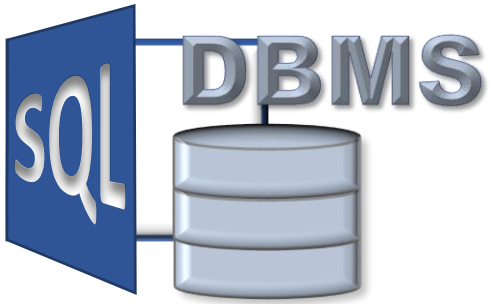
Bazy danych

Relacyjny model danych



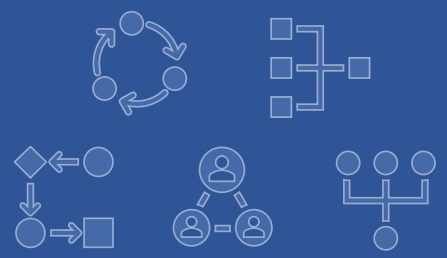
zależności funkcjonalne

I, II, III forma normalna relacji

operacje na strukturach (DDL)

manipulowanie danymi (DML)

Materiały
http://staff.uz.zgora.pl/gpajak
http://staff.uz.zgora.pl/gpajak



Modelowanie danych

Przykład

Zadanie: Należy zaprojektować system informatyczny (bazę danych) dla potrzeb firmy produkcyjnej, która umożliwi gromadzenie i przetwarzanie danych o zleceniach realizowanych dla klientów.

Przeznaczenie: Aplikacja przetwarza dane zleceń realizowanych przez firmę.

Funkcje:

- Przechowuje informacje o zleceniach.
- Przechowuje dane klientów.
- Przechowuje dane wytworzonych produktów.
- Tworzy statystyki (np. wykaz aktualnie realizowanych zleceń, lub zleceń klienta, itp.).

Schemat opisowy: Klient jest identyfikowany przez unikalny numer (np. NIP lub PESEL), ma nazwę i może mieć przydzielonego doradcę. Doradca jest pracownikiem firmy, jest identyfikowany przez PESEL, ma nazwisko i imię. Produkty są identyfikowane przez identyfikator nadawany przez firmę (dowolny ciąg znaków), ma nazwę i cenę. Zlecenie dotyczy konkretnego produktu, jest przypisane do jednego klienta, ma datę początku i zakończenia realizacji oraz liczbę sztuk produktu, która została zamówiona.

Modelowanie danych

Modelowanie danych

proces tworzenia odpowiedniego odwzorowania danych dla potrzeb systemu informatycznego, polega na przejściu od swobodnego opisu fragmentu rzeczywistości (modelowanego sytemu) do jego formalnej reprezentacji.

Elementy świata rzeczywistego

- **obiekt** – składnik rzeczywistego systemu postrzegany jako istotny przez jednostkę lub grupę, przyszłych użytkowników bazy danych,
- **powiązanie** – opis stanu, w którym znalazły się co najmniej dwa obiekty.

Obiekty i powiązania mogą być dodatkowo opisane za pomocą **atrybutów**.

Przykłady

- obiekt: klient, atrybuty: identyfikator, nazwa,
- obiekt: doradca, atrybuty: PESEL, nazwisko, imię,
- obiekt: produkt, atrybuty: identyfikator, nazwa, cena,
- powiązanie: klient-doradca,
- powiązanie: klient-produkt (zlecenie), atrybuty: data rozpoczęcia i zakończenia, liczba.

Model relacyjny – definicje (1)

Relacja R na zbiorach $D_1, D_2, \ldots, D_n$

to dowolny podzbiór iloczynu kartezjańskiego tych zbiorów

Symbol relacji

$$R(D_1, D_2, \ldots, D_n),\ R \subset D_1 \times D_2 \times \ldots \times D_n.$$

Zapis w postaci $R(D_1, D_2, \ldots, D_n)$ jest schematem relacji, R nazwą relacji, a elementy $D_1, D_2, \ldots, D_n$ atrybutami lub składnikami relacji.

Krotka relacji

ciąg wartości atrybutów danego schematu relacji.

Alternatywna definicja relacji

Relacją R na zbiorach $D_1, D_2, \ldots, D_n$ nazywamy dowolny zbiór krotek postaci:

$$<d_1, d_2, \ldots, d_n>$$

takich, że: $d_1 \in D_1,\ d_2 \in D_2,\ \ldots,\ d_n \in D_n$.

Model relacyjny – definicje (2)

Identyfikator relacji

ciąg atrybutów, których wartości określają w sposób jednoznaczny krotkę relacji.

Identyfikator kluczowy (klucz) relacji

jeden, dowolnie wybrany identyfikator relacji (zazwyczaj kryterium wyboru jest długość). W schemacie relacji klucz jest zaznaczany przez podkreślenie atrybutów.

Klucz naturalny

klucz złożony z atrybutów, których obecność wynika z analizy problemu.

Klucz sztuczny

sztucznie wprowadzony atrybut relacji, którego wartości gwarantują jednoznaczną identyfikację krotek. W roli klucza sztucznego najczęściej występuje liczba porządkowa.

Przykład

Klucz naturalny: *Doradca(PESEL, Nazwisko, Imię)*

Klucz sztuczny: *Doradca(ID, PESEL, Nazwisko, Imię)*

ID – identyfikator doradcy, unikalna wartość liczbowa.

Przykład – rozwiązanie trywialne

Schemat relacji Z (Zamówienie) (na podstawie analizy zadania, s.8)

Z (IDK, NazwaK, PESEL, NazwiskoD, ImięD, IDP, NazwaP, CenaP, DataPZ, DataZZ, Liczba)

Atrybuty

- *IDK* – identyfikator klienta,
- *NazwaK* – nazwa klienta,
- *PESEL* – pesel doradcy,
- *NazwiskoD* – nazwisko doradcy,
- *ImięD* – imię doradcy,
- *IDP* – identyfikator produktu,
- *NazwaP* – nazwa produktu,
- *CenaP* – cena jednostkowa produktu,
- *DataPZ* – data początku realizacji zamówienia,
- *DataZZ* – data zakończenia realizacji zamówienia,
- *Liczba* – liczba sztuk zamówionego produktu.

Uwaga: kompletny schemat relacji wymaga określenia klucza, odpowiednia analiza jest przedstawiona na stronie następnej.

Przykładowy zestaw danych

Przykładowy zbiór krotek relacji Z (Zamówienie)

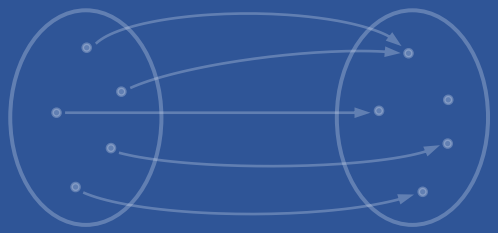
IDK	NazwaK	PESEL	NazwiskoD	ImięD	IDP	NazwaP	CenaP	DataPZ	DataZZ	Liczba
0224111111	ABC Sp.zo.o.	90052000234	Kowalska	Anna	W150	Wałek 150	25	25.01.10	25.01.15	100
0224111111	ABC Sp.zo.o.	90052000234	Kowalska	Anna	P250	Płaskownik 250	15	25.01.10	25.01.13	50
0441222222	XYZ S.A.	90052000234	Kowalska	Anna	W150	Wałek 150	25	25.01.14		200
0441222222	XYZ S.A.	90052000234	Kowalska	Anna	W100	Wałek 100	20	25.01.14	25.01.17	50
0441222222	XYZ S.A.	90052000234	Kowalska	Anna	P500	Płaskownik 500	30	25.01.14	25.01.15	200
0224111111	ABC Sp.zo.o.	90052000234	Kowalska	Anna	W100	Wałek 100	20	25.01.14	25.01.15	80
0441222222	XYZ S.A.	82012090803	Nowak	Andrzej	W150	Wałek 150	25	25.01.21	25.01.23	100
0224111111	ABC Sp.zo.o.	90052000234	Kowalska	Anna	P200	Płaskownik 200	10	25.01.30		125

Uwaga: Każde zamówienie jest przechowywane jest jako osobna krotka relacji nawet jeżeli ten sam klient jednocześnie zamawia klika produktów (ABC Sp.z o.o. 10.01.2025). Umieszczenie kilku produktów w tym samym polu (np. „po przecinku”) jest fundamentalnym błędem modelu i uniemożliwi efektywne przetwarzanie danych w docelowej aplikacji.

Analiza relacji Zamówienie

Anomalie wiązane z zaproponowanym modelem danych

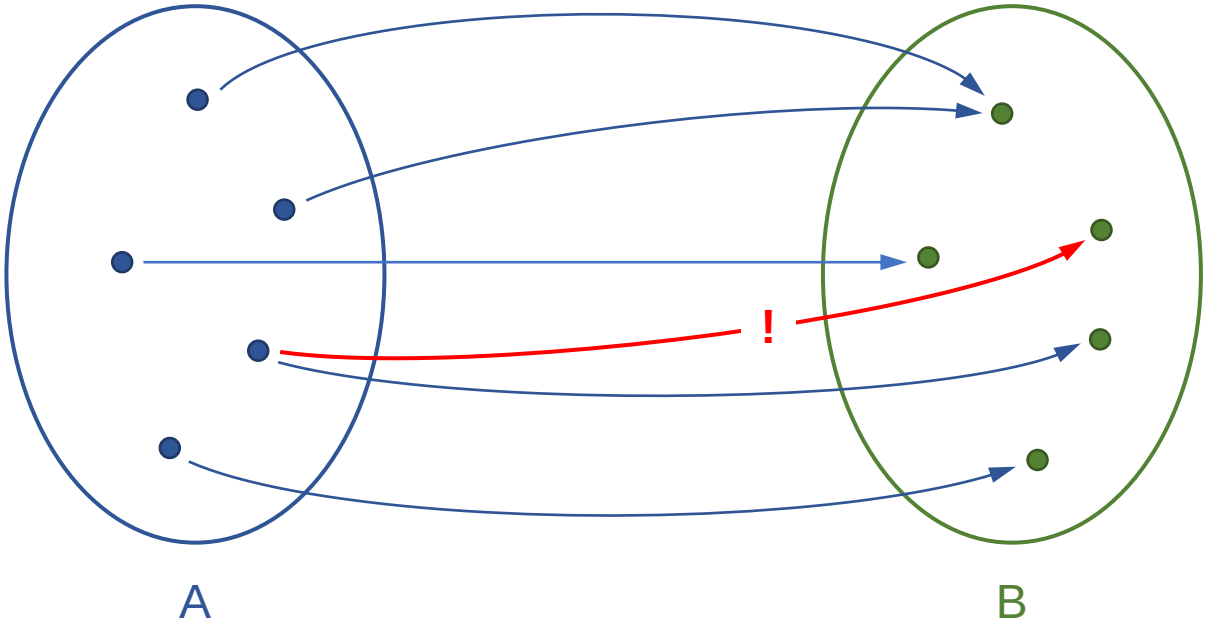
1. Anomalie przy wstawianiu – w bazie nie można umieścić informacji o produktach, które nie zostały zamówione lub doradcach, który nie mają klientów (nowi pracownicy).
2. Redundancja – wiele informacji powtarza się: nazwy klientów, nazwiska i imiona doradców (Kowalska pięciokrotnie, Nowak trzykrotnie), dane produktów (W150 trzykrotnie, W100 dwukrotnie).
3. Niespójność danych – wielokrotne powtarzanie informacji prowadzi do sprzecznych informacji: do 14.01 doradcą XYZ S.A. jest Anna Kowalska, po tym dniu nastąpiła zmiana doradcy i na zamówieniu z 21.01 występuje Andrzej Nowak (ustalenie doradcy wymaga analizy zawartości całej bazy danych).
4. Anomalie przy aktualizacji – zmiana danych osobowych (np. zmiana nazwiska doradcy) wymaga aktualizacji wszystkich krotek związanych z daną osobą (bez tej operacji wystąpi niespójność danych).
5. Anomalie przy usuwaniu – informacje o doradcach nie są odseparowane od zamówień, nie można usunąć danych doradcy, który zmienił miejsce pracy, więc nie można ustalić listy doradców zatrudnionych w firmie.



Zależności funkcjonalne

Zależność funkcjonalna

Składnik B jest **funkcjonalnie zależny** od składnika A w relacji R(A, B, C, D) jeżeli każdej wartości a∈A jest przyporządkowana tylko jedna wartość b∈B. Zależność funkcjonalną B w stosunku do A zapisujemy A→B.



Uwaga: B jest zależny funkcjonalnie od A jeżeli występują wyłącznie zależności reprezentowane przez niebieskie strzałki. Jeżeli występuje zależność przedstawiona czerwoną strzałką (lub istnieje możliwość wystąpienia zależności) jednemu elementowi z A odpowiada kilka elementów z B, więc B nie jest zależny funkcjonalnie od A.

Zależność funkcjonalna – przykłady

Z (IDK, NazwaK, PESEL, NazwiskoD, ImięD, IDP, NazwaP, CenaP, DataPZ, DataZZ, Liczba)

PESEL → *NazwiskoD*

Jest zależnością funkcjonalną, ponieważ:

Każdy doradca ma unikalny numer *PESEL*, więc każdemu numerowi *PESEL* odpowiada dokładnie jedno *NazwiskoD*. Zależność odwrotna nie jest funkcjonalna, ponieważ w zbiorze doradców może wystąpić kilka osób o tym samym *Nazwisku*, a w takim przypadku jednemu *Nazwisku* będzie odpowiadać kilka numerów *PESEL*.

PESEL → *IDK*

Nie jest zależnością funkcjonalną, ponieważ:

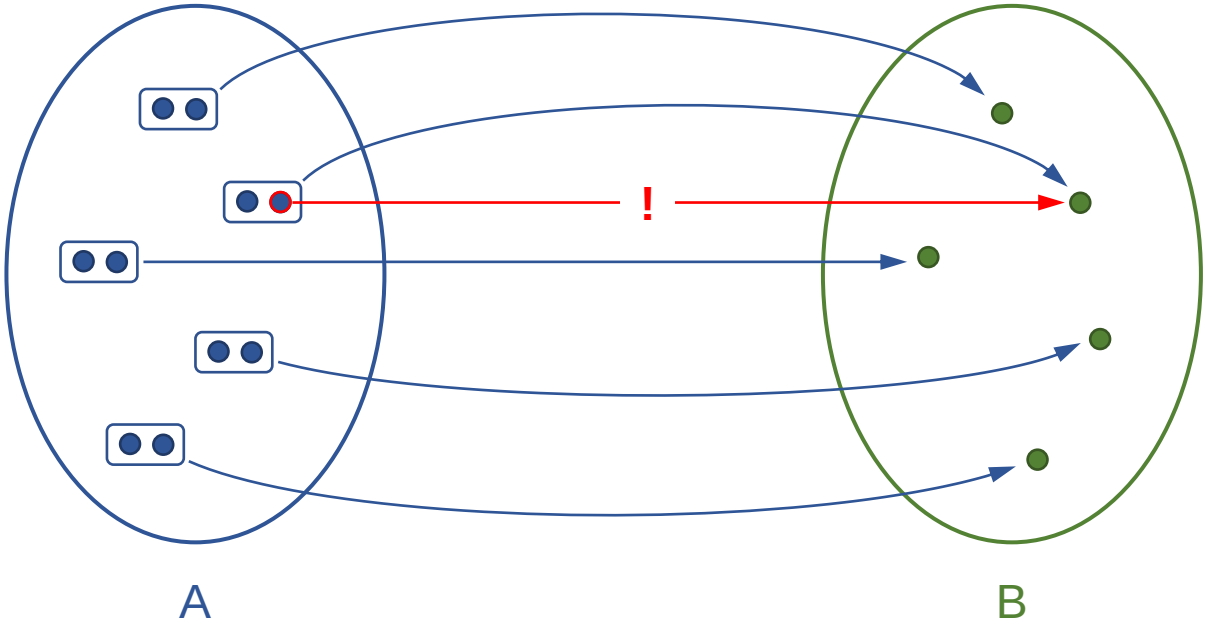
Jeden doradca może mieć (zazwyczaj ma) kilku klientów o różnych identyfikatorach, więc jednemu numerowi *PESEL* może odpowiadać wiele *IDK*.

Uwaga: analizując zależność funkcjonalną należy rozpatrywać wszystkie możliwości, uwzględniając również przypadki które nie występują w aktualnym zbiorze danych, ale mogą pojawić się w przyszłości.

Zależność funkcjonalna elementarna

Składnik B jest w **zależności funkcjonalnej elementarnej** od składnika A w relacji R(A, B, C, D) jeżeli jest funkcjonalnie zależny od całości A i nie jest funkcjonalnie zależny od części A.

Uwaga: Rozpatrywanie zależności funkcjonalnej elementarnej ma sens jedynie gdy składnik A można podzielić na mniejsze części. Jeżeli A stanowi jedną, niepodzielną całość każda zależność funkcjonalna jest jednocześnie elementarna.



Zależność funkcjonalna elementarna – przykłady 1

Z (IDK, NazwaK, PESEL, NazwiskoD, ImięD, IDP, NazwaP, CenaP, DataPZ, DataZZ, Liczba)

IDK, IDP, DataPZ $\rightarrow$ *Liczba*

Jest zależnością funkcjonalną ponieważ:

Identyfikator klienta (*IDK*) z identyfikatorem produktu (*IDP*) i datą początku zamówienia (*DataPZ*) jednoznacznie identyfikują *Liczbę* zamówionych produktów (warunek: w tym samym dniu klient nie składa dwóch zamówień na ten sam produkt).

Zależność jest elementarna ponieważ:

- *Liczba* nie jest funkcjonalnie zależna od *IDK* – jeden klient może składać wiele zamówień na różne liczby produktów (jednemu *IDK* odpowiada wiele *Liczb*),
- *Liczba* nie jest funkcjonalnie zależna od *IDP* – jeden klient może zamawiać różne liczby tego samego produktu w różnych dniach (jednemu *IDP* odpowiada wiele *Liczb*),
- *Liczba* nie jest funkcjonalnie zależna od *DatyPZ* – tego samego dnia klient może zamawiać różne liczby różnych produktów (jednej *DaciePZ* odpowiada wiele *Liczb*),
- *Liczba* nie jest funkcjonalnie zależna od par (*IDK*, *IDP*), (*IDK*, *DataPZ*) i (*IDP*, *DataPZ*).

Wniosek: w celu jednoznacznego określenia *Liczby* zamówionych produktów niezbędne jest podanie *IDK*, *IDP* i *DataPZ*, więc zależność jest funkcjonalna elementarna.

Zależność funkcjonalna elementarna – przykłady 2

Z (IDK, NazwaK, PESEL, NazwiskoD, ImięD, IDP, NazwaP, CenaP, DataPZ, DataZZ, Liczba)

IDK, DataPZ $\rightarrow$ *NazwaK*

Jest zależnością funkcjonalną, ponieważ:

Identyfikator klienta (*IDK*) z datą początku zamówienia (*DataPZ*) jednoznacznie identyfikują nazwę klienta (*NazwaK*), inaczej: na podstawie tych dwóch wartości można określić jak nazywa się klient.

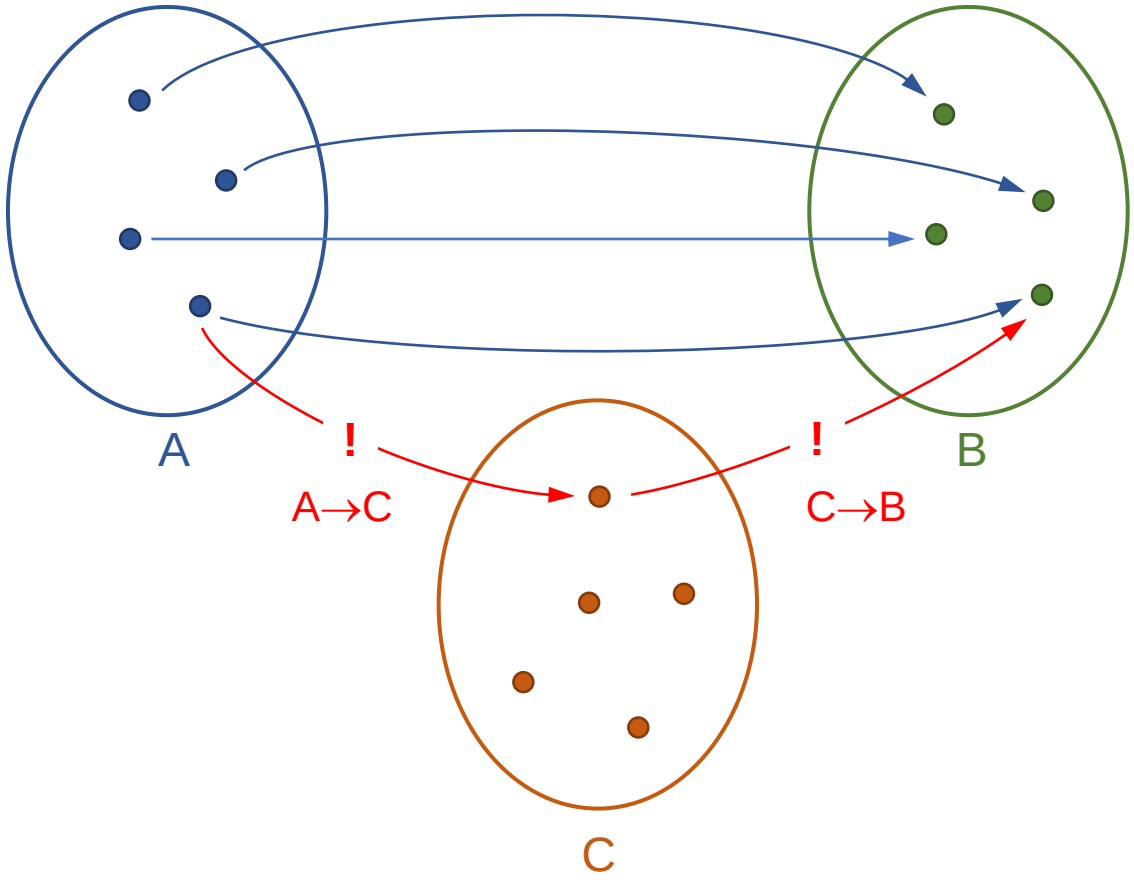
Zależność nie jest elementarna, ponieważ:

NazwaK zależy funkcjonalnie od *IDK*, czyli występuje częściowa zależność funkcjonalna: *IDK* $\rightarrow$ *NazwaK*, inaczej: *DataPZ* nie jest niezbędna do określenia *NazwyK*, wystarczająca jest znajomość *IDK*.

Uwaga: Jeżeli w relacji R występuje zależność funkcjonalna A$\rightarrow$B, to dla dowolnego składnika C z relacji R zachodzi również A,C$\rightarrow$B, jednak takie zależności nie są elementarne.

Zależność funkcjonalna bezpośrednia

Składnik B jest w **zależności funkcjonalnej bezpośredniej** od składnika A w relacji R(A, B, C, D) jeżeli jest funkcjonalnie zależny od A i nie istnieje taki składnik C dla którego zachodzi: $A \rightarrow C$ i $C \rightarrow B$.



Zależność funkcjonalna bezpośrednia – przykłady

Z (IDK, NazwaK, PESEL, NazwiskoD, ImięD, IDP, NazwaP, CenaP, DataPZ, DataZZ, Liczba)

PESEL $\rightarrow$ *ImięD*

Jest zależnością funkcjonalną bezpośrednią, ponieważ:

Jest to zależność funkcjonalna (*PESEL* jednoznacznie określa *ImięD*) i nie istnieje inny składnik, od którego *ImięD* jest zależne funkcjonalnie (*NazwiskoD* może powtórzyć się u kilku doradców, więc *ImęD* nie zależy funkcjonalnie od *NazwiskoD*), nie można więc wskazać składnika, poprzez który uzyskuje się zależność pośrednią.

IDK $\rightarrow$ *NazwiskoD*

Nie jest zależnością bezpośrednią, ponieważ:

Jest to zależność funkcjonalna (*IDK* jednoznacznie określa *NazwiskoD*), ale *IDK* wyznacza jednoznacznie *PESEL* doradcy (*PESEL* jest zależy funkcjonalnie od *IDK*), a *PESEL* określa *NazwiskoD* (*NazwiskoD* jest zależne od *PESEL*, patrz s.10), stąd:

$$IDK \rightarrow PESEL, \quad PESEL \rightarrow NazwiskoD,$$

istnieje więc składnik (*PESEL*), poprzez który uzyskuje się zależność pośrednią.

Formy normalne relacji

I forma normalna relacji

Relacja R jest w **pierwszej formie normalnej (I FN)** jeżeli każdy ze składników, który nie jest elementem klucza, jest w zależności funkcjonalnej od klucza.

Uwaga: Spełnienie warunku I FN gwarantuje prawidłowy wybór klucza relacji.

Cechy relacji w I FN

- Wszystkie atrybuty są zależne funkcjonalnie od klucza.
- Relacja ma prawidłowy klucz, który gwarantuje jednoznaczną identyfikację krotek,
- Relacja jest przygotowana do wykonania kolejnych kroków normalizacji (II FN, III FN), które usuwają bardziej złożone zależności pomiędzy danymi.

Uwagi

- Przed wyznaczeniem klucza należy sprawdzić:
  - czy atrybuty zawierają wartości atomowe (niepodzielne), np. niedozwolone jest przechowywanie listy wartości "po przecinku" w pojedynczym atrybucie relacji,
  - czy istnieją grupy powtarzających się kolumn (kolumny określające wystąpienie tej samej cechy), np. "Produkt1", "Produkt2", itd.
- Uzyskanie I FN wymaga zidentyfikowania wszystkich zależności funkcjonalnych, które występują pomiędzy atrybutami analizowanej relacji.

Graf zależności funkcjonalnych

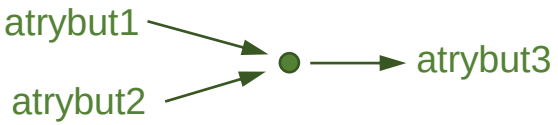
Graf zależności funkcjonalnych

to graf skierowany, którego wierzchołkami są atrybuty (składniki) relacji, a krawędzie skierowane wyznaczają zależności funkcjonalne.

Konstrukcja grafu zależności funkcjonalnych

- Zapisać wszystkie atrybuty relacji;
- Wyznaczyć zależności funkcjonalne, których źródłem są pojedyncze składniki relacji i połączyć odpowiednie wierzchołki grafu krawędziami odpowiadającymi zależnościom;
- Jeżeli istnieją wierzchołki grafu niepołączone z pozostałymi wyznaczyć zależności, których źródłem są pary atrybutów i zaznaczyć je na grafie;
- Powtarzać operację do chwili wyznaczenia grupy atrybutów będących źródłem zależności dla wszystkich pozostałych, taka grupa jest kluczem relacji.

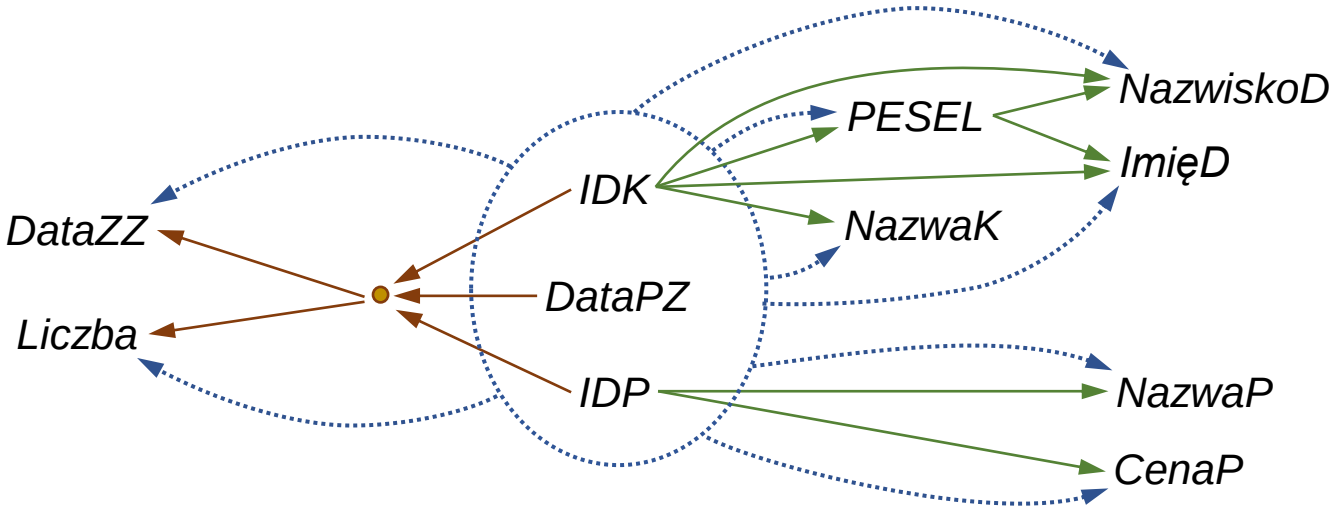
Pojedyncze źródło zależności funkcjonalnej



Wielokrotne źródło zależności funkcjonalnej

Graf zależności funkcjonalnych – przykład

Z (IDK, NazwaK, PESEL, NazwiskoD, ImięD, IDP, NazwaP, CenaP, DataPZ, DataZZ, Liczba)



Zależności funkcjonalne

- *IDK* → *NazwaK, PESEL, NazwiskoD, ImięD,* *PESEL* → *NazwiskoD*, *ImięD,* *IDP* → NazwaP, CenaP
- *IDK,DataPZ,IDP* → DataZZ, Liczba
- *IDK, DataPZ, IDP* są źródłem zależności dla pozostałych atrybutów (warunek: w tym samym dniu klient nie składa dwóch zamówień na ten sam produkt).

I forma normalna – przykład

Relacja *Zamówienie (Z)* w I FN

*Z (**IDK**, NazwaK, PESEL, NazwiskoD, ImięD, **IDP**, NazwaP, CenaP, **DataPZ**, DataZZ, Liczba)*

IDK określa jednoznacznie klienta i przydzielonego mu doradcę, *IDP* identyfikuje produkt i jego cenę, w konkretnym dniu określonym przez *DataPZ* klient składa jedno zamówienie na dany produkt (możliwe jest zamówienie wielu sztuk produktu oraz złożenie zamówień na klika różnych produktów).

Stąd:

- *IDK* $\rightarrow$ *NazwaK*, *PESEL, NazwiskoD, ImięD*, ponieważ *IDK* jednoznacznie określa klienta, który ma przydzielonego co najwyżej jednego doradcę,
- *IDP* $\rightarrow$ *NazwaP*, *CenaP*, ponieważ produkt ma unikalny identyfikator,
- *IDK*, *IDP*, *DataPZ* $\rightarrow$ *DataZZ, Liczba*, ponieważ klient może zamówić jeden produkt raz dziennie (w przypadku zamówienia kilku sztuk produktu zwiększana jest *Liczba*)

Wniosek: wszystkie składniki relacji *Zamówienie* zależą funkcjonalnie od atrybutów *IDK, DataPZ, IDP,* więc stanowią one klucz relacji. Relacja o takim schemacie jest w I FN.

II forma normalna relacji

Relacja R jest w **drugiej formie normalnej (II FN)** jeżeli jest w I FN i każdy ze składników, który nie jest elementem klucza, jest w zależności funkcjonalnej elementarnej od klucza (zależy funkcjonalnie od klucza i nie jest zależny od jego części).

Uwaga: Sprawdzenie IIFN dotyczy wyłącznie relacji, których klucz składa się z kilku atrybutów. W przypadku relacji z kluczem jednoelementowym spełnienie waruneku zależności elementarnej wynika z istnienia zależności funkcjonalnej.

Transformacja relacji z I FN do II FN

- Wyznaczenie wszystkich zależności elementarnych od składników klucza,
- Dekompozycja (podział) relacji pierwotnej poprzez wyłączenie atrybutów, które nie były zależne elementarnie od klucza (źródło zależności elementarnej pozostaje w relacji),
- Utworzenie nowej relacji dla każdego źródła zależności elementarnej zawierającej fragment klucza relacji pierwotnej oraz atrybuty, które były od niego zależne.

Uwaga: Jedna z relacji powstających w wyniku dekompozycji powinna mieć klucz zgodny z relacją pierwotną. W szczególnym przypadku może być pozbawiona dodatkowych atrybutów, nie może jednak zanikać, ponieważ opisuje istotne z punktu widzenia modelu powiązanie pomiędzy rzeczywistymi obiektami.

II forma normalna relacji – analiza przykładu

Relacja *Zamówienie (Z)* w I FN

*Z (**IDK**, NazwaK, PESEL, NazwiskoD, ImięD, **IDP**, NazwaP, CenaP, **DataPZ**, DataZZ, Liczba)*

Analiza

Źródła zależności elementarnych od składników klucza (patrz graf zależności s.21)

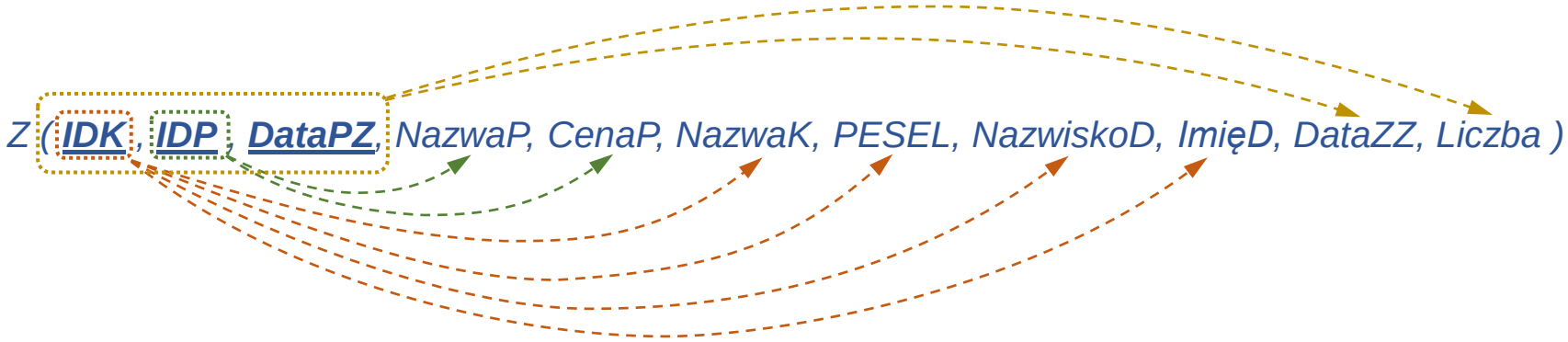
- *IDK* $\rightarrow$ *NazwaK, PESEL, NazwiskoD, ImięD*,
- *IDP* $\rightarrow$ *NazwaP, CenaP,*
- *IDK, IDP, DataPZ* $\rightarrow$ *DataZZ, Liczba*.

Wnioski

- Relacja nie jest w II FN, ponieważ atrybuty *NazwaK, PESEL, NazwiskoD, ImięD, NazwaP, CenaP* nie są zależne elementarnie od klucza.
- Należy wykonać dekompozycję pierwotnej relacji *Zamówienie* tworząc trzy relacje pochodne (istnieją trzy źródła zależności elementarnych).

II forma normalna – dekompozycja

W celu doprowadzenia relacji do II FN należy dokonać dekompozycji na trzy relacje, których klucze stanowią składniki będące źródłem zależności elementarnych (s.23).



Dekompozycja

- pierwsze źródło zależności elementarnej (*IDK*):

  *Z1(**IDK**, NazwaK, PESEL, NazwiskoD, ImięD)*

- drugie źródło zależności elementarnej (*IDP*):

  *Z2(**IDP**, NazwaP, CenaP)*

- trzecie źródło zależności elementarnej (*IDK, IDP, DataPZ*):

  *Z3(**IDK**, **IDP**, **DataPZ**, DataZZ, Liczba)*

III forma normalna relacji

Relacja R jest w **trzeciej formie normalnej (III FN)** jeżeli jest w II FN i każdy ze składników, który nie jest elementem klucza jest w zależności funkcjonalnej bezpośredniej od klucza.

Transformacja relacji z II FN do III FN

- Wyznaczenie wszystkich zależności bezpośrednich pomiędzy atrybutami relacji (niezależnie czy są one składnikami klucza),
- Dekompozycja (podział) relacji pierwotnej poprzez wyłączenie atrybutów, które nie były zależne bezpośrednio (źródło zależności bezpośredniej pozostaje w relacji),
- Utworzenie nowej relacji dla każdego źródła zależności bezpośredniej zawierającej źródło zależności (jako klucz) oraz wszystkie atrybuty, które były od niego zależne bezpośrednio.

Uwaga: Źródła zależności bezpośrednich stają się kluczami nowopowstałych relacji, jednak pozostają również w relacji pierwotnej. Ich obecność gwarantuje utrzymanie powiązania pomiędzy tworzonymi relacjami.

III forma normalna relacji – analiza przykładu

Zestaw relacji w II FN

*Z1(**IDK**, NazwaK, PESEL, NazwiskoD, ImięD)*

*Z2(**IDP**, NazwaP, CenaP)*

*Z3(**IDK**, **IDP**, **DataPZ**, DataZZ, Liczba)*

Analiza

- Relacje *Z2* i *Z3* są w III FN (nie ma zależności pośrednich)
- Relacja *Z1* nie spełnia warunków III FN (istnieje zależność pośrednia)

  istnieje zależność funkcjonalna (patrz graf s.21): $PESEL \rightarrow NazwiskoD, ImięD,$

  stąd: $IDK \rightarrow PESEL, \quad PESEL \rightarrow NazwiskoD, ImięD,$

  tzn. istnieją składniki (*NazwiskoD*, *ImięD*), które nie są zależne bezpośrednio od *IDK*.

Wniosek

Należy wykonać dekompozycję relacji *Z1* wyodrębniając nazwisko i imię doradcy do osobnej relacji.

III forma normalna – dekompozycja

W celu doprowadzenia relacji do III FN należy dokonać dekompozycji na dwie relacje, których kluczem zostaną składniki będące źródłem zależności bezpośrednich (patrz graf zależności funkcjonalnych s.21).

Dekompozycja

- pierwsze źródło zależności bezpośredniej (*IDK*):

  *Z1-1(**IDK**, NazwaK, PESEL)*

- drugie źródło zależności bezpośredniej (*PESEL*):

  *Z1-2(**PESEL**, NazwiskoD, ImięD)*

Podsumowanie – znormalizowany zestaw relacji

Proces kolejnych transformacji, którym poddano pierwotną relację *Zamówienie* (s.8) określa się normalizacją danych. Jego celem jest uzyskanie optymalnego modelu danych, pozbawionego redundancji i pozostałych anomalii (s.9).

Wynikiem normalizacji jest zestaw relacji w III FN postaci:

*Z1-1(**IDK**, NazwaK, PESEL)*

*Z1-2(**PESEL**, NazwiskoD, ImięD)*

*Z2(**IDP**, NazwaP, CenaP)*

*Z3(**IDK**, **IDP**, **DataPZ**, DataZZ, Liczba)*

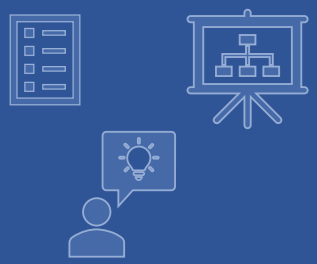
Ostatnim krokiem modelowania jest przyjęcie nazw relacji, które określają ich faktyczną zawartość. Ostateczny zestaw relacji przyjmuje postać:

*Klient(**IDK**, NazwaK, PESEL)*

*Doradca(**PESEL**, NazwiskoD, ImięD)*

*Produkt(**IDP**, NazwaP, CenaP)*

*Zamówienie(**IDK**, **IDP**, **DataPZ**, DataZZ, Liczba)*



Analiza przykładu

Przykładowy zestaw danych

Klucze obce:
- IDK
- IDP

Zamówienie (Z)

IDK	IDP	DataPZ	DataZZ	Liczba
0224111111	W150	25.01.10	25.01.15	100
0224111111	P250	25.01.10	25.01.13	50
0441222222	W150	25.01.14		200
0441222222	W100	25.01.14	25.01.17	50
0441222222	P500	25.01.14	25.01.15	200
0224111111	W100	25.01.14	25.01.15	80
0441222222	W150	25.01.21	25.01.23	100
0224111111	P200	25.01.30		125

Produkt (P)

IDP	NazwaP	CenaP
P200	Płaskownik 200	10
P250	Płaskownik 250	15
P500	Płaskownik 500	30
W100	Wałek 100	20
W150	Wałek 150	25

Klient (K)

IDK	NazwaK	PESEL
0224111111	ABC Sp.zo.o.	90052000234
0441222222	XYZ S.A.	90052000234

Klucz obcy: PESEL

Doradca (D)

PESEL	NazwiskoD	ImięD
90052000234	Kowalska	Anna
82012090803	Nowak	Andrzej

Rozmiar bazy danych

Pierwotny schemat relacji (s.8)

Z (IDK, NazwaK, PESEL, NazwiskoD, ImięD, IDP, NazwaP, CenaP, DataPZ, DataZZ, Liczba)

Znormalizowany zestaw relacji (s.29)

Klient(IDK, NazwaK, PESEL)

Doradca(PESEL, NazwiskoD, ImięD)

Produkt(IDP, NazwaP, CenaP)

Zamówienie(IDK, IDP, DataPZ, DataZZ, Liczba)

Rozmiary atrybutów (w znakach): IDK (11), NazwaK (50), PESEL (11), NazwiskoD (15), ImięD (15), IDP (10), NazwaP (25), Cena (10), DataPZ (10), DataZZ (10), Liczba (5).

Pierwotna relacja: $(11+50+15+15+10+25+10+10+10+5)\times 8 = 161\times 8 = 1288$

Relacje znormalizowane: $(11+50+11)\times 2+(11+15+15)\times 2+(10+25+10)\times 5+(11+10+10+10+5)\times 8 =$

$72\times 2 + 41\times 2 + 45\times 5 + 46\times 8 = 144 + 82 + 225 + 368 = 819$

Uwaga: każde kolejne zamówienie powiększa rozmiar bazy danych o:

- *161 znaków w przypadku relacji pierwotnej,*
- *46 znaków w przypadku relacji znormalizowanych.*

Analiza rozwiązania znormalizowanego

Cechy znormalizowanego modelu danych:

1. Optymalny zestaw relacji, który pozwala wykonać aplikację zgodną z pierwotnym przeznaczeniem, realizującą zaplanowane funkcje (s.8).

2. Większa elastyczność modelu danych, łatwa rozbudowa schematów relacji bez nadmiernego powiększania rozmiaru bazy danych (np. uzupełnienie relacji *Klient*).

3. Wyeliminowanie redundancji, znacząca redukcja rozmiaru bazy danych w porównaniu do pierwotnej formy relacji (różnice tym większe im większa liczba zamówień).

4. Wyeliminowanie niespójności danych, np. określenie doradcy klienta nie wymaga analizy całej bazy.

5. Brak anomalii przy wstawianiu, aktualizacji i usuwaniu, baza może zawierać dane wszystkich produktów firmy (niezależnie czy były kiedykolwiek zamówione), aktualizacja danych osobowych doradców nie wymaga zmian w wielu wierszach, dane doradców odseparowane od zamówień.

CREATE DATABASE

USE

CREATE TABLE

ALTER FOREIN KEY

Operacje na strukturach danych (DDL)

Manipulowanie bazą danych

Wyświetlenie istniejących baz danych

```
SHOW DATABASES;
```

Tworzenie nowej bazy danych

```
CREATE DATABASE <nazwa>;
```

Usuwanie istniejącej bazy danych

```
DROP DATABASE <nazwa>;
```

Wybór bazy danych

```
USE <nazwa>;
```

Kopia bazy danych

```
BACKUP DATABASE <nazwa> TO DISK = <nazwa_pliku>
[WITH DIFFERENTIAL];
```

Opcjonalna klauzula **WITH DIFFERENTIAL** oznacza wykonanie kopii różnicowej, uwzględniającej wyłącznie elementy, które uległy zmianie od ostatniej kopii bazy.

Uwaga: <nazwa> powinna być dozwolonym i unikalnym identyfikatorem (polecenie **CREATE**) lub nazwą istniejącej bazy danych (polecenia **DROP, USE** i **BACKUP**).

Tworzenie tabel

Tworzenie nowej tabeli

```
CREATE TABLE <nazwa> (
     <atrybut1> typ_danych więzy_spójności,
     <atrybut2> typ_danych więzy_spójności,
     …
     <atrybutN> typ_danych więzy_spójności
);
```

Tworzenie tabeli na podstawie wyniku kwerendy

```
CREATE TABLE <nazwa> AS
     SELECT atrybuty
     FROM tabela
     WHERE … ;
```

Uwaga: `tabela` musi być nazwą istniejącego źródła (źródeł) danych. Jeżeli źródło znajduje się w innej bazie danych niż bieżąca należy użyć konstrukcji `nazwa_bazy.nazwa_źródła`.

Typy danych (MySQL)

Typy znakowe

- `CHAR(N)` – ciąg znaków o stałej długości do 255 znaków,
- `VARCHAR(N)` – ciąg znaków o zmiennej długości do 65535 znaków.

Typy całkowite

- `TINYINT [UNSIGNED]` – liczba całkowita -127..128 (0..255),
- `SMALL [UNSIGNED]` – liczba całkowita -32768..32767 (0..65535),
- `INT [UNSIGNED]` – liczba całkowita −2147483648..2147483647 (0..4294967295).

Typy rzeczywiste

- `DECIMAL(N,D)` – liczba rzeczywista stałoprzecinkowa, N cyfr, D cyfr po przecinku,
- `FLOAT` – rzeczywista zmiennoprzecinkowa, pojedyncza prec. −1,4e+38 .. 3,4e+38,
- `DOUBLE` – rzeczywista zmiennoprzecinkowa, podwójna prec. −1,8+308 .. −1,8+308.

Typy daty i czasu

- `DATE` – data w formacie `rrrr-mm-dd`, 1000-01-01..9999-12-31,
- `TIME` – czas w formacie `gg:mm:ss`, -838:59:59..838:59:59
- `DATETIME` – data i czas w formacie `rrrr-mm-dd gg:mm:ss`.

Więzy spójności (ograniczenia)

Więzy spójności specyfikują reguły określające poprawność danych poprzez nałożenie dodatkowych warunków na wartości, które mogą być wprowadzone do bazy. Są sprawdzane przed wykonaniem dowolnej operacji na danych (dodawanie, modyfikacja, usuwanie) i przerywają ją w przypadku wykrycia naruszenia reguł.

Więzy spójności w SQL

- `NOT NULL` – zabroniona wartość `NULL`, wartość musi być wprowadzona,
- `UNIQUE` – wartość atrybutu musi być unikalna,
- `PRIMARY KEY` – klucz główny (`NOT NULL+UNIQUE`), wartość jednoznacznie identyfikuje krotki tabeli,
- `FOREIGN KEY` – klucz obcy, dozwolone wartości, które stanowią klucz główny w innej tabeli,
- `CHECK` – wartość atrybutu musi spełniać warunek (dozwolone kilka warunków),
- `DEFAULT` – domyślna wartość atrybutu.

PRIMARY KEY, FOREIGN KEY, DEFAULT, CHECK (MySQL)

Składnia ograniczeń PRIMARY KEY

```
CREATE TABLE <nazwa> (
   <atr1> typ PRIMARY KEY,
   <atr2> typ, …
   <atrN> typ
);
```

```
CREATE TABLE <nazwa> (
   <atr1> typ, …
   <atrN> typ,
   PRIMARY KEY (lista_atrybutów)
);
```

Składnia ograniczeń FOREIGN KEY

```
CREATE TABLE <nazwa> (
   <atr1> typ PRIMARY KEY,
   <atr2> typ, …
   <atrN> typ,
   FOREIGN KEY (atrybut) REFERENCES tabela(atrybut)
);
```

Składnia ograniczeń CHECK i DEFAULT

```
<atrybut> typ_danych CHECK (warunek)
<atrybut> typ_danych DEFAULT (wartość | wyrażenie)
```

CREATE TABLE – przykłady

Nowa baza danych test i tabela "Klient".

```
CREATE DATABASE test;
USE test;
CREATE TABLE Klient (
    IDK CHAR(11) PRIMARY KEY CHECK (LENGTH(IDK)>=10) ,
    NazwaK VARCHAR(50) NOT NULL,
    PESEL VARCHAR(11),
    FOREIGN KEY (PESEL) REFERENCES Doradca(PESEL)
);
```

Uwaga: baza danych musi zawierać tabelę Doradca, w której kluczem głównym jest PESEL.

Tabela „Klient” jako kopia istniejącej tabeli „K”.

```
CREATE TABLE Klient AS
     SELECT *
     FROM moja_baza.K;
```

Uwaga: „moja_baza” jest przykładową nazwą bazy, która zawiera tabelę "K" i jest dostępna dla użytkownika wykonującego polecenie `CREATE`.

Modyfikacja struktury tabeli

Dodawanie kolumny

```
ALTER TABLE <nazwa_tabeli>
ADD <atrybut> typ_danych;
```

Usuwanie kolumny

```
ALTER TABLE <nazwa_tabeli>
DROP COLUMN atrybut;
```

Modyfikacja kolumny (MySQL)

```
ALTER TABLE <nazwa_tabeli>
MODYFY COLUMN <atrybut> typ_danych;
```

Uwaga: postać instrukcji modyfikującej kolumnę zależy od serwera bazy danych:

- **MODYFY COLUMN** w MySQL i starszych wersjach Oracle,
- **MODYFY** w nowych wersjach Oracle,
- **ALTER COLUMN** w SQL Server.

Modyfikacja struktury tabeli – przykład

Modyfikacja bazy "test":

- dodanie kolumny do tabeli "Klient",
- modyfikacja typu dodanej kolumny,
- usunięcie dodanej kolumny.

```
USE test;

ALTER TABLE Klient                  <-- dodanie kolumny Adres
ADD Adres CHAR(50);                     do tabeli Klient

ALTER TABLE Klient                  <-- zmiana typu kolumny
MODIFY COLUMN Adres VARCHAR(75);        z CHAR(50) na VARCHAR(75)

ALTER TABLE Klient                  <-- usunięcie pola
DROP COLUMN Adres;                      Adres z tabeli
```

Uwaga: Instrukcja **`ALTER TABLE`** pozwala również na dodawanie, modyfikację i usuwanie ograniczeń (*s.28*). Składnia operacji jest zależna od typu ograniczenia.

INSERT

UPDATE

DELETE

Operacje na danych (DML)

Manipulowanie danymi

Wstawianie krotek

```
INSERT INTO <nazwa_tabeli> (atrybut1, atrybut2,…,atrybutN)
VALUES (wartość1-1, wartość1-2, …,wartość1-N),
       (wartość2-1, wartość2-2, …,wartość2-N),…;
```

Uwaga: nazwy atrybutów mogą być pominięte, jeżeli instrukcja wstawia wartości do wszystkich kolumn i wartości są wymienione w kolejności zgodnej z definicją w tabeli.

Aktualizacja krotek

```
UPDATE <nazwa_tabeli>
SET atrybut1 = wartość1, atrybut2 = wartość2,…
WHERE warunek;
```

Usuwanie krotek

```
DELETE FROM <nazwa_tabeli>
WHERE warunek;
```

Uwaga: klauzula **WHERE** w instrukcjach **UPDATE** I **DELETE** jest opcjonalna, brak oznacza aktualizację/usunięcie wszystkich krotek tabeli (składnia *warunku* jak w **SELECT**).

Manipulowanie danymi – przykłady

Dodanie nowych klientów

```
INSERT INTO Klient(IDK, NazwaK, PESEL)
VALUES('3847561920', 'AutoParts Polska', '95102203451'),
      ('5678901234', 'Optima Logistics', '95102203451');
```

Zmiana doradcy klienta o id **5678901234**

```
UPDATE Klient
SET PESEL = '90052000234'
WHERE IDK = '5678901234';
```

Usunięcie klienta o id **5678901234**

```
DELETE FROM Klient
WHERE IDK = '5678901234';
```