Spis treści

[ZAŁOŻENIA 2](#_Toc134439078)

[Tabele: 2](#_Toc134439079)

[Relacje: 2](#_Toc134439080)

[Schemat relacyjny 3](#_Toc134439081)

[INSTALACJA 4](#_Toc134439082)

[OPERACJE HURTOWE 4](#_Toc134439083)

[Baza imion i nazwisk 4](#_Toc134439084)

[Hurtowe dodawanie agentów: 4](#_Toc134439085)

[Hurtowe dodawanie polis: 5](#_Toc134439086)

[Hurtowe dodawanie szkód: 5](#_Toc134439087)

[GENERATORY 5](#_Toc134439088)

[OPERACJE DETALICZNE 6](#_Toc134439089)

[Dodawanie agenta: 6](#_Toc134439090)

[Dodawanie polisy: 6](#_Toc134439091)

[Dodawanie szkody: 6](#_Toc134439092)

[FUNKCJONALNOŚCI BAZY 6](#_Toc134439093)

[Widoki zmaterializowane 6](#_Toc134439094)

[Widoki 7](#_Toc134439095)

[Indexy 8](#_Toc134439096)

[CIEKAWE MIEJSCA w PROGRAMIE 8](#_Toc134439097)

[Tabela Kontrahenci 8](#_Toc134439098)

[Triggery LOGON I LOGOFF 9](#_Toc134439099)

[Do Zrobienia 9](#_Toc134439100)

# ZAŁOŻENIA

Projekt został wykonany w środowisku Oracle Express Edition 21c

Tematem jest baza danych, która symuluje (w dużym uproszczeniu) działanie towarzystwa ubezpieczeniowego.

Możliwe jest dodawanie, usuwanie oraz modyfikacja polis, agentów oraz osób wraz z powiązanymi danymi z innych tabel.

Każda polisa jest powiązana z 1 agentem, polisa może obejmować 1 lub wiele osób.

Każdy agent może wprowadzić wiele polis.

Do każdej polisy są przypisane osoby: jedna jako ubezpieczający oraz jedna lub wiele jako ubezpieczeni.

Z każdej polisy można zgłosić jedną lub wiele szkód.

Szkoda ma 4 możliwe statusy: zgłoszona (1) , rozpatrywana (2), odrzucona (3), wypłacona (4).

Szkoda powinna być rozpatrzona w czasie 14 dni od zgłoszenia

Baza zapisuje w tabeli info\_log informacje dotyczące logowania i wylogowania użytkowników (triggery LOGON, LOGOFF)

Baza zapisuje w tabeli info\_dane informacje dotyczące pracy z danymi w tabelach agenci, polisy, szkody (trigger DML)

Baza 1 raz dziennie uruchamia job o nazwie „job\_stat” który odświeża statystyki tabel faktów (tabele polisy, kontrahenci, szkody) oraz zapisuje dane o liczbie ich wierszy do tabeli stat\_info.

Baza 1 raz dziennie odświeża widok zmaterializowany o nazwie „mv\_polisy\_koniec” który zawiera informacje nt. polis oraz ich właścicieli, dla polis których termin ważności upływa w ciągu 7 dni

## Tabele:

agenci

imiona\_meskie - baza imion

imiona\_zenskie - baza imion

info\_log - info na temat logowania / wylogowania

kontrahenci - tabela pośrednia polisy-osoby

mv\_polisy\_koniec - widok zmaterializowany – kończące się polisy

nazwiska\_meskie - baza nazwisk

nazwiska\_zenskie - baza nazwisk

osoby

polisy (podzielona na partycje względem lat)

rola

stat\_info - tabela z informacjami dot. statystyk fact tables

szkody (podzielona na partycje względem lat)

szkody\_status - opisy statusów szkód

## Relacje:

agenci polisy 1:M

osoby kontrahenci 1:M

polisy kontrahenci 1:M

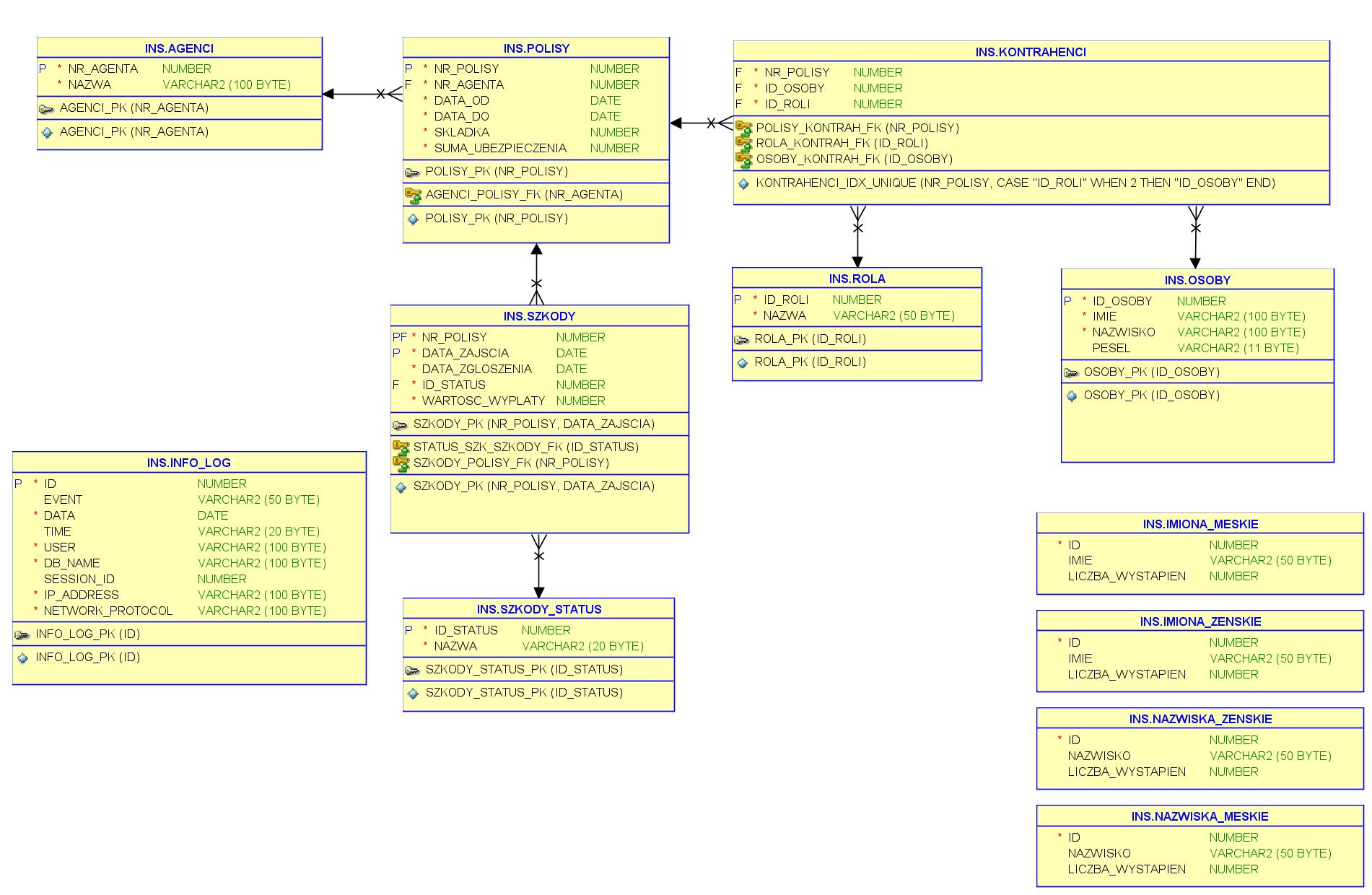
polisy szkody 1:M

szkody\_status szkody 1:M

Baza posiada możliwość indywidualnego oraz hurtowego dodawania danych (w celu uzyskania szybkiego przyrostu informacji w bazie)

(funkcjonalność w trakcie tworzenia, na chwilę obecną można hurtowo dodać agentów)

## Schemat relacyjny



# INSTALACJA

Instalacja bazy danych może odbyć się na 2 sposoby:

- poprzez uruchomienie w bazie danych skryptu z pliku **00\_create\_db.sql** (należy ustawić w nim swoje parametry w sekcji USTAWIENIA BAZY, z tego pliku uruchamiane są kolejne skrypty) Instalacja na samym początku sprawdza czy istnieje podany schemat użytkownika w wybranej lokalizacji i jeśli tak to go usuwa po czym tworzy od nowa użytkownika i wszystkie potrzebne składniki.

- poprzez import pliku insurance\_schema.dmp wygenerowanego przez expdp, zawierającego ostatnią wersję schematu INS z bazą danych (w tym wypadku należy mieć już utworzony schemat użytkownika z nadanymi odpowiednimi uprawnieniami)

Import można wykonać za pomocą pliku /datapump/ 00\_import\_schema\_remap.bat, natomiast utworzony wcześniej schemat powinien mieć następujące uprawnienia (dane oczywiście trzeba dostosować do swojej bazy):

create user ins identified by ins;

alter user ins quota 100M on users;

create or replace directory ins\_datapump AS ' sciezka do katalogu z plikiem dmp ';

create or replace directory ins\_external\_table AS ' sciezka do katalogu external\_table';

grant connect, resource to ins;

grant read, write on directory ins\_external\_table to ins;

grant read, write on directory ins\_datapump to ins;

grant create view to ins;

grant create materialized view to ins;

grant create trigger to ins;

GRANT CREATE JOB TO ins;

Ustawienia językowe bazy danych to EE8PC852

# OPERACJE HURTOWE

## Baza imion i nazwisk

Baza korzysta z tabel imion i nazwisk które powstały poprzez przeniesienie pierwszych 2000 wierszy z zewnętrznej bazy imion oraz nazwisk (pliki csv)

https://dane.gov.pl/pl/dataset/1681,nazwiska-osob-zyjacych-wystepujace-w-rejestrze-pesel

https://dane.gov.pl/pl/dataset/1501

Dane źródłowe są posortowane od najczęściej do najrzadziej występujących.

Najpierw tworzone są tabele zewnętrzne (external tables) aby uzyskać dostęp do danych z plików \*.csv

następnie na ich podstawie tworzone są właściwe tabele ograniczone do 2000 wierszy z najczęściej występującymi imionami i nazwiskami

**Tabele**

- imiona\_meskie

- imiona\_zenskie

- nazwiska\_meskie

- nazwiska\_zenskie

## Hurtowe dodawanie agentów:

**agenci\_pkg.dodaj\_agentow\_hurt(**

p\_ilosc:=1, - ilość agentów którzy mają być dodani

p\_nazwa\_agenta:=’Agent’, - nazwa agenta, domyślnie ‘Agent’

p\_autonum:=TRUE - czy do nazwy agenta ma być dodawany kolejny numer

(Agent 001, Agent 002 itd.)

);

Procedura sprawdza max numer agenta jaki istnieje w tabeli agenci i dodaje na kolejnych miejscach nowych agentów. Domyślna nazwa to ‘Agent’. Domyślnie dodawana jest autonumeracja po nazwie

Domyślne nazwy agentów z autonumeracją to 'Agent 00x' , gdzie 'x' = nr\_agenta (generowany przez sekwencję identity)

**Wszystkie funkcje pakietu agenci\_pkg:**

- dodaj\_agenta (pub) - dodaj podaną liczbę agentów do tabeli agenci

- dodaj\_agenta\_hurt (pub) - dodaj podaną liczbę agentów do tabeli agenci

- wylosuj\_agenta (pub) - zwraca losowego agenta spośród dostępnych

- ustaw\_seq (priv) - ustawia seq na ostatni nr\_agenta w tabeli agenci (automatyczne

dodawanie agentów przydziela wtedy nr\_agenta zaraz po ostatnim nr\_agenta w tabeli, niezależnie od tego jaką aktualnie wartość ma sekwencja kolumny nr\_agenta)

## Hurtowe dodawanie polis:

**polisy\_pkg.dodaj\_polise\_hurt** (

p\_ilosc\_polis number, - ilość polis które chcemy dodać

p\_data\_polisy\_od date, - najstarsza możliwa data dla początku polisy

p\_data\_pollisy\_do date, - najnowsza możliwa data dla początku polisy

p\_suma\_min number, - minimalna suma ubezpieczenia

p\_suma\_max number, - maksymalna suma ubezpieczenia

p\_ilosc\_osob number, - maksymalna ilość osób, które mogą być na 1 polisie

p\_data\_osoby\_od date, - data urodzenia najmłodszej możliwej osoba na polisach p\_pesel\_osoby\_do date, - data urodzenia najstarszej możliwej osoba na polisach

p\_procent number - szansa (0-100) że ubezpieczający będzie też na tej samej

polisie ubezpieczonym

);

Składka jest liczona automatycznie jako 1% sumy ubezpieczenia

Procedura wypełnia 3 kolekcje

- dla polis

- dla osób

- dla kontrahentów

Dodanie polisy musi dodawać co najmniej 1 wpis w tabeli osoby i co najmniej 2 wpisy w tabeli kontrahenci (1 osoba która kupuje polisę jako ubezpieczający i 1 osoba jako ubezpieczony, może to być ta sama osoba)

## Hurtowe dodawanie szkód:

**Pakiet szkody\_pkg.dodaj\_szkody(ilosc,max\_ilosc\_szkod) – do zrobienia**

Z tabeli polisy losowane są numery polis na których zostanie zgłoszona szkoda oraz jej status

Dla każdej wylosowanej polisy zostaną wygenerowana od 1 do n ilość szkód (n=max\_ilość\_szkód)

Tylko dla szkód ze statusem zatwierdzona zostanie wpisana wyplata w losowej kwocie z zakresu od 0 do suma\_ubezpieczenia

# GENERATORY

Baza posiada pakiet generatory\_pkg, który służy do generowania wartości potrzebnych do hurtowego wypełniania tabel. Składnikami pakietu są funkcje, które jako wynik zwracają wygenerowane wartości.

Składniki pakietu GENERATORY\_PKG:

- generuj\_dane\_osobowe: na podstawie numeru pesel generuje rekord z danymi osobowymi (imię, nazwisko, PESEL) korzystając z bazy danych imion i nazwisk. Funkcja rozpoznaje czy PESEL należy do mężczyzny czy kobiety (10 cyfra identyfikuje płeć) i generuje dane w oparciu o odpowiednie tabele (imiona\_meskie, imiona\_zenskie itd.)

- generuj\_date: generuje datę spośród podanego zakresu dat (zamienia daty na dni w oparciu o kalendarz juliański i generuje losową liczbę z podanego zakresu)

- generuj\_pesel: generuje dla podanego zakresu dat prawidłowy PESEL wraz z cyfrą kontrolną

Obsługuje lata 1900 – 2099.

- generuj\_sume\_ubezp: generuje sumę ubezpieczenia dla podanego zakresu kwot i zwraca wartość zaokrągloną do całych setek

# OPERACJE DETALICZNE

## Dodawanie agenta:

**Pakiet agenci\_pkg**

dodaj\_agenta(p\_nazwa\_agenta,p\_autonum:=FALSE)

- dodaje 1 agenta o wybranej nazwie wywołując wewnątrz procedurę dodaj\_agentow\_hurt

- somyślnie nie dodaje autonumeracji po nazwie

## Dodawanie polisy:

**Pakiet polisy\_pkg**

polisy\_pkg\_dodaj\_polise (

p\_nr\_agenta number, - numer agenta do którego będzie przypisana polisa

p\_data\_pol\_od date, - data początku polisy

p\_data\_pol\_do date, - data końca polisy

p\_suma\_ubezp number - suma ubezpieczenia na polisie

);

## Dodawanie szkody:

**Pakiet szkody\_pkg.dodaj\_szkode**

1) dodać szkodę do wybranej polisy

# FUNKCJONALNOŚCI BAZY

## Widoki zmaterializowane

1. **mv\_polisy\_koniec**

– polisy których termin ważności kończy się do 7 dni. Widok jest odświeżany codziennie o północy

1. mv\_kwoty\_odszkodowan – do zrobienia
   1. kwoty wypłaconych odszkodowań w podziale na m-ce wg daty zgłoszenia w wybranym zakresie czasu
2. mv\_sr\_czas\_wyplaty - – do zrobienia
   1. średni czas (w dniach) jaki upłynął od zajścia szkody do jej zgłoszenia w wybranym kwartale które zakończyły się wypłatą
3. mv\_bilans\_roczny - – do zrobienia
   1. suma wpłaconych składek do sumy wypłaconych odszkodowań w podziale na m-ce w wybranym roku

## Widoki

1. **v\_polisy\_osoby**
   1. dane wszystkich osób + liczba polis, gdzie osoby występują jako 'ubezpieczający'
2. **v\_polisy\_wlasciciele**
   1. dane tylko tych osób które są właścicielami polis + ilość powiązanych polis
3. **v\_polisy\_indywidualne**
   1. numer polis na których jest tylko 1 osoba ubezpieczona (dowolna osoba)
4. **v\_polisy\_indywiduane\_wlascicieli**
   1. numery polis dla których ubezpieczający i ubezpieczany to ta sama osoba i nie ma innych osób na polisie 🡪1 osoba na polisie i jest to ubezpieczający
5. **v\_agenci\_wskazniki**
   1. wszyscy agenci (również ci bez polis) wraz z ilością sprzedanych polis oraz wskaźnikiem, ile tych polis jest aktywnych na dzisiaj
6. **v\_szkody\_przeterminowane**
   1. szkody, które zaszły w okresie ubezpieczenia a zostały zgłoszone po zakończeniu okresu ubezpieczenia
7. **v\_szkody\_wysokie**
   1. liczba szkód oraz wartość wypłaconych odszkodowań pogrupowanych według peseli osób, które zawarły ubezpieczenia (występują w roli ubezpieczający) Na zestawieniu są tylko te pozycje, dla których łączna wartość wypłat przekracza 100 tys.
8. **v\_polisy\_przekroczona\_wartosc**
   1. polisy, do których zostały zgłoszone szkody i dla których wartość wypłaty przekracza sumę ubezpieczenia.
9. **v\_polisy\_bez\_szkod**
   1. polisy, do których nie została zgłoszona szkoda.
10. **v\_szkody\_opoznione**
    1. szkody, które zostały zgłoszone a nie zostały rozpatrzone w czasie 7 dni
11. **v\_wskazniki\_szkodowosci**
    1. stosunek (wyrażony w %) liczby polis ze szkodami do łącznej liczby polis oraz stosunek (wyrażony w %) łącznej kwoty wypłaty odszkodowania do łącznej sumy ubezpieczenia z polis, do których została zgłoszona co najmniej jedna szkoda.
12. **v\_bilans\_roczny – do zrobienia**
    1. ilość wpłaconych składek do ilości wypłaconych odszkodowań w podziale na lata

# Indeksy

W bazie zostały utworzone dodatkowe indeksy:

B-Tree Index:

polisy\_data\_od\_idx on polisy(data\_od)

polisy\_data\_do\_idx on polisy(data\_do)

polisy\_nr\_agenta\_idx on polisy(nr\_agenta)

szkody\_wart\_wypl\_idx on szkody(wartosc\_wyplaty)

szkody\_data\_zgloszenia\_idx on szkody(data\_zgloszenia)

kontrahenci\_id\_osoby\_idx on kontrahenci(id\_osoby)

Bitmap Index:

kontrahenci\_id\_roli\_idx on kontrahenci(id\_roli);

# CIEKAWE MIEJSCA w PROGRAMIE

## Tabela Kontrahenci

Jest to tabela pośrednia łącząca polisy z osobami

Pola tabeli: nr\_polisy, id\_osoby, id\_roli (1-ubezpieczający, 2-ubezpieczony).

Początkowo na tych wszystkich polach był zdefiniowany Primary Key (Composite PK)

Wymaga on jednak podania już w czasie definicji tabeli wszystkich kolumn, w których wartości mają być sprawdzane pod kątem unikalności.

Aby móc zachować założoną funkcjonalność tabeli, czyli dla każdej polisy:

- może być tylko 1 osoba jako ubezpieczający

- może być wiele osób jako ubezpieczeni

nie można użyć PK, gdyż:

dla PK(nr\_polisy)

- nie można dodać do tej samej polisy więcej niż 1 osoby

dla PK(nr\_polisy, id\_osoby)

- nie można dodać do polisy tej samej osoby jako ubezpieczający i ubezpieczony

- można dodać do polisy kilka różnych osób i wszystkie jako ubezpieczający

dla PK(nr\_polisy, id\_osoby, id\_roli)

- można dodać do polisy różne osoby jako ubezpieczający

**Utworzenie Indexu Unique zamiast PK rozwiązuje sprawę:**

CREATE UNIQUE INDEX Idx\_Kontrahenci\_Unique on kontrahenci

(

nr\_polisy,

CASE WHEN id\_roli=2 THEN id\_osoby END

);

nr\_polisy jest zawsze sprawdzany

id\_osoby jest dodawane do sprawdzania gdy id\_roli=2 (ubezpieczony)

## Triggery LOGON I LOGOFF

Próba nadania w triggerze automatycznej nazwy schematu w zależności od nazwy użytkownika (np. INS.SCHEMA 🡪 &&v\_user.SCHEMA) spowodowała wystąpienie błędu, gdyż wyrażenie „&&v\_user.” jest interpretowane jako próba złączenia zmiennej &&v\_user ze stringiem SCHEMA. Jeżeli &&v\_user= INS to wynikiem jest INSSCHEMA zamiast INS.SCHEMA.

Poprawny zapis wygląda następująco: INS.SCHEMA 🡪 &&v\_user..SCHEMA

Pierwsza kropka to znak łączenia, druga kropka odpowiada za .SCHEMA, czyli mamy INS.SCHEMA

# Do Zrobienia

Dodawanie szkód hurtowe i detaliczne