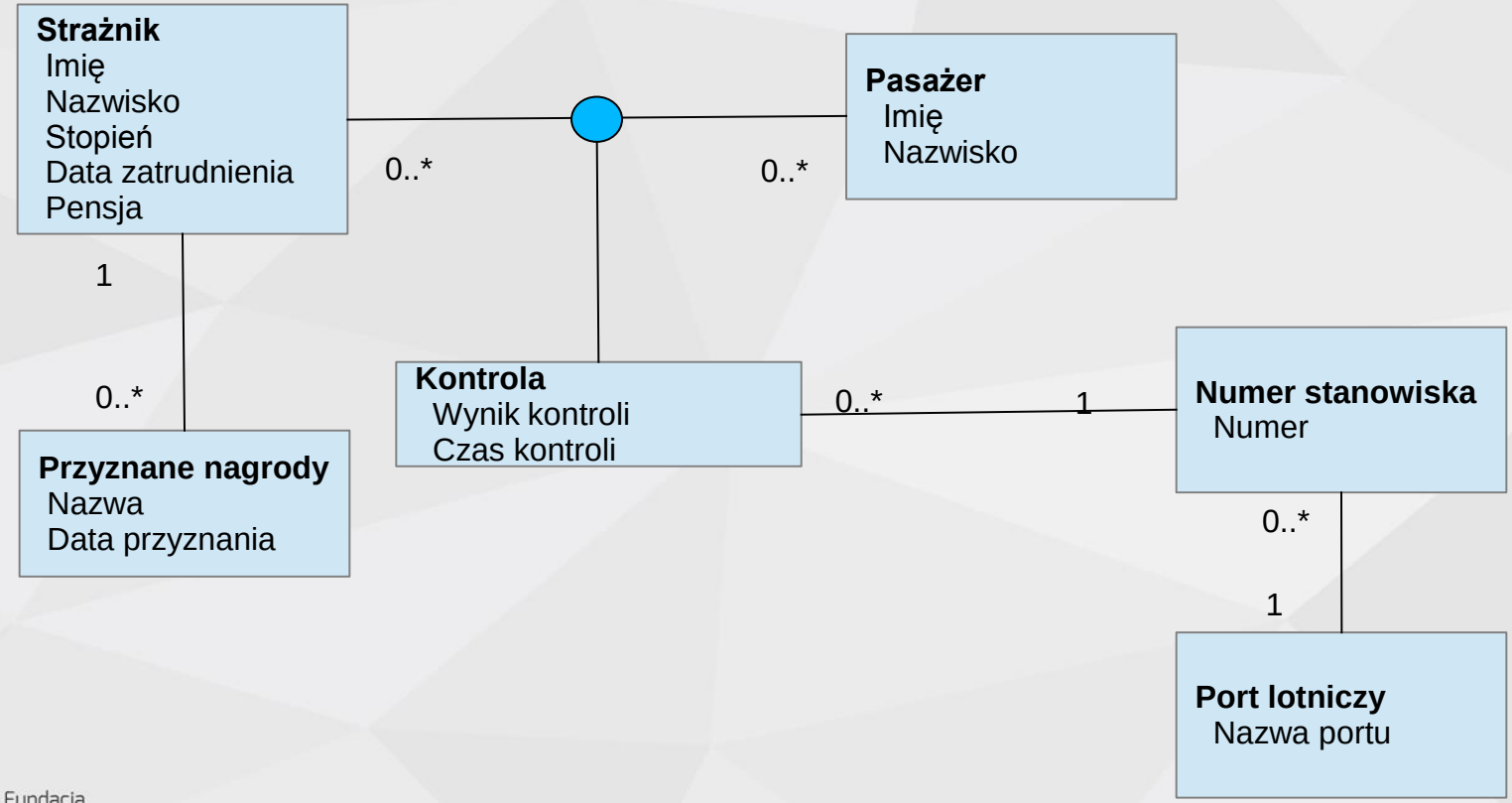
Relacje – ćwiczenie

Proszę zaprojektować relacje dla wcześniej zaprojektowanego schematu dla „System kontrola”

Relacje – ćwiczenie



Relacje – ćwiczenie..rozbudowana wersja



SQL – typy danych

- INT / INT8 – liczby całkowite
- DOUBLE – zmiennoprzecinkowe
- NUMERIC – z podaną precyzją
- INT AUTO_INCREMENT– auto increment

SQL – typy danych, znakowe, binarne..

- VARCHAR – ciągi znakowe
- BYTEA, BLOB – binarne
- BOLEAN - logiczne

SQL – typy danych, znakowe, binarne..

- Date – rozdzielczość 1 dnia
- DATETIME 2004-10-19 10:23:54+02
- Time – czas

SQL – typy danych – mniej popularne

- Point, polygon .. - geometryczne
- INET – sieciowe
- JSON

SQL – Tworzenie tabel - podstawy

```
CREATE TABLE klient_archiwalny (
    klient_id       INT8,
    imie            CHAR(100),
    nazwisko        CHAR(100)
);
```

SQL – Tworzenie tabel - ćwiczenie

Proszę założyć tabelę przechowującą informacje o filmach. Tak żeby posiadała kolumny:

- film_id
- nazwa_filmu
- rezyser
- rok_produkcji – z dokładością do dnia
- typ_filmu
- dostepnosc_na_VOD -- (tak/nie)

Po dodaniu tabeli proszę dodać parę przykładowych wierszy

SQL – Tworzenie tabel - ćwiczenie

Proszę stworzyć plik z definicją tabel dla „System kontroli”
Wszystko ma zostać umieszone w schemacie „Kontrola”.

UWAG! Proszę o nie tworzenie tabel w bazie.

SQL – Ograniczenia ang.constraints

Ograniczenia – zabezpieczają bazę danych przez utratą spójności. Często określa się je jako „Integrity Constraints”

Innymi słowy zabezpieczają nas przez błędnymi danymi w bazie, które mogłyby zostać wprowadzone przez operację DELETE,UPDATE i INSERT

SQL – Ograniczenia - przykłady

- *Płeć* – mężczyzna / kobieta
- *Wiek* – większy niż 0
- *Kolor* – wartość z przygotowanej listy
- Identyfikator użytkownika musi być unikalny

Uwaga! Oprócz zabezpieczenia spójności czasami ograniczenia pełnią rolę ‚informacyjną' dla DB

SQL – Check constraints

Warunek który sprawdza czy dana wartość w kolumnie spełnia określony warunek.

```
CREATE TABLE products (
  product_no  integer,
  name text,
 price numeric CHECK (price > 0) );
```

```
CREATE TABLE products (
  product_no integer,
  name text,
  price numeric CHECK (price > 0),
  discounted_price numeric CHECK (discounted_price > 0),
  CHECK (price > discounted_price) );
```

SQL – NULL constrain

Warunek sprawdzający czy dana wartość jest / nie jest nullem

```
CREATE TABLE products (
    product_no integer NOT NULL,
    name text NOT NULL,
    price numeric NOT NULL CHECK (price > 0) );
```

SQL – UNIQUE / PRIMARY KEY

Gwarancja unikalności (pamiętajcie wartości NULL != NULL)

```
CREATE TABLE example (
        a integer,
        b integer,
        c integer, UNIQUE (a, c) );
```

Klucz główny – to samo co UNIQUE ale nie może zawierać NULLi

```
CREATE TABLE products (
  product_no   integer PRIMARY KEY,
  name         text,
  price        numeric );
```

```
CREATE TABLE example (
    a integer,
    b integer,
    c integer,
    PRIMARY KEY (a, c) );
```

SQL – klucze obce

Klucz obcy - definiuje relacje pomiędzy dwoma tabelami

```
CREATE TABLE ksiazka(
  ksiazka_id    INT8          PRIMARY KEY,
  rodzaj        VARCHAR(00)   NOT NULL,
  ……
  CONSTRAINT ksiazka_rodzaj_fkey FOREIGN KEY (rodzaj)     REFERENCES   rodzaj_ksiazki (rodzaj )
```

SQL – klucze obce

```
CREATE TABLE recenzja
(
  recenzja_id bigserial                NOT NULL,
  tytul_ksiazki                        CHAR(200),
  autor_imie_ksiazki                   CHAR(100),
  autor_nazwisko_ksiazki               CHAR(100),
  ..............
  CONSTRAINT recenzja_ksiazka_id_fkey FOREIGN KEY (tytul_ksiazki,autor_imie_ksiazki,autor_nazwisko_ksiazki  )
     REFERENCES ksiazka (tytul, autor_imie, autor_nazwisko)
)
```

SQL – klucze obce a kasowanie

- Zablokowanie operacji
- Ustawienie na null
- Skasowanie kaskadowe

Id	X1	X2	X3	..
1				
2				
3				
4				

Id	Id_X1	Y2	Y3	..
	2			
	3			
	3			

SQL – klucze obce a kasowanie

```
CREATE TABLE zlecenie
(
  numer zlecenie     INT8 PRIMARY KEY,
  klient_id              INT8,
 CONSTRAINT klient_id_fkey FOREIGN KEY (klient_id)     REFERENCES   klient(id)   ON DELETE  RESTRICT
)

CREATE TABLE zlecenie
(
  numer zlecenie     INT8 PRIMARY KEY,
  klient_id              INT8,
 CONSTRAINT klient_id_fkey FOREIGN KEY (klient_id)     REFERENCES   klient(id)   ON DELETE  CASCADE
)

CREATE TABLE zlecenie
(
  numer zlecenie     INT8 PRIMARY KEY,
  klient_id              INT8,
 CONSTRAINT klient_id_fkey FOREIGN KEY (klient_id)     REFERENCES   klient(id)   ON DELETE  SET NULL
)
```

SQL – Tworzenie tabel - ćwiczenie

Proszę utworzyć dla „System kontrola”:

- Klucze główne
- Klucze obce
- Ograniczenia dla pól w tabelach
- Klucze unikalne