

Bazy danych 2024

19 marca 2024

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12 Ełk, Kwiatowa 102	64,2	$\Leftarrow t_1$
NULL NULL ...		64,2	$\Leftarrow t_2$

- $t_1.\text{PESEL} = t_2.\text{PESEL}$

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- t_1 .PESEL = t_2 .PESEL UNKNOWN!!!

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL NULL ...	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$

- t_1 .PESEL = t_2 .PESEL UNKNOWN!!!
- t_1 .PESEL = t_1 .Adres

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL NULL ...	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$

- t_1 .PESEL = t_2 .PESEL UNKNOWN!!!
- t_1 .PESEL = t_1 .Adres UNKNOWN!!!

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1.$ PESEL = $t_2.$ PESEL UNKNOWN!!!
- $t_1.$ PESEL = $t_1.$ Adres UNKNOWN!!!
- $t_1.$ Metraż = $t_2.$ Metraż

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- t_1 .PESEL = t_2 .PESEL UNKNOWN!!!
- t_1 .PESEL = t_1 .Adres UNKNOWN!!!
- t_1 .Metraż = t_2 .Metraż TRUE

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- $t_1.$ PESEL = $t_2.$ PESEL UNKNOWN!!!
- $t_1.$ PESEL = $t_1.$ Adres UNKNOWN!!!
- $t_1.$ Metraż = $t_2.$ Metraż TRUE
- $t_1.$ Adres = $t_2.$ Adres

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- t_1 .PESEL = t_2 .PESEL UNKNOWN!!!
- t_1 .PESEL = t_1 .Adres UNKNOWN!!!
- t_1 .Metraż = t_2 .Metraż TRUE
- t_1 .Adres = t_2 .Adres FALSE

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- t_1 .PESEL = t_2 .PESEL UNKNOWN!!!
- t_1 .PESEL = t_1 .Adres UNKNOWN!!!
- t_1 .Metraż = t_2 .Metraż TRUE
- t_1 .Adres = t_2 .Adres FALSE
- t_1 .PESEL = NULL

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- t_1 .PESEL = t_2 .PESEL UNKNOWN!!!
- t_1 .PESEL = t_1 .Adres UNKNOWN!!!
- t_1 .Metraż = t_2 .Metraż TRUE
- t_1 .Adres = t_2 .Adres FALSE
- t_1 .PESEL = NULL UNKNOWN!!!

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- t_1 .PESEL = t_2 .PESEL UNKNOWN!!!
- t_1 .PESEL = t_1 .Adres UNKNOWN!!!
- t_1 .Metraż = t_2 .Metraż TRUE
- t_1 .Adres = t_2 .Adres FALSE
- t_1 .PESEL = NULL UNKNOWN!!!
- t_1 .PESEL = ' '

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- t_1 .PESEL = t_2 .PESEL UNKNOWN!!!
- t_1 .PESEL = t_1 .Adres UNKNOWN!!!
- t_1 .Metraż = t_2 .Metraż TRUE
- t_1 .Adres = t_2 .Adres FALSE
- t_1 .PESEL = NULL UNKNOWN!!!
- t_1 .PESEL = ' ' UNKNOWN!!!

Wartość pusta (NULL)

PESEL : char(11)	Adres : varchar(50)	Metraż : real	
...			
NULL	Poznań, Szeroka 10/12	64,2	$\Leftarrow t_1$
NULL	Ełk, Kwiatowa 102	64,2	$\Leftarrow t_2$
...			

- t_1 .PESEL = t_2 .PESEL UNKNOWN!!!
- t_1 .PESEL = t_1 .Adres UNKNOWN!!!
- t_1 .Metraż = t_2 .Metraż TRUE
- t_1 .Adres = t_2 .Adres FALSE
- t_1 .PESEL = NULL UNKNOWN!!!
- t_1 .PESEL = ' ' UNKNOWN!!!
- t_1 .PESEL IS NULL TRUE
- t_1 .Adres IS NOT NULL TRUE

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)
- ❸ dana wartość jest nieobecna *by design* - zły projekt bazy (np. numer indeksu dla nie-studenta, poprzednik pierwszego elementu);

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)
- ❸ dana wartość jest nieobecna *by design* - zły projekt bazy (np. numer indeksu dla nie-studenta, poprzednik pierwszego elementu);
- W scenariuszu pierwszym - NULLe reprezentujemy jako zmienną z danej dziedziny.

id	nazwisko	ocena
1	Kowalski	5
2	Nowak	x
3	y	4.5
v	w	z

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)
- ❸ dana wartość jest nieobecna *by design* - zły projekt bazy (np. numer indeksu dla nie-studenta, poprzednik pierwszego elementu);
- W scenariuszu pierwszym - NULLe reprezentujemy jako zmienną z danej dziedziny.

id	nazwisko	ocena
1	Kowalski	5
2	Nowak	x
3	y	4.5
v	w	z

- zły projekt bazy (np. 10 kolumn na numery telefonów) vs. *bazy semistrukturalne* → JSON

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)
- ❸ dana wartość jest nieobecna *by design* - zły projekt bazy (np. numer indeksu dla nie-studenta, poprzednik pierwszego elementu);
- W scenariuszu pierwszym - NULLe reprezentujemy jako zmienną z danej dziedziny.

id	nazwisko	ocena
1	Kowalski	5
2	Nowak	x
3	y	4.5
v	w	z

- zły projekt bazy (np. 10 kolumn na numery telefonów) vs. *bazy semistrukturalne* → JSON
- różne rzeczy się komplikują - logika trójwartościowa!

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)
- ❸ dana wartość jest nieobecna *by design* - zły projekt bazy (np. numer indeksu dla nie-studenta, poprzednik pierwszego elementu);
- W scenariuszu pierwszym - NULLe reprezentujemy jako zmienną z danej dziedziny.

id	nazwisko	ocena
1	Kowalski	5
2	Nowak	x
3	y	4.5
v	w	z

- zły projekt bazy (np. 10 kolumn na numery telefonów) vs. *bazy semistrukturalne* → JSON
- różne rzeczy się komplikują - logika trójwartościowa!
- czy dla UNKNOWN warunek selekcji jest spełniony czy nie? a jego zaprzeczenie(!!!)?

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)
- ❸ dana wartość jest nieobecna *by design* - zły projekt bazy (np. numer indeksu dla nie-studenta, poprzednik pierwszego elementu);
- W scenariuszu pierwszym - NULLe reprezentujemy jako zmienną z danej dziedziny.

id	nazwisko	ocena
1	Kowalski	5
2	Nowak	x
3	y	4.5
v	w	z

- zły projekt bazy (np. 10 kolumn na numery telefonów) vs. *bazy semistrukturalne* → JSON
- różne rzeczy się komplikują - logika trójwartościowa!
- czy dla UNKNOWN warunek selekcji jest spełniony czy nie? a jego zaprzeczenie(!!!)?
- a więc (np. wynik_matury \leq 100) jest spełniony ? a jego zaprzeczenie(!!!)?

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)
- ❸ dana wartość jest nieobecna *by design* - zły projekt bazy (np. numer indeksu dla nie-studenta, poprzednik pierwszego elementu);
- W scenariuszu pierwszym - NULLe reprezentujemy jako zmienną z danej dziedziny.

id	nazwisko	ocena
1	Kowalski	5
2	Nowak	x
3	y	4.5
v	w	z

- zły projekt bazy (np. 10 kolumn na numery telefonów) vs. *bazy semistrukturalne* → JSON
- różne rzeczy się komplikują - logika trójwartościowa!
- czy dla UNKNOWN warunek selekcji jest spełniony czy nie? a jego zaprzeczenie(!!!)?
- a więc (np. wynik_matury \leq 100) jest spełniony ? a jego zaprzeczenie(!!!)?
- możesz coś komuś sprzedać bez zapłaty, nie przechwycisz wrogiej rakiety - szczegóły w części o SQL

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)
- ❸ dana wartość jest nieobecna *by design* - zły projekt bazy (np. numer indeksu dla nie-studenta, poprzednik pierwszego elementu);
- W scenariuszu pierwszym - NULLe reprezentujemy jako zmienną z danej dziedziny.

id	nazwisko	ocena
1	Kowalski	5
2	Nowak	x
3	y	4.5
v	w	z

- zły projekt bazy (np. 10 kolumn na numery telefonów) vs. *bazy semistrukturalne* → JSON
- różne rzeczy się komplikują - logika trójwartościowa!
- czy dla UNKNOWN warunek selekcji jest spełniony czy nie? a jego zaprzeczenie(!!!)?
- a więc (np. wynik_matury \leq 100) jest spełniony ? a jego zaprzeczenie(!!!)?
- możesz coś komuś sprzedać bez zapłaty, nie przechwycisz wrogiej rakiety - szczegóły w części o SQL
- wartości NULL i złączenia

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)
- ❸ dana wartość jest nieobecna *by design* - zły projekt bazy (np. numer indeksu dla nie-studenta, poprzednik pierwszego elementu);
- W scenariuszu pierwszym - NULLe reprezentujemy jako zmienną z danej dziedziny.

id	nazwisko	ocena
1	Kowalski	5
2	Nowak	x
3	y	4.5
v	w	z

- zły projekt bazy (np. 10 kolumn na numery telefonów) vs. *bazy semistrukturalne* → JSON
- różne rzeczy się komplikują - logika trójwartościowa!
- czy dla UNKNOWN warunek selekcji jest spełniony czy nie? a jego zaprzeczenie(!!!)?
- a więc (np. wynik_matury \leq 100) jest spełniony ? a jego zaprzeczenie(!!!)?
- możesz coś komuś sprzedać bez zapłaty, nie przechwycisz wrogiej rakiety - szczegóły w części o SQL
- wartości NULL i złączenia → złączenia zewnętrzne

Wartość NULL

Może oznaczać, że

- ❶ nie znamy danej wartości (np. student już dostał ocenę ale nie wiadomo jaką, awaria czujnika, itp.)
- ❷ dana wartość w ogóle nie jest jeszcze zdefiniowana (np. egzamin się jeszcze nie odbył, data kasacji, itp.)
- ❸ dana wartość jest nieobecna *by design* - zły projekt bazy (np. numer indeksu dla nie-studenta, poprzednik pierwszego elementu);
- W scenariuszu pierwszym - NULLe reprezentujemy jako zmienną z danej dziedziny.

id	nazwisko	ocena
1	Kowalski	5
2	Nowak	x
3	y	4.5
v	w	z

- zły projekt bazy (np. 10 kolumn na numery telefonów) vs. *bazy semistrukturalne* → JSON
- różne rzeczy się komplikują - logika trójwartościowa!
- czy dla UNKNOWN warunek selekcji jest spełniony czy nie? a jego zaprzeczenie(!!!)?
- a więc (np. wynik_matury \leq 100) jest spełniony ? a jego zaprzeczenie(!!!)?
- możesz coś komuś sprzedać bez zapłaty, nie przechwycisz wrogiej rakiety - szczegóły w części o SQL
- wartości NULL i złączenia → złączenia zewnętrzne
- COUNT(*) vs COUNT(actors.id) vs. COUNT(actors.gender)

Dobry projekt bazy danych

NULL-e — krotki powinny pasować do schematu tabeli,

Redundancja — informacja nie powinna być zapisywana wielokrotnie,

Kontrola więzów — sprawdzanie własności klucza, unikalności i innych więzów powinno być łatwe,

Obliczanie złączeń — jest trudne, więc nie należy rozdrabniać zbyt wiele bazy.

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)
złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu)

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu) **1NF**

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu) **1NF**

Codd: *to permit data to be queried and manipulated using a "universal data sub-language"*

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu) **1NF**

Codd: *to permit data to be queried and manipulated using a "universal data sub-language"*

anomia UPDATE — zmiana biura prowadzącego → zmiana w wielu miejscach (redundacja, niejednoznaczność)

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu) **1NF**

Codd: *to permit data to be queried and manipulated using a "universal data sub-language"*

anomia UPDATE — zmiana biura prowadzącego → zmiana w wielu miejscach (redundacja, niejednoznaczność)

anomia INSERT — jak dodać nowego prowadzącego (bez grup)?

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu) **1NF**

Codd: *to permit data to be queried and manipulated using a "universal data sub-language"*

anomia UPDATE — zmiana biura prowadzącego → zmiana w wielu miejscach (redundacja, niejednoznaczność)

anomia INSERT — jak dodać nowego prowadzącego (bez grup)? Null?

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu) **1NF**

Codd: *to permit data to be queried and manipulated using a "universal data sub-language"*

anomia UPDATE — zmiana biura prowadzącego → zmiana w wielu miejscach (redundacja, niejednoznaczność)

anomia INSERT — jak dodać nowego prowadzącego (bez grup)? Null? jaki klucz główny?

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu) **1NF**

Codd: *to permit data to be queried and manipulated using a "universal data sub-language"*

anomia UPDATE — zmiana biura prowadzącego → zmiana w wielu miejscach (redundacja, niejednoznaczność)

anomia INSERT — jak dodać nowego prowadzącego (bez grup)? Null? jaki klucz główny?

anomia DELETE — co po usunięciu ostatniej grupy prowadzącego?

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu) **1NF**

Codd: *to permit data to be queried and manipulated using a "universal data sub-language"*

anomia UPDATE — zmiana biura prowadzącego → zmiana w wielu miejscach (redundacja, niejednoznaczność)

anomia INSERT — jak dodać nowego prowadzącego (bez grup)? Null? jaki klucz główny?

anomia DELETE — co po usunięciu ostatniej grupy prowadzącego?

zalety ?

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu) **1NF**

Codd: *to permit data to be queried and manipulated using a "universal data sub-language"*

anomia UPDATE — zmiana biura prowadzącego → zmiana w wielu miejscach (redundacja, niejednoznaczność)

anomia INSERT — jak dodać nowego prowadzącego (bez grup)? Null? jaki klucz główny?

anomia DELETE — co po usunięciu ostatniej grupy prowadzącego?

zalety ? nie ma joinów!!!

Dobry projekt bazy danych - anomalie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

złożone typy — lista_zapisanych_w_json → Zapisy(ids, idg, data_zapisu) **1NF**

Codd: *to permit data to be queried and manipulated using a "universal data sub-language"*

anomia UPDATE — zmiana biura prowadzącego → zmiana w wielu miejscach (redundacja, niejednoznaczność)

anomia INSERT — jak dodać nowego prowadzącego (bez grup)? Null? jaki klucz główny?

anomia DELETE — co po usunięciu ostatniej grupy prowadzącego?

zalety ? nie ma joinów!!! kiedy to jest dobrze, a kiedy źle?

Dobry projekt bazy danych - rozwiązanie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)

Dobry projekt bazy danych - rozwiązanie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)
Propozycje?

Dobry projekt bazy danych - rozwiązanie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)
Propozycje?

- Grupa(idg, id_przed, id_prow, inne_dane_grupy)
- Prowadzący(id_prow, prow_nazwisko, prow_biuro)
- Zapisy(id_stud, idg, data_zapisu)
- Przedmioty(id_przed, opis_przedmiotu)

Dobry projekt bazy danych - rozwiązanie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)
Propozycje?

- Grupa(idg, id_przed, id_prow, inne_dane_grupy)
- Prowadzący(id_prow, prow_nazwisko, prow_biuro)
- Zapisy(id_stud, idg, data_zapisu)
- Przedmioty(id_przed, opis_przedmiotu)

Trzeba uważać aby dobrze ustalić co jest atrybutem czego!

Dobry projekt bazy danych - rozwiązanie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)
Propozycje?

- Grupa(idg, id_przed, id_prow, inne_dane_grupy)
- Prowadzący(id_prow, prow_nazwisko, prow_biuro)
- Zapisy(id_stud, idg, data_zapisu)
- Przedmioty(id_przed, opis_przedmiotu)

Trzeba uważać aby dobrze ustalić co jest atrybutem czego!

Przedmioty(id_przed, opis_przedmiotu, konsultacje_biuro) ???

Dobry projekt bazy danych - rozwiązanie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)
Propozycje?

- Grupa(idg, id_przed, id_prow, inne_dane_grupy)
- Prowadzący(id_prow, prow_nazwisko, prow_biuro)
- Zapisy(id_stud, idg, data_zapisu)
- Przedmioty(id_przed, opis_przedmiotu)

Trzeba uważać aby dobrze ustalić co jest atrybutem czego!

Przedmioty(id_przed, opis_przedmiotu, konsultacje_biuro) ???

Grupa(idg, id_przed, id_prow, inne_dane_grupy, konsultacje_biuro) ???

Dobry projekt bazy danych - rozwiązanie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)
Propozycje?

- Grupa(idg, id_przed, id_prow, inne_dane_grupy)
- Prowadzący(id_prow, prow_nazwisko, prow_biuro)
- Zapisy(id_stud, idg, data_zapisu)
- Przedmioty(id_przed, opis_przedmiotu)

Trzeba uważać aby dobrze ustalić co jest atrybutem czego!

Przedmioty(id_przed, opis_przedmiotu, konsultacje_biuro) ???

Grupa(idg, id_przed, id_prow, inne_dane_grupy, konsultacje_biuro) ???

Co jak dwóch prowadzących prowadzi jedną grupę?

Dobry projekt bazy danych - rozwiązanie

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro, lista_zapisanych_w_json)
Propozycje?

- Grupa(idg, id_przed, id_prow, inne_dane_grupy)
- Prowadzący(id_prow, prow_nazwisko, prow_biuro)
- Zapisy(id_stud, idg, data_zapisu)
- Przedmioty(id_przed, opis_przedmiotu)

Trzeba uważać aby dobrze ustalić co jest atrybutem czego!

Przedmioty(id_przed, opis_przedmiotu, konsultacje_biuro) ???

Grupa(idg, id_przed, id_prow, inne_dane_grupy, konsultacje_biuro) ???

Co jak dwóch prowadzących prowadzi jedną grupę? Albo co gdy zmienimy pokój prowadzącemu?

Normalizacja

- Rozbijamy dane tak aby nie było redundancji

Normalizacja

- Rozbijamy dane tak aby nie było redundancji
- Pilnując różnych zależności

Normalizacja

- Rozbijamy dane tak aby nie było redundancji
- Pilnując różnych zależności
- Najlepiej rozrysować to sobie z użyciem diagramów ER (będzie więcej o tym!)

Normalizacja

- Rozbijamy dane tak aby nie było redundancji
- Pilnując różnych zależności
- Najlepiej rozrysować to sobie z użyciem diagramów ER (będzie więcej o tym!)
- Czasem lepiej odpuścić aby nie robić tylu joinów (świadoma decyzja!)

Skąd się biorą redundancje?

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro,

- Kluczem jest **idg**

Skąd się biorą redundancje?

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro,

- Kluczem jest **idg**
- Każda grupa z danego przedmiotu ma powtórzone dane przedmiotu

Skąd się biorą redundancje?

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro,

- Kluczem jest **idg**
- Każda grupa z danego przedmiotu ma powtórzone dane przedmiotu
- Każdy prowadzący ma powtórzone dane w każdym swoim przedmiocie

Skąd się biorą redundancje?

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro,

- Kluczem jest **idg**
- Każda grupa z danego przedmiotu ma powtórzone dane przedmiotu
- Każdy prowadzący ma powtórzone dane w każdym swoim przedmiocie
- Podzbiór kolumn jest **determinowany** przez coś co nie jest kluczem!

Skąd się biorą redundancje?

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro,

- Kluczem jest **idg**
- Każda grupa z danego przedmiotu ma powtórzone dane przedmiotu
- Każdy prowadzący ma powtórzone dane w każdym swoim przedmiocie
- Podzbiór kolumn jest **determinowany** przez coś co nie jest kluczem!
- nazwa_przedmiotu → opis_przedmiotu

Skąd się biorą redundancje?

GR(idg, nazwa_przedmiotu, opis_przedmiotu, prow_nazwisko, prow_biuro,

- Kluczem jest **idg**
- Każda grupa z danego przedmiotu ma powtórzone dane przedmiotu
- Każdy prowadzący ma powtórzone dane w każdym swoim przedmiocie
- Podzbiór kolumn jest **determinowany** przez coś co nie jest kluczem!
- nazwa_przedmiotu → opis_przedmiotu
- prow_nazwisko → prow_biuro

Definicja i przykład

Definition (Zależność funkcyjna)

Dla relacji $R = A_1 A_2 \dots A_k$ oraz zbiorów jej atrybutów $\alpha, \beta \subseteq \{A_1 A_2 \dots A_k\}$ zachodzi zależność funkcyjna $\alpha \rightarrow \beta$, jeżeli dla każdego stanu r relacji R zachodzi:

$$(\forall t_1, t_2 \in r)((t_1.\alpha = t_2.\alpha) \Rightarrow (t_1.\beta = t_2.\beta))$$

Definicja i przykład

Definition (Zależność funkcyjna)

Dla relacji $R = A_1 A_2 \dots A_k$ oraz zbiorów jej atrybutów $\alpha, \beta \subseteq \{A_1 A_2 \dots A_k\}$ zachodzi zależność funkcyjna $\alpha \rightarrow \beta$, jeżeli dla każdego stanu r relacji R zachodzi:

$$(\forall t_1, t_2 \in r)((t_1.\alpha = t_2.\alpha) \Rightarrow (t_1.\beta = t_2.\beta))$$

Kolumny α determinują kolumny β ! Tyle!

Definicja i przykład

Definition (Zależność funkcyjna)

Dla relacji $R = A_1 A_2 \dots A_k$ oraz zbiorów jej atrybutów $\alpha, \beta \subseteq \{A_1 A_2 \dots A_k\}$ zachodzi zależność funkcyjna $\alpha \rightarrow \beta$, jeżeli dla każdego stanu r relacji R zachodzi:

$$(\forall t_1, t_2 \in r)((t_1.\alpha = t_2.\alpha) \Rightarrow (t_1.\beta = t_2.\beta))$$

Kolumny α determinują kolumny β ! Tyle!

W relacjach $PR(ido, tytuł, nazwisko, adres)$, $ST(ido, indeks, nazwisko, adres)$ i $GR(idg, idk, idp, limit)$, $TS(idg, termin, sala)$ zachodzą zależności

Definicja i przykład

Definition (Zależność funkcyjna)

Dla relacji $R = A_1 A_2 \dots A_k$ oraz zbiorów jej atrybutów $\alpha, \beta \subseteq \{A_1 A_2 \dots A_k\}$ zachodzi zależność funkcyjna $\alpha \rightarrow \beta$, jeżeli dla każdego stanu r relacji R zachodzi:

$$(\forall t_1, t_2 \in r)((t_1.\alpha = t_2.\alpha) \Rightarrow (t_1.\beta = t_2.\beta))$$

Kolumny α determinują kolumny β ! Tyle!

W relacjach $PR(ido, tytuł, nazwisko, adres)$, $ST(ido, indeks, nazwisko, adres)$ i $GR(idg, idk, idp, limit)$, $TS(idg, termin, sala)$ zachodzą zależności

- w PR : $ido \rightarrow nazwisko, adres, tytuł$,

Definicja i przykład

Definition (Zależność funkcyjna)

Dla relacji $R = A_1 A_2 \dots A_k$ oraz zbiorów jej atrybutów $\alpha, \beta \subseteq \{A_1 A_2 \dots A_k\}$ zachodzi zależność funkcyjna $\alpha \rightarrow \beta$, jeżeli dla każdego stanu r relacji R zachodzi:

$$(\forall t_1, t_2 \in r)((t_1.\alpha = t_2.\alpha) \Rightarrow (t_1.\beta = t_2.\beta))$$

Kolumny α determinują kolumny β ! Tyle!

W relacjach $PR(ido, tytuł, nazwisko, adres)$, $ST(ido, indeks, nazwisko, adres)$ i $GR(idg, idk, idp, limit)$, $TS(idg, termin, sala)$ zachodzą zależności

- w PR : $ido \rightarrow nazwisko, adres, tytuł$,
- w ST : $ido \rightarrow nazwisko, adres, indeks$ oraz $indeks \rightarrow ido, nazwisko, adres$;

Definicja i przykład

Definition (Zależność funkcyjna)

Dla relacji $R = A_1 A_2 \dots A_k$ oraz zbiorów jej atrybutów $\alpha, \beta \subseteq \{A_1 A_2 \dots A_k\}$ zachodzi zależność funkcyjna $\alpha \rightarrow \beta$, jeżeli dla każdego stanu r relacji R zachodzi:

$$(\forall t_1, t_2 \in r)((t_1.\alpha = t_2.\alpha) \Rightarrow (t_1.\beta = t_2.\beta))$$

Kolumny α determinują kolumny β ! Tyle!

W relacjach $PR(ido, tytuł, nazwisko, adres)$, $ST(ido, indeks, nazwisko, adres)$ i $GR(idg, idk, idp, limit)$, $TS(idg, termin, sala)$ zachodzą zależności

- w PR : $ido \rightarrow nazwisko, adres, tytuł$,
- w ST : $ido \rightarrow nazwisko, adres, indeks$ oraz $indeks \rightarrow ido, nazwisko, adres$;
- w GR : $idg \rightarrow idk, idp, limit$

Definicja i przykład

Definition (Zależność funkcyjna)

Dla relacji $R = A_1 A_2 \dots A_k$ oraz zbiorów jej atrybutów $\alpha, \beta \subseteq \{A_1 A_2 \dots A_k\}$ zachodzi zależność funkcyjna $\alpha \rightarrow \beta$, jeżeli dla każdego stanu r relacji R zachodzi:

$$(\forall t_1, t_2 \in r)((t_1.\alpha = t_2.\alpha) \Rightarrow (t_1.\beta = t_2.\beta))$$

Kolumny α determinują kolumny β ! Tyle!

W relacjach $PR(ido, tytuł, nazwisko, adres)$, $ST(ido, indeks, nazwisko, adres)$ i $GR(idg, idk, idp, limit)$, $TS(idg, termin, sala)$ zachodzą zależności

- w PR : $ido \rightarrow nazwisko, adres, tytuł$,
- w ST : $ido \rightarrow nazwisko, adres, indeks$ oraz $indeks \rightarrow ido, nazwisko, adres$;
- w GR : $idg \rightarrow idk, idp, limit$
- w TS : $termin, sala \rightarrow idg$

Zależności funkcyjne

Spostrzeżenie: Jeśli K jest kluczem R , to $K \rightarrow R$.

Zależności funkcyjne

Spostrzeżenie: Jeśli K jest kluczem R , to $K \rightarrow R$.

Spostrzeżenie: Zależność funkcyjna to własność rzeczywistości, nie danych!

Zależności funkcyjne

Spostrzeżenie: Jeśli K jest kluczem R , to $K \rightarrow R$.

Spostrzeżenie: Zależność funkcyjna to własność rzeczywistości, nie danych!

Spostrzeżenie: Na podstawie danych można tylko sprawdzić czy zależność jest spełniona. Nie da się ustalić czy zależność zachodzi!

Przykład

- Mafia(Miasto, Gang, Proceder)

Przykład

- Mafia(Miasto, Gang, Proceder)
- ξ_1 : Gang \rightarrow Miasto oraz ξ_2 : Miasto, Proceder \rightarrow Gang.

Przykład

- Mafia(Miasto, Gang, Proceder)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$ oraz $\xi_2: \text{Miasto, Proceder} \rightarrow \text{Gang}$.
- Co można wywnioskować o organizacji mafii wiedząc, że zachodzą ξ_1 i ξ_2 ?

Przykład

- Mafia(Miasto, Gang, Proceder)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$ oraz $\xi_2: \text{Miasto, Proceder} \rightarrow \text{Gang}$.
- Co można wywnioskować o organizacji mafii wiedząc, że zachodzą ξ_1 i ξ_2 ?
- Jakie są klucze relacji Mafia?

Przykład

- Mafia(Miasto, Gang, Proceder)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$ oraz $\xi_2: \text{Miasto, Proceder} \rightarrow \text{Gang}$.
- Co można wywnioskować o organizacji mafii wiedząc, że zachodzą ξ_1 i ξ_2 ?
- Jakie są klucze relacji Mafia?
- Anomalie?

Przykład

- Mafia(Miasto, Gang, Proceder)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$ oraz $\xi_2: \text{Miasto, Proceder} \rightarrow \text{Gang}$.
- Co można wywnioskować o organizacji mafii wiedząc, że zachodzą ξ_1 i ξ_2 ?
- Jakie są klucze relacji Mafia?
- Anomalie?
- Podaj odwracalny rozkład relacji Mafia do postaci BCNF(!?).

Przykład

- Mafia(Miasto, Gang, Proceder)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$ oraz $\xi_2: \text{Miasto, Proceder} \rightarrow \text{Gang}$.
- Co można wywnioskować o organizacji mafii wiedząc, że zachodzą ξ_1 i ξ_2 ?
- Jakie są klucze relacji Mafia?
- Anomalie?
- Podaj odwracalny rozkład relacji Mafia do postaci BCNF(!?).
- Rozkład?

Przykład

- Mafia(Miasto, Gang, Proceder)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$ oraz $\xi_2: \text{Miasto, Proceder} \rightarrow \text{Gang}$.
- Co można wywnioskować o organizacji mafii wiedząc, że zachodzą ξ_1 i ξ_2 ?
- Jakie są klucze relacji Mafia?
- Anomalie?
- Podaj odwracalny rozkład relacji Mafia do postaci BCNF(!?).
- Rozkład?
- Odwracalny?

Przykład

- Mafia(Miasto, Gang, Proceder)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$ oraz $\xi_2: \text{Miasto, Proceder} \rightarrow \text{Gang}$.
- Co można wywnioskować o organizacji mafii wiedząc, że zachodzą ξ_1 i ξ_2 ?
- Jakie są klucze relacji Mafia?
- Anomalie?
- Podaj odwracalny rozkład relacji Mafia do postaci BCNF(!?).
- Rozkład?
- Odwracalny?
- BCNF?

Przykład

- Mafia(Miasto, Gang, Proceder)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$ oraz $\xi_2: \text{Miasto, Proceder} \rightarrow \text{Gang}$.
- Co można wywnioskować o organizacji mafii wiedząc, że zachodzą ξ_1 i ξ_2 ?
- Jakie są klucze relacji Mafia?
- Anomalie?
- Podaj odwracalny rozkład relacji Mafia do postaci BCNF(!?).
- Rozkład?
- Odwracalny?
- BCNF?
- Czy rozkład ten zachowuje zależności?

Przykład

- Mafia(Miasto, Gang, Proceder)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$ oraz $\xi_2: \text{Miasto, Proceder} \rightarrow \text{Gang}$.
- Co można wywnioskować o organizacji mafii wiedząc, że zachodzą ξ_1 i ξ_2 ?
- Jakie są klucze relacji Mafia?
- Anomalie?
- Podaj odwracalny rozkład relacji Mafia do postaci BCNF(!?).
- Rozkład?
- Odwracalny?
- BCNF?
- Czy rozkład ten zachowuje zależności?
- Jakie rodzi to problemy?

Przykład

- Mafia(Miasto, Gang, Proceder)
- ξ_1 : Gang \rightarrow Miasto oraz ξ_2 : Miasto, Proceder \rightarrow Gang.
- Co można wywnioskować o organizacji mafii wiedząc, że zachodzą ξ_1 i ξ_2 ?
- Jakie są klucze relacji Mafia?
- Anomalie?
- Podaj odwracalny rozkład relacji Mafia do postaci BCNF(!?).
- Rozkład?
- Odwracalny?
- BCNF?
- Czy rozkład ten zachowuje zależności?
- Jakie rodzi to problemy?
- Czyli jak SZBD pilnuje zależności w bazie?

Postać normalna Boyce-Codda

Definition (Postać normalna Boyce-Codda, BCNF)

Relacja R ze zbiorem zależności funkcyjnych F jest w postaci normalnej Boyce-Codda, jeśli dla każdej nietrywialnej zależności $\alpha \rightarrow \beta$ ($\alpha \cap \beta = \emptyset$) zbiór α jest nadkluczem.

Postać normalna Boyce-Codda

Definition (Postać normalna Boyce-Codda, BCNF)

Relacja R ze zbiorem zależności funkcyjnych F jest w postaci normalnej Boyce-Codda, jeśli dla każdej nietrywialnej zależności $\alpha \rightarrow \beta$ ($\alpha \cap \beta = \emptyset$) zbiór α jest nadkluczem.

Uwagi:

Postać normalna Boyce-Codda

Definition (Postać normalna Boyce-Codda, BCNF)

Relacja R ze zbiorem zależności funkcyjnych F jest w postaci normalnej Boyce-Codda, jeśli dla każdej nietrywialnej zależności $\alpha \rightarrow \beta$ ($\alpha \cap \beta = \emptyset$) zbiór α jest nadkluczem.

Uwagi:

- 1 Relacja w BCNF ma tylko zależności trywialne i wynikające z nadklucza.

Postać normalna Boyce-Codda

Definition (Postać normalna Boyce-Codda, BCNF)

Relacja R ze zbiorem zależności funkcyjnych F jest w postaci normalnej Boyce-Codda, jeśli dla każdej nietrywialnej zależności $\alpha \rightarrow \beta$ ($\alpha \cap \beta = \emptyset$) zbiór α jest nadkluczem.

Uwagi:

- 1 Relacja w BCNF ma tylko zależności trywialne i wynikające z nadklucza.
- 2 Kontrola zależności funkcyjnych w relacji w BCNF sprowadza się do kontroli własności klucza.

Postać normalna Boyce-Codda

Definition (Postać normalna Boyce-Codda, BCNF)

Relacja R ze zbiorem zależności funkcyjnych F jest w postaci normalnej Boyce-Codda, jeśli dla każdej nietrywialnej zależności $\alpha \rightarrow \beta$ ($\alpha \cap \beta = \emptyset$) zbiór α jest nadkluczem.

Uwagi:

- 1 Relacja w BCNF ma tylko zależności trywialne i wynikające z nadklucza.
- 2 Kontrola zależności funkcyjnych w relacji w BCNF sprowadza się do kontroli własności klucza.

Przykłady:

Postać normalna Boyce-Codda

Definition (Postać normalna Boyce-Codda, BCNF)

Relacja R ze zbiorem zależności funkcyjnych F jest w postaci normalnej Boyce-Codda, jeśli dla każdej nietrywialnej zależności $\alpha \rightarrow \beta$ ($\alpha \cap \beta = \emptyset$) zbiór α jest nadkluczem.

Uwagi:

- 1 Relacja w BCNF ma tylko zależności trywialne i wynikające z nadklucza.
- 2 Kontrola zależności funkcyjnych w relacji w BCNF sprowadza się do kontroli własności klucza.

Przykłady:

- 1 Mafia(Miasto, Gang, Proceder),

Postać normalna Boyce-Codda

Definition (Postać normalna Boyce-Codda, BCNF)

Relacja R ze zbiorem zależności funkcyjnych F jest w postaci normalnej Boyce-Codda, jeśli dla każdej nietrywialnej zależności $\alpha \rightarrow \beta$ ($\alpha \cap \beta = \emptyset$) zbiór α jest nadkluczem.

Uwagi:

- 1 Relacja w BCNF ma tylko zależności trywialne i wynikające z nadklucza.
- 2 Kontrola zależności funkcyjnych w relacji w BCNF sprowadza się do kontroli własności klucza.

Przykłady:

- 1 Mafia(Miasto, Gang, Proceder),
- 2 Gang \rightarrow Miasto, Miasto, Proceder \rightarrow Gang

Postać normalna Boyce-Codda

Definition (Postać normalna Boyce-Codda, BCNF)

Relacja R ze zbiorem zależności funkcyjnych F jest w postaci normalnej Boyce-Codda, jeśli dla każdej nietrywialnej zależności $\alpha \rightarrow \beta$ ($\alpha \cap \beta = \emptyset$) zbiór α jest nadkluczem.

Uwagi:

- 1 Relacja w BCNF ma tylko zależności trywialne i wynikające z nadklucza.
- 2 Kontrola zależności funkcyjnych w relacji w BCNF sprowadza się do kontroli własności klucza.

Przykłady:

- 1 Mafia(Miasto, Gang, Proceder),
- 2 $\text{Gang} \rightarrow \text{Miasto}$, $\text{Miasto} \rightarrow \text{Proceder} \rightarrow \text{Gang}$
- 3 Lokalizacje(Miasto, Gang), Procedery(Gang, Proceder) są w BCNF

Postać normalna Boyce-Codda

Definition (Postać normalna Boyce-Codda, BCNF)

Relacja R ze zbiorem zależności funkcyjnych F jest w postaci normalnej Boyce-Codda, jeśli dla każdej nietrywialnej zależności $\alpha \rightarrow \beta$ ($\alpha \cap \beta = \emptyset$) zbiór α jest nadkluczem.

Uwagi:

- 1 Relacja w BCNF ma tylko zależności trywialne i wynikające z nadklucza.
- 2 Kontrola zależności funkcyjnych w relacji w BCNF sprowadza się do kontroli własności klucza.

Przykłady:

- 1 Mafia(Miasto, Gang, Proceder),
- 2 $\text{Gang} \rightarrow \text{Miasto}$, $\text{Miasto} \rightarrow \text{Proceder} \rightarrow \text{Gang}$
- 3 Lokalizacje(Miasto, Gang), Procedery(Gang, Proceder) są w BCNF
- 4 $\text{Miasto}, \text{Proceder} \rightarrow \text{Gang}$ może być niespełniona

Postać normalna Boyce-Codda

Definition (Postać normalna Boyce-Codda, BCNF)

Relacja R ze zbiorem zależności funkcyjnych F jest w postaci normalnej Boyce-Codda, jeśli dla każdej nietrywialnej zależności $\alpha \rightarrow \beta$ ($\alpha \cap \beta = \emptyset$) zbiór α jest nadkluczem.

Uwagi:

- 1 Relacja w BCNF ma tylko zależności trywialne i wynikające z nadklucza.
- 2 Kontrola zależności funkcyjnych w relacji w BCNF sprowadza się do kontroli własności klucza.

Przykłady:

- 1 Mafia(Miasto, Gang, Proceder),
- 2 $\text{Gang} \rightarrow \text{Miasto}$, $\text{Miasto} \rightarrow \text{Proceder} \rightarrow \text{Gang}$
- 3 Lokalizacje(Miasto, Gang), Procedery(Gang, Proceder) są w BCNF
- 4 $\text{Miasto}, \text{Proceder} \rightarrow \text{Gang}$ może być niespełniona inne sposoby np. trigger

Rozkład relacji

- Rozkładem relacji R nazywamy zbiór relacji $\{R_1, \dots, R_k\}$ taki, że $R = R_1 \cup \dots \cup R_k$.

Rozkład relacji

- Rozkładem relacji R nazywamy zbiór relacji $\{R_1, \dots, R_k\}$ taki, że $R = R_1 \cup \dots \cup R_k$.
- Oznaczmy przez F^+ domknięcie zbioru zależności F
(wszystkie zależności, którą muszą zachodzić gdy zależności z F zachodzą)

Rozkład relacji

- **Rozkładem relacji R** nazywamy zbiór relacji $\{R_1, \dots, R_k\}$ taki, że $R = R_1 \cup \dots \cup R_k$.
- Oznaczmy przez F^+ **domknięcie zbioru zależności F**
(wszystkie zależności, którą muszą zachodzić gdy zależności z F zachodzą)
- Dla F — zbioru zależności R , **rzutem F na R_i** jest $F_i = \{\alpha \rightarrow \beta \in F^+ \mid \alpha, \beta \in R_i\}$.

Rozkład relacji

- **Rozkładem relacji R** nazywamy zbiór relacji $\{R_1, \dots, R_k\}$ taki, że $R = R_1 \cup \dots \cup R_k$.
- Oznaczmy przez F^+ **domknięcie zbioru zależności F**
(wszystkie zależności, którą muszą zachodzić gdy zależności z F zachodzą)
- Dla F — zbioru zależności R , **rzutem F na R_i** jest $F_i = \{\alpha \rightarrow \beta \in F^+ \mid \alpha, \beta \in R_i\}$.
- Dla r — stanu relacji R , **stanem R_i** jest $r_i = \pi_{R_i}(r)$.

Rozkład relacji

- **Rozkładem relacji R** nazywamy zbiór relacji $\{R_1, \dots, R_k\}$ taki, że $R = R_1 \cup \dots \cup R_k$.
- Oznaczmy przez F^+ **domknięcie zbioru zależności F**
(wszystkie zależności, którą muszą zachodzić gdy zależności z F zachodzą)
- Dla F — zbioru zależności R , **rzutem F na R_i** jest $F_i = \{\alpha \rightarrow \beta \in F^+ \mid \alpha, \beta \in R_i\}$.
- Dla r — stanu relacji R , **stanem R_i** jest $r_i = \pi_{R_i}(r)$.
- **Złączenie naturalne** jest operacją przeciwną do rozkładu.

Rozkład relacji

- **Rozkładem relacji R** nazywamy zbiór relacji $\{R_1, \dots, R_k\}$ taki, że $R = R_1 \cup \dots \cup R_k$.
- Oznaczmy przez F^+ **domknięcie zbioru zależności F**
(wszystkie zależności, którą muszą zachodzić gdy zależności z F zachodzą)
- Dla F — zbioru zależności R , **rzutem F na R_i** jest $F_i = \{\alpha \rightarrow \beta \in F^+ \mid \alpha, \beta \in R_i\}$.
- Dla r — stanu relacji R , **stanem R_i** jest $r_i = \pi_{R_i}(r)$.
- **Złączenie naturalne** jest operacją przeciwną do rozkładu.
- Rozkład R na R_1, \dots, R_k jest **odwracalny**, jeśli dla każdego poprawnego stanu r (spełniającego zależności F) zachodzi:

$$r = r_1 \bowtie r_2 \bowtie \dots \bowtie r_k$$

Rozkład relacji

- **Rozkładem relacji R** nazywamy zbiór relacji $\{R_1, \dots, R_k\}$ taki, że $R = R_1 \cup \dots \cup R_k$.
- Oznaczmy przez F^+ **domknięcie zbioru zależności F**
(wszystkie zależności, którą muszą zachodzić gdy zależności z F zachodzą)
- Dla F — zbioru zależności R , **rzutem F na R_i** jest $F_i = \{\alpha \rightarrow \beta \in F^+ \mid \alpha, \beta \in R_i\}$.
- Dla r — stanu relacji R , **stanem R_i** jest $r_i = \pi_{R_i}(r)$.
- **Złączenie naturalne** jest operacją przeciwną do rozkładu.
- Rozkład R na R_1, \dots, R_k jest **odwracalny**, jeśli dla każdego poprawnego stanu r (spełniającego zależności F) zachodzi:

$$r = r_1 \bowtie r_2 \bowtie \dots \bowtie r_k$$

- Rozkład R na R_1, \dots, R_k **zachowuje zależności**, jeśli:

$$F^+ = (F_1 \cup F_2 \cup \dots \cup F_k)^+$$

Rozkład relacji

- **Rozkładem relacji R** nazywamy zbiór relacji $\{R_1, \dots, R_k\}$ taki, że $R = R_1 \cup \dots \cup R_k$.
- Oznaczmy przez F^+ **domknięcie zbioru zależności F**
(wszystkie zależności, którą muszą zachodzić gdy zależności z F zachodzą)
- Dla F — zbioru zależności R , **rzutem F na R_i** jest $F_i = \{\alpha \rightarrow \beta \in F^+ \mid \alpha, \beta \in R_i\}$.
- Dla r — stanu relacji R , **stanem R_i** jest $r_i = \pi_{R_i}(r)$.
- **Złączenie naturalne** jest operacją przeciwną do rozkładu.
- Rozkład R na R_1, \dots, R_k jest **odwracalny**, jeśli dla każdego poprawnego stanu r (spełniającego zależności F) zachodzi:

$$r = r_1 \bowtie r_2 \bowtie \dots \bowtie r_k$$

- Rozkład R na R_1, \dots, R_k **zachowuje zależności**, jeśli:

$$F^+ = (F_1 \cup F_2 \cup \dots \cup F_k)^+$$

- Rozkład relacji na składowe **MUSI** być odwracalny i **POWINIEN** zachowywać zależności.

Rozkład relacji do BCNF

Rozkład relacji do BCNF

Lemma

Niech R będzie relacją i F jej zbiorem zależności funkcyjnych. Jeżeli $\alpha \rightarrow \beta \in F^+$ jest nietrywialna ($\alpha \cap \beta = \emptyset$), to rozkład R na $R_1 = \alpha\beta$ i $R_2 = R \setminus \beta$ jest odwracalny.

Rozkład relacji do BCNF

Lemma

Niech R będzie relacją i F jej zbiorem zależności funkcyjnych. Jeżeli $\alpha \rightarrow \beta \in F^+$ jest nietrywialna ($\alpha \cap \beta = \emptyset$), to rozkład R na $R_1 = \alpha\beta$ i $R_2 = R \setminus \beta$ jest odwracalny.

Lemma

Każda relacja ma **odwracalny** rozkład na składowe w BCNF.

Rozkład relacji do BCNF

Lemma

Niech R będzie relacją i F jej zbiorem zależności funkcyjnych. Jeżeli $\alpha \rightarrow \beta \in F^+$ jest nietrywialna ($\alpha \cap \beta = \emptyset$), to rozkład R na $R_1 = \alpha\beta$ i $R_2 = R \setminus \beta$ jest odwracalny.

Lemma

Każda relacja ma **odwracalny** rozkład na składowe w BCNF.

Lemma

Istnieją relacje, które nie mają **odwracalnego i zachowującego zależności** rozkładu na składowe w BCNF.

Rozkład relacji do BCNF

Lemma

Niech R będzie relacją i F jej zbiorem zależności funkcyjnych. Jeżeli $\alpha \rightarrow \beta \in F^+$ jest nietrywialna ($\alpha \cap \beta = \emptyset$), to rozkład R na $R_1 = \alpha\beta$ i $R_2 = R \setminus \beta$ jest odwracalny.

Lemma

Każda relacja ma **odwracalny** rozkład na składowe w BCNF.

Lemma

Istnieją relacje, które nie mają **odwracalnego i zachowującego zależności** rozkładu na składowe w BCNF.

Przykład

Lokalizacje(Miasto, Gang), Zajęcia(Gang, Proceder) vs. Miasto, Proceder \rightarrow Gang

Trzecia postać normalna

Definition (Trzecia postać normalna, 3NF)

Relacja R z zależnościami funkcyjnymi F jest w trzeciej postaci normalnej, jeśli każda zależność $\alpha \rightarrow B \in F$

Trzecia postać normalna

Definition (Trzecia postać normalna, 3NF)

Relacja R z zależnościami funkcyjnymi F jest w trzeciej postaci normalnej, jeśli każda zależność $\alpha \rightarrow B \in F$

- jest trywialna ($B \in \alpha$) albo

Trzecia postać normalna

Definition (Trzecia postać normalna, 3NF)

Relacja R z zależnościami funkcyjnymi F jest w trzeciej postaci normalnej, jeśli każda zależność $\alpha \rightarrow B \in F$

- jest trywialna ($B \in \alpha$) albo
- wynika z nadklucza ($(\alpha)_F^+ = R$) albo

Trzecia postać normalna

Definition (Trzecia postać normalna, 3NF)

Relacja R z zależnościami funkcyjnymi F jest w trzeciej postaci normalnej, jeśli każda zależność $\alpha \rightarrow B \in F$

- jest trywialna ($B \in \alpha$) albo
- wynika z nadklucza ($(\alpha)_F^+ = R$) albo
- ma po prawej stronie atrybut główny (B należy do jakiegoś klucza).

Trzecia postać normalna

Definition (Trzecia postać normalna, 3NF)

Relacja R z zależnościami funkcyjnymi F jest w trzeciej postaci normalnej, jeśli każda zależność $\alpha \rightarrow B \in F$

- jest trywialna ($B \in \alpha$) albo
- wynika z nadklucza ($(\alpha)_F^+ = R$) albo
- ma po prawej stronie atrybut główny (B należy do jakiegoś klucza).

Lemma

Każda relacja ma odwracalny i zachowujący zależności rozkład na składowe w postaci 3NF.

Algorytm rozkładu do 3NF

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

- nie ma zależności trywialnych, np. $AB \rightarrow AC$

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

- nie ma zależności trywialnych, np. $AB \rightarrow C$

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

- nie ma zależności trywialnych, np. $AB \rightarrow C$
- nie ma zależności nadmiarowych, czyli wynikających z pozostałych zależności F_{min} , np. $A \rightarrow B, B \rightarrow C, A \rightarrow C$

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

- nie ma zależności trywialnych, np. $AB \rightarrow C$
- nie ma zależności nadmiarowych, czyli wynikających z pozostałych zależności F_{min} , np. $A \rightarrow B, B \rightarrow C$,

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

- nie ma zależności trywialnych, np. $AB \rightarrow C$
- nie ma zależności nadmiarowych, czyli wynikających z pozostałych zależności F_{min} , np. $A \rightarrow B, B \rightarrow C$,
- nie ma atrybutów lewostronnie nadmiarowych $AB \rightarrow C, A \rightarrow B$.

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

- nie ma zależności trywialnych, np. $AB \rightarrow C$
- nie ma zależności nadmiarowych, czyli wynikających z pozostałych zależności F_{min} , np. $A \rightarrow B, B \rightarrow C$,
- nie ma atrybutów lewostronnie nadmiarowych $A \rightarrow C, A \rightarrow B$.

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

- nie ma zależności trywialnych, np. $AB \rightarrow C$
- nie ma zależności nadmiarowych, czyli wynikających z pozostałych zależności F_{min} , np. $A \rightarrow B, B \rightarrow C$,
- nie ma atrybutów lewostronnie nadmiarowych $A \rightarrow C, A \rightarrow B$.

Algorytm rozkładu do 3NF

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

- nie ma zależności trywialnych, np. $AB \rightarrow C$
- nie ma zależności nadmiarowych, czyli wynikających z pozostałych zależności F_{min} , np. $A \rightarrow B, B \rightarrow C$,
- nie ma atrybutów lewostronnie nadmiarowych $A \rightarrow C, A \rightarrow B$.

Algorytm rozkładu do 3NF

- 1 Wyznacz F_{min} .

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

- nie ma zależności trywialnych, np. $AB \rightarrow C$
- nie ma zależności nadmiarowych, czyli wynikających z pozostałych zależności F_{min} , np. $A \rightarrow B, B \rightarrow C$,
- nie ma atrybutów lewostronnie nadmiarowych $A \rightarrow C, A \rightarrow B$.

Algorytm rozkładu do 3NF

- 1 Wyznacz F_{min} .
- 2 Dla każdej zależności $\alpha \rightarrow \beta \in F_{min}$ utwórz składową $R_i = \alpha\beta$. Usuń składowe zawierające się w innych.

Algorytm rozkładu do 3NF

Definition (F_{min})

Minimalnym pokryciem zbioru zależności funkcyjnych F nazwiemy równoważny F zbiór F_{min} , w którym:

- nie ma zależności trywialnych, np. $AB \rightarrow C$
- nie ma zależności nadmiarowych, czyli wynikających z pozostałych zależności F_{min} , np. $A \rightarrow B, B \rightarrow C$,
- nie ma atrybutów lewostronnie nadmiarowych $A \rightarrow C, A \rightarrow B$.

Algorytm rozkładu do 3NF

- 1 Wyznacz F_{min} .
- 2 Dla każdej zależności $\alpha \rightarrow \beta \in F_{min}$ utwórz składową $R_i = \alpha\beta$. Usuń składowe zawierające się w innych.
- 3 Jeśli żadna z utworzonych składowych nie zawiera klucza R , to dodaj do rozkładu składową K dla pewnego klucza K relacji R .

Przykład c.d.

- Mafia(Miasto, Gang, Proceder, Szef)

Przykład c.d.

- Mafia(Miasto, Gang, Proceder, Szef)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$, $\xi_2: \text{Miasto, Proceder} \rightarrow \text{Gang}$, $\xi_3: \text{Gang} \rightarrow \text{Szef}$

Przykład c.d.

- Mafia(Miasto, Gang, Proceder, Szef)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$, $\xi_2: \text{Miasto}, \text{Proceder} \rightarrow \text{Gang}$, $\xi_3: \text{Gang} \rightarrow \text{Szef}$
- Jakie są klucze relacji Mafia?

Przykład c.d.

- Mafia(Miasto, Gang, Proceder, Szef)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$, $\xi_2: \text{Miasto}, \text{Proceder} \rightarrow \text{Gang}$, $\xi_3: \text{Gang} \rightarrow \text{Szef}$
- Jakie są klucze relacji Mafia?
- Podaj odwracalny i zachowujący zależności rozkład relacji Mafia do postaci 3NF.

Przykład c.d.

- Mafia(Miasto, Gang, Proceder, Szef)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$, $\xi_2: \text{Miasto}, \text{Proceder} \rightarrow \text{Gang}$, $\xi_3: \text{Gang} \rightarrow \text{Szef}$
- Jakie są klucze relacji Mafia?
- Podaj odwracalny i zachowujący zależności rozkład relacji Mafia do postaci 3NF.
- Miasta(Gang, Miasto), Mafia(Miasto, Gang, Proceder), Szefowie(Gang, Szef)

Przykład c.d.

- Mafia(Miasto, Gang, Proceder, Szef)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$, $\xi_2: \text{Miasto}, \text{Proceder} \rightarrow \text{Gang}$, $\xi_3: \text{Gang} \rightarrow \text{Szef}$
- Jakie są klucze relacji Mafia?
- Podaj odwracalny i zachowujący zależności rozkład relacji Mafia do postaci 3NF.

Mafia(Miasto, Gang, Proceder), Szefowie(Gang, Szef)

Przykład c.d.

- Mafia(Miasto, Gang, Proceder, Szef)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$, $\xi_2: \text{Miasto}, \text{Proceder} \rightarrow \text{Gang}$, $\xi_3: \text{Gang} \rightarrow \text{Szef}$
- Jakie są klucze relacji Mafia?
- Podaj odwracalny i zachowujący zależności rozkład relacji Mafia do postaci 3NF.
Mafia(Miasto, Gang, Proceder), Szefowie(Gang, Szef)
- Dlaczego zachowuje zależności?

Przykład c.d.

- Mafia(Miasto, Gang, Proceder, Szef)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$, $\xi_2: \text{Miasto}, \text{Proceder} \rightarrow \text{Gang}$, $\xi_3: \text{Gang} \rightarrow \text{Szef}$
- Jakie są klucze relacji Mafia?
- Podaj odwracalny i zachowujący zależności rozkład relacji Mafia do postaci 3NF.

Mafia(Miasto, Gang, Proceder), Szefowie(Gang, Szef)

- Dlaczego zachowuje zależności?
- Dlaczego jest odwracalny?

Przykład c.d.

- Mafia(Miasto, Gang, Proceder, Szef)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$, $\xi_2: \text{Miasto}, \text{Proceder} \rightarrow \text{Gang}$, $\xi_3: \text{Gang} \rightarrow \text{Szef}$
- Jakie są klucze relacji Mafia?
- Podaj odwracalny i zachowujący zależności rozkład relacji Mafia do postaci 3NF.

Mafia(Miasto, Gang, Proceder), Szefowie(Gang, Szef)

- Dlaczego zachowuje zależności?
- Dlaczego jest odwracalny? Wskazówka: Zaczynij od krotki z relacji z kluczem

Przykład c.d.

- Mafia(Miasto, Gang, Proceder, Szef)
- $\xi_1: \text{Gang} \rightarrow \text{Miasto}$, $\xi_2: \text{Miasto}, \text{Proceder} \rightarrow \text{Gang}$, $\xi_3: \text{Gang} \rightarrow \text{Szef}$
- Jakie są klucze relacji Mafia?
- Podaj odwracalny i zachowujący zależności rozkład relacji Mafia do postaci 3NF.
Mafia(Miasto, Gang, Proceder), Szefowie(Gang, Szef)
- Dlaczego zachowuje zależności?
- Dlaczego jest odwracalny? Wskazówka: Zaczynij od krotki z relacji z kluczem
- cf. BCNF: Miasta(Miasto, Gang), Procedery(Gang, Proceder), Szefowie(Gang, Szef)