

BAZY DANYCH

Sprawozdanie z projektu

Prowadzący: mgr inż. Józef Woźniak

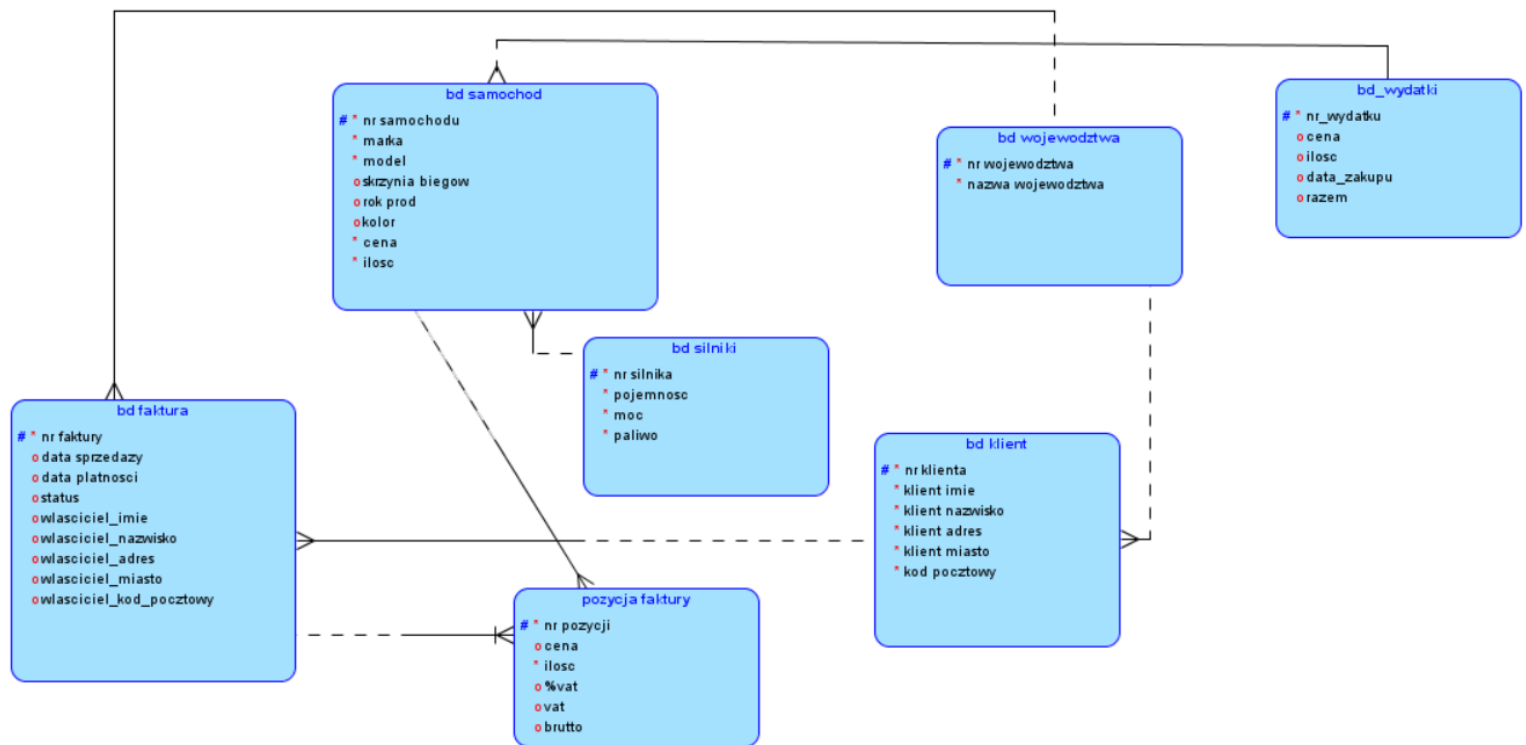
Student: Radosław Żurawicz

Grupa szkoleniowa: I7Y7S1

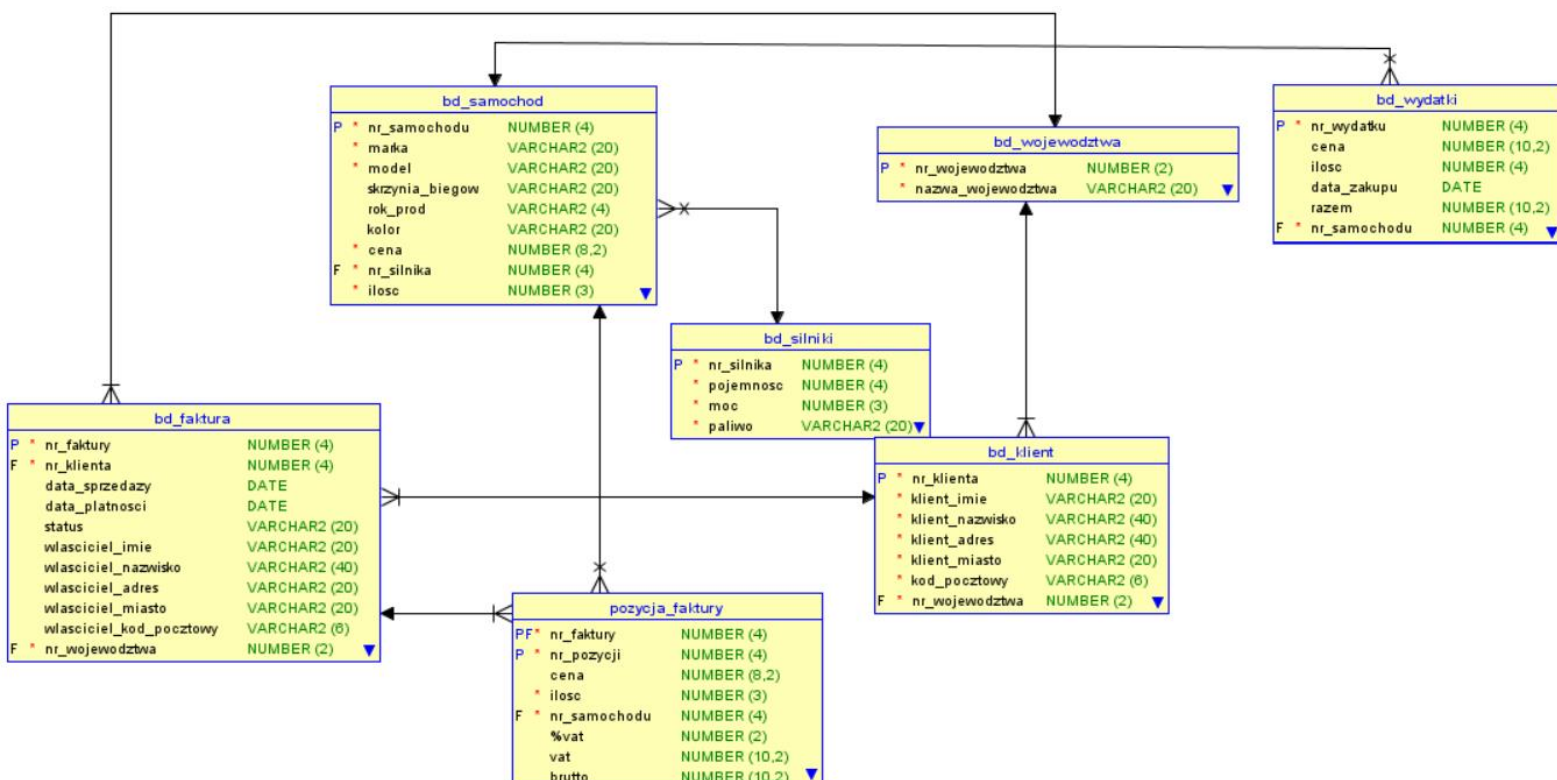
1. Treść zadania

Pewna osoba prowadzi salon samochodowy, który ma wielu klientów oraz prowadzi sprzedaż samochodów. Należy zamodelować tak fragment rzeczywistości, aby przedstawiał on transakcje pomiędzy klientami oraz salonem, możliwe było ewidencjonowanie zakupów, dostępnych samochodów.

2. Model logiczny i relacyjny



Rys. 1 – Model Logiczny



Rys. 2 – Model relacyjny

3. Oprogramowanie tworzące generator danych

Oprogramowanie tworzące generator danych

- procedura generuj_dane – główny zarządca

```
create or replace procedure generuj_dane
as
v_nr_klienta bd_klient.nr_klienta%type;
v_nr_faktury bd_faktura.nr_faktury%type;
v_data_sprzedazy_faktury date;
v_nr_samochodu bd_samochod.nr_samochodu%type;
v_ilosc bd_samochod.ilosc%type;
v_ile_samochodow_z_roku bd_samochod.nr_samochodu%type;
v_counter number;
v_losowa_ilosc_samochodu bd_samochod.nr_samochodu%type;
v_czy_jest_juz number;
v_zakonczenie_petli_ile number;
v_zakonczenie_petli_data date;
v_stop boolean;
begin

v_stop:=false;

while v_stop!=true loop

select count(*) into v_zakonczenie_petli_ile
from bd_faktura;

if v_zakonczenie_petli_ile=0 then
    v_zakonczenie_petli_data:='18/01/01';
else
    select max(data_sprzedazy) into v_zakonczenie_petli_data
    from bd_faktura;
end if;

if v_zakonczenie_petli_data+13<='19/12/31' then
v_nr_klienta:=fn_daj_numer_klienta();

insert into bd_faktura(nr_klienta)
values(
v_nr_klienta);
```

```

--pobieram nr faktury na ktorej dzialam
select nr_faktury into v_nr_faktury
from bd_faktura where status='OTWARTA';

select data_sprzedazy into v_data_sprzedazy_faktury from bd_faktura
where nr_faktury=v_nr_faktury;

--teraz robie zapytanie, ktore bd odpowiadalo za ilosc generowanych
rekordow

select count(*) into v_ile_samochodow_z_roku
from bd_samochod
where rok_prod=(extract(year from v_data_sprzedazy_faktury));

select round(dbms_random.value(1,v_ile_samochodow_z_roku)) into
v_losowa_ilosc_samochodu
from dual;

v_counter:=0;

while v_counter!=v_losowa_ilosc_samochodu loop

v_nr_samochodu:=fn_daj_samochod_losowo(v_data_sprzedazy_faktury);

select count(*) into v_czy_jest_juz
from bd_pozycja_faktury
where nr_faktury=v_nr_faktury and nr_samochodu=v_nr_samochodu;

if v_czy_jest_juz=0 then

v_ilosc:=fn_daj_losowa_ilosc_samochodu(v_nr_samochodu);

if v_ilosc>0 then

insert into bd_pozycja_faktury(nr_faktury,ilosc,nr_samochodu)
values(
v_nr_faktury,v_ilosc,v_nr_samochodu
);

```

```

v_counter:=v_counter+1;
end if;
end if;

end loop;

--zmieniam status faktury na Zakonczona

update bd_faktura
set status='ZAMKNIETA'
where nr_faktury=v_nr_faktury;

elsif v_zakonczenie_petli_data+13>'19/12/31' then
v_stop:=true;
end if;

end loop;

end;
```

Opis działania:

- 1) W pętli, która będzie się wykonywała, dopóki **data_sprzedaży** ostatniej faktury będzie mniejsza lub równa 13 dni od daty '19/12/31'. Liczba 13 została przeze mnie ustalona, gdyż założyłem, że kolejne faktury będą generowane z datą sprzedaży od 9 do 13 dni późniejszą od daty sprzedaży ostatniej faktury. Jeśli podany wyżej warunek nie będzie spełniony pętla zakończy się.
- 2) Losuję klienta, który jest potrzebny do dodania rekordu do tabeli **bd_faktura**
- 3) Dodaję nowy rekord w tabeli **bd_faktura**. Nie muszę dodawać nr_faktury, gdyż jest on tworzony za pomocą **triggera**, który uruchamia się w momencie próby dodania nowego rekordu do tabeli i jest równy kolejnym wartościom sekwencji **seq_faktura**.
- 4) Losuję liczbę samochodów z tego samego roku co faktura, na której działam
- 5) W pętli od 1 do wylosowanej liczby samochodów:

- a) Losuję samochód dopóki nie będzie takiego samego na danej fakturze.
 - b) Losuję ilość dla podanego numeru samochodu dopóki nie będzie równa 0.
 - c) Dodaję do tabeli **bd_pozycja_faktury** rekord o wartościach wyżej wylosowanych, resztę załatwi **trigger**, którego działanie opiszę później.
- 6) Po zakończeniu pętli, zmieniam status faktury na 'ZAMKNIĘTA'

- **Trigger** użyte w wyżej wymienionej procedurze

1) **Trigger** wywoływany w momencie dodawania rekordu do tabeli **bd_faktura** – **tr_ins_upd_faktura**

```
create or replace TRIGGER TR_INS_FAKTURA
BEFORE INSERT OR UPDATE ON BD_FAKTURA
FOR EACH ROW
declare
v_tmp number;
v_ile_faktur number;
v_wylosowana number;
v_date date;

v_sam_upd number(4);
v_s number(4);
v_ile_pozycji number(4);
v_ilosc_samochodu number(3);
v_cena number(8,2);
v_data_zakupu date;
v_ilosc_tmp number(3);
v_razem number(10,2);
BEGIN
```

```
if inserting then
```

```
v_tmp:=SEQ_FAKTURA.NEXTVAL;
```

```
select count(*) into v_ile_faktur  
from bd_faktura;
```

```
--zaczynam faktury od 1 stycznia
```

```
if v_ile_faktur=0 then
```

```
v_date:='18/01/01';
```

```
--jesli juz jakas jest to biore ostatnia date i losuje o ile dni bd  
przesunieta nowa faktura
```

```
else
```

```
select max(data_sprzedazy) into v_date  
from bd_faktura;
```

```
--jesli jest mniejsza od '19/12/31' o 13 dni to jeszcze jest dobrze
```

```
if v_date+13<='19/12/31' then
```

```
select dbms_random.value(9,13) into v_wylosowana  
from dual;
```

```
v_date:=v_date+v_wylosowana;
```

```
else v_date:= '19/12/31';
```

```
end if;
```

```
end if;
```

```
:NEW.nr_faktury:=v_tmp;
```

```
:NEW.data_sprzedazy:=v_date;
```

```
:NEW.data_platnosci:=:NEW.data_sprzedazy+7;
```

```
elsif updating('status') then
```

```
select count(*) into v_ile_pozycji  
from bd_pozycja_faktury  
where nr_faktury=:OLD.nr_faktury;
```

```

for counter in 1..v_ile_pozycji loop
select * into v_s,v_sam_upd from (
select rownum as "LP",nr_samochodu from bd_pozycja_faktury
where nr_faktury=:OLD.nr_faktury)
where "LP "=counter;

select ilosc into v_ilosc_samochodu
from bd_samochod
where nr_samochodu=v_sam_upd;

if v_ilosc_samochodu<2 then

select cena into v_cena
from bd_samochod
where nr_samochodu=v_sam_upd;

if v_cena<100000 then
v_ilosc_tmp:=4;
elsif v_cena>=100000 and v_cena<250000 then
v_ilosc_tmp:=3;
elsif v_cena>=250000 then v_ilosc_tmp:=1;
end if;

update bd_samochod
set ilosc=ilosc+v_ilosc_tmp
where nr_samochodu=v_sam_upd;

if v_cena>=20000 and v_cena<30000 then
v_cena:=v_cena-2000;
elsif v_cena>=30000 and v_cena<50000 then
v_cena:=v_cena-5000;
elsif v_cena>=50000 and v_cena<150000 then
v_cena:=v_cena-10000;
elsif v_cena>=150000 then
v_cena:=v_cena-20000;
end if;

v_razem:=v_cena*v_ilosc_tmp;

```



```

insert into bd_wydatki (cena,ilosc,data_zakupu,nr_samochodu,razem)
values (
v_cena,v_ilosc_tmp,:OLD.data_sprzedazy,v_sam_upd,v_razem
);

end if;

end loop;
end if;

END;

```

Opis działania:

1. Przypisuję kluczowi głównemu wartość, którą osiągnie sekwencja w kolejnym wywołaniu
2. Chcąc dodać datę sprzedaży do faktury, sprawdzam ile jest faktur w ewidencji. W przypadku, gdy:
 - a) 0; Wtedy dodaję fakturę z datą umowną '18/01/01', gdyż w mojej ewidencji są samochody o roku produkcji od 2018 do 2019
 - b) >0; Wtedy pobieram maksymalną datę – czyli datę ostatniej faktury – z ewidencji faktur i losuję liczbę z przedziału (9,13), o którą będzie 'starsza' dodawana faktura.
3. Data płatności to 7 dni od daty sprzedaży
4. W przypadku zmieniania statusu faktury na 'ZAMKNIETA' należy sprawdzić, czy nie trzeba domówić samochodów, które znalazły się na danej fakturze. Ustawiłem sobie, że domawiam samochody, jeśli ilość danego samochodu spadnie poniżej 2. Oczywiście takie zamówienie powoduje dodanie nowego rekordu do tabeli **bd_wydatki**. Dodałem warunki na cenę oraz ilość dodawanego samochodu. Im droższy

samochód, tym ilość zamówionego samochodu jest mniejsza.

Oczywiście cena samochodu zamawianego, czyli wydatek, jest mniejsza od ceny za którą sprzedajemy samochód.

2) **Trigger** wywoływany w momencie dodawania rekordu do tabeli

bd_pozycja_faktury – tr_ins_upd_del_pozycja_faktury

```
create or replace TRIGGER TR_INS_UPD_DEL_POZYCJA_FAKTURY
BEFORE DELETE OR INSERT OR UPDATE ON BD_POZYCJA_FAKTURY
FOR EACH ROW

declare

v_czy_mozna_dodac number;
v_czy_jest_taki_sam number;
v_ile_wszystkich number;
v_znajdz_cene_pojazdu bd_samochod.cena%type;
v_znajdz_numer_takiego_samego number;
v_tmp_ins number;

v_update_znajdz_zmieniana_pozycje number;
v_update_tmp number;--zmienna ktora powie, czy mozna zupdatowac dany
rekord

v_tmp number;
v_dokad number;
v_status varchar2(20);

BEGIN

if inserting then

select ilosc into v_czy_mozna_dodac--sprawdzam, czy mozna wogole dodac
taka ilosc jakos pozycje faktury

from bd_samochod

where nr_samochodu=:NEW.nr_samochodu;

if v_czy_mozna_dodac>=:NEW.ilosc then

select count(*) into v_czy_jest_taki_sam--sprawdzam czy bede dodawal
nowa pozycje czy jednak moze tylko zmienic ilosc na pozycji
```

```

from BD_POZYCJA_FAKTURY

where NR_samochodu=:NEW.nr_samochodu and NR_FAKTURY=:NEW.nr_faktury;

if v_czy_jest_taki_sam=0 then

select count(*) into v_tmp_ins from bd_pozycja_faktury--sprawdzam ile
jest pozycji danej faktury

where nr_faktury=:NEW.nr_faktury;

if v_tmp_ins=0 then v_ile_wszystkich:=0;
else
select max(nr_pozycji) into v_ile_wszystkich--ustalam max, na tej
podstawie wylicze nowa pozycje dodawanej pozycji
from bd_pozycja_faktury
where NR_FAKTURY=:NEW.nr_faktury;
end if;

v_ile_wszystkich:=v_ile_wszystkich+1;

select cena into v_znajdz_cene_pojazdu from bd_samochod
where nr_samochodu=:NEW.nr_samochodu;

:NEW.nr_pozycji:=v_ile_wszystkich;
:NEW.cena:=v_znajdz_cene_pojazdu;
:NEW.vat:=:NEW."%vat"*:NEW.ilosc*v_znajdz_cene_pojazdu/100;
:NEW.brutto:=v_znajdz_cene_pojazdu*:NEW.ilosc+:NEW.vat;

elsif v_czy_jest_taki_sam>0 then

raise_application_error(-20001,'Nie mozna dodac do faktury pozycji o
takim samym samochodzie!

Mozesz zupdatowac ta pozycje');

end if;

update bd_samochod
set ilosc=ilosc-:NEW.ilosc
where nr_samochodu=:NEW.nr_samochodu;

elsif v_czy_mozna_dodac<:NEW.ilosc then

```

```

raise_application_error(-20001,'Nie mozna dodac do faktury pozycji o
ilosci wiekszej niz jest dostepna!');

end if;

elsif updating ('ilosc') then--bedzie mozna nadpisac ilosc

select ilosc into v_czy_mozna_dodac--sprawdzam, czy mozna wogole dodac
taka ilosc jakos pozycje faktury

from bd_samochod

where nr_samochodu=:NEW.nr_samochodu;

v_update_tmp:=(:OLD.ilosc+v_czy_mozna_dodac)-:NEW.ilosc;

--stad

if v_update_tmp>=0 then

update bd_samochod--zmieniam ilosc starego samochodu

set ilosc=ilosc+(:OLD.ilosc-:NEW.ilosc)

where nr_samochodu=:OLD.nr_samochodu;

elsif v_update_tmp<0 then

raise_application_error(-20001,'Nie mozna dodac do faktury pozycji o
ilosci wiekszej niz jest dostepna!');

end if;

--dotad

elsif deleting then

NULL;

end if;

END;

```

Opis działania:

 W przypadku dodawania nowego rekordu(insert)


1. Sprawdzam, czy dodawana ilość nie jest większa od ilości danego towaru w ewidencji. Gdy jest niewystarczająca ilość towaru wyświetlany jest błąd. Później dałem również warunek sprawdzający, czy danego samochodu nie ma już w danej fakturze. Jeśli jest, również zostanie wyświetlony błąd.
2. Sprawdzam ile jest pozycji w danej fakturze. Dzięki temu ustalę wartość

klucza głównego **nr_pozycji**. W przypadku gdy:

- a) 0; Wtedy wartość klucza głównego **nr_pozycji** będzie równa 1
- b) >0; Wtedy pobieram maksymalny numer pozycji. Nowa wartość klucza głównego **nr_pozycji** będzie większa od wartości pobranej o 1

3. Określam jakie wartości będą miały kolumny: „vat” oraz „brutto” zgodnie z przyjętymi normami. Kolumna „%vat” jest ustawiona domyślnie na 23.

4. Zmniejszam ilość dostępnego towaru, na którym teraz działam o ilość, którą chcę dodać pozycji danej faktury.

 W przypadku zmieniania ilości danego samochodu na pozycji danej faktury(update)

1. Sprawdzam czy ilość danego samochodu którą chcemy wstawić do pozycji danej faktury jest w ogóle dostępna.

2. Jeśli jest możliwa zmiana w pozycji faktury, to jest ona dokonywana, jeśli nie wyświetlany jest błąd.

Poza wyżej wymienionymi **triggerami**, które bezpośrednio biorą udział w procesie generowania danych stworzyłem również kilka innych, których zadanie sprowadzało się do nadania wartości klucza głównego danej tabeli na podstawie odpowiedniej dla tabeli sekwencji, po to, aby czynność ta wykonywała się automatycznie.

3) **Trigger** wywoływany w momencie dodawania rekordu do tabeli **bd_silniki**
– tr_ins_bd_silniki

```
create or replace TRIGGER TR_INS_BD_SILNIKI
BEFORE INSERT ON BD_SILNIKI
FOR EACH ROW
DECLARE
v_tmp number;
BEGIN
v_tmp:=seq_silniki.nextval;
```

```
:NEW.nr_silnika:=v_tmp;
```

```
END;
```

4) **Trigger** wywoływany w momencie dodawania rekordu do tabeli **bd_klient**

– tr_ins_klient

```
create or replace TRIGGER "TR_INS_KLIENT" BEFORE INSERT ON BD_KLIENT
```

```
FOR EACH ROW
```

```
declare v_tmp number;
```

```
v_nr number;
```

```
BEGIN
```

```
v_tmp:=SEQ_NR_KLIENTA.NEXTVAL;
```

```
:NEW.nr_klienta:=v_tmp;
```

```
END;
```

5) Trigger wywoływany w momencie dodawania rekordu do tabeli

bd_samochod – tr_ins_samochod

```
create or replace TRIGGER TR_INS_SAMOCOD
BEFORE INSERT ON BD_SAMOCOD
FOR EACH ROW
DECLARE
v_tmp number;
v_nr_silnika number;
BEGIN
v_tmp:=SEQ_SAMOCOD.NEXTVAL;

select * into v_nr_silnika from
(select nr_silnika
from bd_silniki
order by dbms_random.value)
where rownum=1;

:NEW.nr_samochodu:=v_tmp;
:NEW.nr_silnika:=v_nr_silnika;

END;
```

Jedyną zmianą w odróżnieniu od poprzedniego **triggera** jest fakt, iż dodatkowo jest losowany nr_silnika ze słownika, który jest kluczem obcym w tabeli **bd_samochod**.

6) Trigger wywoływany w momencie dodawania rekordu do tabeli

bd_wydatki – tr_ins_wydatki

```
create or replace TRIGGER TR_INS_WYDATKI
BEFORE INSERT ON BD_WYDATKI
for each row
declare v_tmp number;
BEGIN
v_tmp:=seq_wydatki.nextval;

:NEW.nr_wydatku:=v_tmp;

END;
```

- Funkcje użyte w automatycznym generowaniu danych

1) Funkcja **fn_daj_losowa_ilosc_samochodu**(v_numer_samochodu
bd_samochod.nr_samochodu%type)

```
function fn_daj_losowa_ilosc_samochodu(v_numer_samochodu  
bd_samochod.nr_samochodu%type) return bd_samochod.ilosc%type AS  
  
v_ilosc bd_samochod.ilosc%type;  
v_zwroc bd_samochod.ilosc%type;  
  
begin  
  
select ilosc into v_ilosc  
from bd_samochod  
where nr_samochodu=v_numer_samochodu;  
v_ilosc:=round((v_ilosc/2));  
  
if v_ilosc=0 then  
return 0;  
else  
select round(dbms_random.value(1,v_ilosc)) into v_zwroc  
from dual;  
return v_zwroc;  
end if;  
end;
```

Opis działania

1. Zapisuję w zmiennej ilość samochodu o numerze podanym do funkcji – **v_numer_samochodu**. Następnie ilość tą dzielę przez 2, po to aby nie robić za często zamówień u producenta. Jeśli ilość jest równa 0, to zwracam 0, jeśli nie to losuję liczbę z zakresu od 1 do ilości samochodów i ją zwracam na końcu funkcji.

2) Funkcja **fn_daj_numer_klienta**

```
function fn_daj_numer_klienta return bd_klient.nr_klienta%type AS  
  
v_tmp bd_klient.nr_klienta%type;  
  
begin  
  
select * into v_tmp from(
```



```

select nr_klienta
from bd_klient
order by dbms_random.value)
where rownum=1;

return v_tmp;
end;

```

Opis działania:

1. Losuję po prostu numer klienta z listy klientów w tabeli **bd_klient**.

Działanie to polega na pobraniu pierwszego numeru klienta po uprzednim posortowaniu wg losowej liczby.

3) Funkcja **fn_daj_samochod_losowo(v_date date)**

```

function fn_daj_samochod_losowo(v_date date) return
bd_samochod.nr_samochodu%type as
v_numer bd_samochod.nr_samochodu%type;
begin

select * into v_numer from(
select nr_samochodu
from bd_samochod
where bd_samochod.ROK_PROD=extract(year from v_date)
order by dbms_random.value)
where rownum=1;

return v_numer;
end;

```

Opis działania:

1. Losuję numer samochodu spośród wszystkich w tabeli **bd_samochod**, który jest z tego samego roku, co zmienna **v_date**(jest to data_sprzedazy faktury). Szczegółowy opis działania jest taki sam, jak w funkcji wyżej.

4. Skrypty instalujące i deinstalujące zrealizowany projekt

1) Skrypty instalujące projekt

```
CREATE SEQUENCE seq_faktura START WITH 1 INCREMENT BY 1 NOCACHE;
```

```
CREATE SEQUENCE seq_klient START WITH 1 INCREMENT BY 1 NOCACHE;
```

```
CREATE SEQUENCE seq_samochod START WITH 1 INCREMENT BY 1 NOCACHE;
```

```
CREATE SEQUENCE seq_silniki START WITH 1 INCREMENT BY 1 NOCACHE;
```

```
CREATE SEQUENCE seq_wojewodztwa START WITH 1 INCREMENT BY 1 NOCACHE;
```

```
CREATE SEQUENCE seq_wydatki START WITH 1 INCREMENT BY 1 MINVALUE 1  
NOCACHE;
```

```
CREATE TABLE bd_faktura (  
    nr_faktury          NUMBER(4) NOT NULL,  
    nr_klienta          NUMBER(4) NOT NULL,  
    data_sprzedazy      DATE DEFAULT SYSDATE,  
    data_platnosci      DATE DEFAULT sysdate+7,  
    status              VARCHAR2(20) DEFAULT 'OTWARTA',  
    wlasciciel_imie     VARCHAR2(20) DEFAULT 'Radoslaw',  
    wlasciciel_nazwisko VARCHAR2(40) DEFAULT 'Zurawicz',  
    wlasciciel_adres    VARCHAR2(20) DEFAULT 'Koszykarska 55/50',  
    wlasciciel_miasto   VARCHAR2(20) DEFAULT 'Lublin',  
    wlasciciel_kod_pocztowy VARCHAR2(6) DEFAULT '20-015',  
    nr_wojewodztwa      NUMBER(2) DEFAULT 3 NOT NULL  
);
```

```
ALTER TABLE bd_faktura ADD CONSTRAINT bd_faktura_pk PRIMARY KEY (  
nr_faktury );
```

```
CREATE TABLE bd_klient (  
    nr_klienta          NUMBER(4) NOT NULL,  
    klient_imie         VARCHAR2(20) NOT NULL,  
    klient_nazwisko     VARCHAR2(40) NOT NULL,  
    klient_adres        VARCHAR2(40) NOT NULL,  
    klient_miasto       VARCHAR2(20) NOT NULL,
```

```
        kod_pocztowy      VARCHAR2(6) NOT NULL,

        nr_wojewodztwa    NUMBER(2) NOT NULL

    );
```

```
ALTER TABLE bd_klient ADD CONSTRAINT bd_klient_pk PRIMARY KEY (
nr_klienta );
```

```
CREATE TABLE bd_samochod (

    nr_samochodu          NUMBER(4) NOT NULL,

    marka                 VARCHAR2(20) NOT NULL,

    model                 VARCHAR2(20) NOT NULL,

    skrzynia_biegow       VARCHAR2(20) DEFAULT 'MANUALNA',

    rok_prod              VARCHAR2(4) DEFAULT EXTRACT(YEAR FROM SYSDATE),

    kolor                 VARCHAR2(20) DEFAULT 'CZARNY',

    cena                  NUMBER(8,2) NOT NULL,

    nr_silnika            NUMBER(4) NOT NULL,

    ilosc                 NUMBER(3) NOT NULL

);
```

```
ALTER TABLE bd_samochod ADD CONSTRAINT bd_samochod_pk PRIMARY KEY (
nr_samochodu );
```

```
CREATE TABLE bd_silniki (

    nr_silnika           NUMBER(4) NOT NULL,

    pojemnosc            NUMBER(4) NOT NULL,

    moc                  NUMBER(3) NOT NULL,

    paliwo               VARCHAR2(20) NOT NULL

);
```

```
ALTER TABLE bd_silniki ADD CONSTRAINT bd_silniki_pk PRIMARY KEY (
nr_silnika );
```

```
CREATE TABLE bd_wojewodztwa (

    nr_wojewodztwa       NUMBER(2) NOT NULL,

    nazwa_wojewodztwa    VARCHAR2(20) NOT NULL

);
```

```
ALTER TABLE bd_wojewodztwa ADD CONSTRAINT bd_wojewodztwa_pk PRIMARY KEY (
nr_wojewodztwa );
```

```
CREATE TABLE bd_wydatki (
```

```

        nr_wydatku      NUMBER(4) NOT NULL,
        cena            NUMBER(10,2) ,
        ilosc           NUMBER(4) DEFAULT 4,
        data_zakupu     DATE,
        razem           NUMBER(10,2) ,
        nr_samochodu    NUMBER(4) NOT NULL
    );

```

```

ALTER TABLE bd_wydatki ADD CONSTRAINT bd_wydatki_pk PRIMARY KEY (
nr_wydatku );

```

```

CREATE TABLE pozycja_faktury (
    nr_faktury          NUMBER(4) NOT NULL,
    nr_pozycji          NUMBER(4) NOT NULL,
    cena                NUMBER(8,2) ,
    ilosc               NUMBER(3) NOT NULL,
    nr_samochodu        NUMBER(4) NOT NULL,
    "%vat"              NUMBER(2) DEFAULT 23,
    vat                 NUMBER(10,2) ,
    brutto              NUMBER(10,2)
);

```

```

ALTER TABLE pozycja_faktury ADD CONSTRAINT pozycja_faktury_pk PRIMARY KEY
( nr_faktury,nr_pozycji );

```

```

ALTER TABLE pozycja_faktury
    ADD CONSTRAINT bd_faktura_fk FOREIGN KEY ( nr_faktury )
        REFERENCES bd_faktura ( nr_faktury )
        ON DELETE CASCADE;

```

```

ALTER TABLE bd_faktura
    ADD CONSTRAINT bd_klient_fk FOREIGN KEY ( nr_klienta )
        REFERENCES bd_klient ( nr_klienta );

```

```

ALTER TABLE pozycja_faktury
    ADD CONSTRAINT bd_samochod_fk FOREIGN KEY ( nr_samochodu )
        REFERENCES bd_samochod ( nr_samochodu )
        ON DELETE CASCADE;

```

```

ALTER TABLE bd_wydatki

```

```

ADD CONSTRAINT bd_samochod_fkv2 FOREIGN KEY ( nr_samochodu )
REFERENCES bd_samochod ( nr_samochodu )
ON DELETE CASCADE;

```

```

ALTER TABLE bd_samochod

```

```

ADD CONSTRAINT bd_silniki_fk FOREIGN KEY ( nr_silnika )
REFERENCES bd_silniki ( nr_silnika );

```

```

ALTER TABLE bd_klient

```

```

ADD CONSTRAINT bd_wojewodztwa_fk FOREIGN KEY ( nr_wojewodztwa )
REFERENCES bd_wojewodztwa ( nr_wojewodztwa );

```

```

ALTER TABLE bd_faktura

```

```

ADD CONSTRAINT bd_wojewodztwa_fkv2 FOREIGN KEY ( nr_wojewodztwa )
REFERENCES bd_wojewodztwa ( nr_wojewodztwa );

```

```

create or replace PACKAGE BODY GENEROWANIE_PROJEKT AS

```

```

function fn_daj_losowa_ilosc_samochodu(v_numer_samochodu in
bd_samochod.nr_samochodu%type) return bd_samochod.ilosc%type AS

```

```

v_ilosc bd_samochod.ilosc%type;

```

```

v_zwroc bd_samochod.ilosc%type;

```

```

begin

```

```

select ilosc into v_ilosc

```

```

from bd_samochod

```

```

where nr_samochodu=v_numer_samochodu;

```

```

v_ilosc:=round((v_ilosc/2));

```

```

if v_ilosc=0 then

```

```

return 0;

```

```

else

```

```

select round(dbms_random.value(1,v_ilosc)) into v_zwroc

```

```

from dual;

```

```

return v_zwroc;

```

```

end if;

```

```

end fn_daj_losowa_ilosc_samochodu;

```

```

function fn_daj_numer_klienta return bd_klient.nr_klienta%type AS

```

```

v_tmp bd_klient.nr_klienta%type;

```

```
begin
```

```
select * into v_tmp from(  
select nr_klienta  
from bd_klient  
order by dbms_random.value)  
where rownum=1;
```

```
return v_tmp;  
end fn_daj_numer_klienta;
```

```
function fn_daj_samochod_losowo(v_date in date) return  
bd_samochod.nr_samochodu%type as
```

```
v_numer bd_samochod.nr_samochodu%type;  
begin
```

```
select * into v_numer from(  
select nr_samochodu  
from bd_samochod  
where bd_samochod.ROK_PROD=extract(year from v_date)  
order by dbms_random.value)  
where rownum=1;
```

```
return v_numer;  
end fn_daj_samochod_losowo;
```

```
procedure generuj_dane AS
```

```
BEGIN
```

```
declare  
v_nr_klienta bd_klient.nr_klienta%type;  
v_nr_faktury bd_faktura.nr_faktury%type;  
v_data_sprzedazy_faktury date;  
v_nr_samochodu bd_samochod.nr_samochodu%type;  
v_ilosc bd_samochod.ilosc%type;  
v_ile_samochodow_z_roku bd_samochod.nr_samochodu%type;  
v_counter number;  
v_losowa_ilosc_samochodu bd_samochod.nr_samochodu%type;  
v_czy_jest_juz number;  
v_zakonczenie_petli_ile number;  
v_zakonczenie_petli_data date;
```

```

v_stop boolean;

begin

v_stop:=false;

while v_stop!=true loop

select count(*) into v_zakonczenie_petli_ile
from bd_faktura;

if v_zakonczenie_petli_ile=0 then
    v_zakonczenie_petli_data:='18/01/01';
else
    select max(data_sprzedazy) into v_zakonczenie_petli_data
    from bd_faktura;
end if;

if v_zakonczenie_petli_data+13<='19/12/31' then

v_nr_klienta:=fn_daj_numer_klienta();

insert into bd_faktura(nr_klienta)
values(
v_nr_klienta);

--pobieram nr faktury na ktorej dzialam
select nr_faktury into v_nr_faktury
from bd_faktura where status='OTWARTA';

select data_sprzedazy into v_data_sprzedazy_faktury from bd_faktura
where nr_faktury=v_nr_faktury;

--teraz robie zapytanie, ktore bd odpowiadalo za ilosc generowanych
rekordow

select count(*) into v_ile_samochodow_z_roku
from bd_samochod
where rok_prod=(extract(year from v_data_sprzedazy_faktury));

```

```

        select round(dbms_random.value(1,v_ile_samochodow_z_roku)) into
v_losowa_ilosc_samochodu

        from dual;

v_counter:=0;

while v_counter!=v_losowa_ilosc_samochodu loop

v_nr_samochodu:=fn_daj_samochod_losowo(v_data_sprzedazy_faktury);

select count(*) into v_czy_jest_juz
from bd_pozycja_faktury
where nr_faktury=v_nr_faktury and nr_samochodu=v_nr_samochodu;

if v_czy_jest_juz=0 then

v_ilosc:=fn_daj_losowa_ilosc_samochodu(v_nr_samochodu);

if v_ilosc>0 then

insert into bd_pozycja_faktury(nr_faktury,ilosc,nr_samochodu)
values(
v_nr_faktury,v_ilosc,v_nr_samochodu
);

v_counter:=v_counter+1;
end if;
end if;

end loop;

--zmieniam status faktury na Zakonczone

update bd_faktura
set status='ZAMKNIETA'
where nr_faktury=v_nr_faktury;

elsif v_zakonczenie_petli_data+13>'19/12/31' then
v_stop:=true;
end if;

```



```

end loop;

end;

END generuj_dane;

procedure dodaj_samochod_klient_wojewodztwa as
begin

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Opel','Astra','2018','CZARNY','56000','3');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Opel','Insignia','2019','CZERWONY','76000','5');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Opel','Insignia','2018','CZERWONY','72000','3');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Opel','Vectra','2018','ZIELONY','46000','5');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Opel','Vectra','2019','ZIELONY','49000','5');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Polonez','Caro','2019','BORDOWY','41000','5');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Polonez','Caro','2018','BIALY','36000','3');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Seat','Toledo','2019','BIALY','81000','5');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Seat','Toledo','2018','BIALY','76000','5');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Seat','Cordoba','2018','SZARY','52000','2');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Seat','Cordoba','2019','SZARY','57000','5');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Seat','Exeo','2018','GRANATOWY','89000','2');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Seat','Exeo','2019','ROZOWY','99000','5');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Volvo','S40','2018','ROZOWY','55000','3');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Volvo','S40','2019','CZARNY','60000','5');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Volvo','S60','2019','CZARNY','65000','5');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Volvo','S60','2018','NIEBIESKI','61000','3');

    Insert into BD_SAMOCOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Volvo','S80','2018','NIEBIESKI','84000','5');

```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Volvo','S80','2019','ZOLTY','89000','5');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Volvo','S90','2019','CZERWONY','79000','5');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Volvo','S90','2018','CZERWONY','69000','5');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Porsche','Cayenne','2018','ZIELONY','100000','4');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Porsche','Cayenne','2019','SREBRNY','140000','3');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Porsche','Panamera','2019','BIALY','250000','1');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Porsche','Panamera','2018','POMARANCZOWY','230000','4');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Porsche','911','2018','POMARANCZOWY','650000','1');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Porsche','911','2019','ZLOTY','750000','1');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Fiat','500','2019','CZARNY','21500','5');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Fiat','500','2018','CZARNY','20500','5');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Nissan','Micra','2018','CZARNY','25500','3');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Nissan','350Z','2018','CZARNY','125500','2');
```

```
Insert into BD_SAMUCHOD (MARKA,MODEL,ROK_PROD,KOLOR,CENA,ILOSC)
values ('Seat','Alhambra','2018','SIWY','65500','2');
```

```
Insert into BD_KLIENT
(KLIENT_IMIE,KLIENT_NAZWISKO,KLIENT_ADRES,KLIENT_MIASTO,KOD_POCZTOWY,NR_W
OJEWODZTWA) values ('Rafal','Piotrowski','Konwaliowa 5','Lublin','20-
023','3');
```

```
Insert into BD_KLIENT
(KLIENT_IMIE,KLIENT_NAZWISKO,KLIENT_ADRES,KLIENT_MIASTO,KOD_POCZTOWY,NR_W
OJEWODZTWA) values ('Krzysztof','Andersen','Waska 13','Rzeszow','35-
032','9');
```

```
Insert into BD_KLIENT
(KLIENT_IMIE,KLIENT_NAZWISKO,KLIENT_ADRES,KLIENT_MIASTO,KOD_POCZTOWY,NR_W
OJEWODZTWA) values ('Marek','Mostowiak','Pilsudskiego 135','Opole','45-
016','8');
```

```
Insert into BD_KLIENT
(KLIENT_IMIE,KLIENT_NAZWISKO,KLIENT_ADRES,KLIENT_MIASTO,KOD_POCZTOWY,NR_W
OJEWODZTWA) values ('Dariusz','Michalski','Piekna 21','Lodz','90-
019','5');
```

```
Insert into BD_KLIENT
(KLIENT_IMIE,KLIENT_NAZWISKO,KLIENT_ADRES,KLIENT_MIASTO,KOD_POCZTOWY,NR_W
OJEWODZTWA) values ('Barbara','Cichosz','Spokojna 1','Olsztyn','10-
015','14');
```

```
Insert into BD_KLIENT
(KLIENT_IMIE,KLIENT_NAZWISKO,KLIENT_ADRES,KLIENT_MIASTO,KOD_POCZTOWY,NR_W
```

```
OJEWODZTWA) values ('Teofila','Kowalska','Pospolita 39','Gdansk','80-022','11');
```

```
Insert into BD_KLIENT  
(KLIENT_IMIE,KLIENT_NAZWISKO,KLIENT_ADRES,KLIENT_MIASTO,KOD_POCZTOWY,NR_W  
OJEWODZTWA) values ('Stanislaw','Szczygiel','Browarna 100','Torun','85-825','2');
```

```
Insert into BD_KLIENT  
(KLIENT_IMIE,KLIENT_NAZWISKO,KLIENT_ADRES,KLIENT_MIASTO,KOD_POCZTOWY,NR_W  
OJEWODZTWA) values ('Dagmara','Pisklak','Cicha 45','Czestochowa','42-215','12');
```

```
Insert into BD_KLIENT  
(KLIENT_IMIE,KLIENT_NAZWISKO,KLIENT_ADRES,KLIENT_MIASTO,KOD_POCZTOWY,NR_W  
OJEWODZTWA) values ('Henryk','Gall','Nieznana 99','Katowice','40-014','12');
```

```
Insert into BD_KLIENT  
(KLIENT_IMIE,KLIENT_NAZWISKO,KLIENT_ADRES,KLIENT_MIASTO,KOD_POCZTOWY,NR_W  
OJEWODZTWA) values ('Lukasz','Stanislaw','Poczatkowa 33','Krasnik','23-200','3');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values  
('dolnoslaskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values ('kujawsko-  
pomorskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values ('lubelskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values ('lubuskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values ('lodzkie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values  
('malopolskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values  
('mazowieckie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values ('opolskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values  
('podkarpackie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values ('podlaskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values ('pomorskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values ('slaskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values  
('swietokrzyskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values ('warminsko-  
mazurskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values  
('wielkopolskie');
```

```
Insert into BD_WOJEWODZTWA (NAZWA_WOJEWODZTWA) values  
('zachodniopomorskie');
```

```
end dodaj_samochod_klient_wojewodztwa;
```

```
END GENEROWANIE_PROJEKT;
```

```

create or replace TRIGGER TR_INS_BD_SILNIKI
BEFORE INSERT ON BD_SILNIKI
FOR EACH ROW
DECLARE
v_tmp number;
BEGIN
v_tmp:=seq_silniki.nextval;
:NEW.nr_silnika:=v_tmp;
END;

```

```

create or replace TRIGGER TR_INS_FAKTURA
BEFORE INSERT OR UPDATE ON BD_FAKTURA
FOR EACH ROW
declare
v_tmp number;
v_ile_faktur number;
v_wylosowana number;
v_date date;

```

```

v_sam_upd number(4);
v_s number(4);
v_ile_pozycji number(4);
v_ilosc_samochodu number(3);
v_cena number(8,2);
v_data_zakupu date;
v_ilosc_tmp number(3);
v_razem number(10,2);
BEGIN

```

```

if inserting then

```

```

v_tmp:=SEQ_FAKTURA.NEXTVAL;

```

```

select count(*) into v_ile_faktur
from bd_faktura;

```

```

--zaczynam faktury od 1 stycznia
if v_ile_faktur=0 then

v_date:='18/01/01';

--jesli juz jakas jest to biore ostatnia date i losuje o ile dni bd
przesunieta nowa faktura
else
select max(data_sprzedazy) into v_date
from bd_faktura;
--jesli jest mniejsza od '19/12/31' o 9 dni to jeszcze git
if v_date+13<='19/12/31' then

select dbms_random.value(9,13) into v_wylosowana
from dual;

v_date:=v_date+v_wylosowana;

else v_date:= '19/12/31';
end if;

end if;

:NEW.nr_faktury:=v_tmp;
:NEW.data_sprzedazy:=v_date;
:NEW.data_platnosci:=:NEW.data_sprzedazy+7;

elsif updating('status') then

select count(*) into v_ile_pozycji
from bd_pozycja_faktury
where nr_faktury=:OLD.nr_faktury;

for counter in 1..v_ile_pozycji loop
select * into v_s,v_sam_upd from (
select rownum as "LP",nr_samochodu from bd_pozycja_faktury
where nr_faktury=:OLD.nr_faktury)
where "LP"=counter;

```

```

select ilosc into v_ilosc_samochodu
from bd_samochod
where nr_samochodu=v_sam_upd;

if v_ilosc_samochodu<2 then

select cena into v_cena
from bd_samochod
where nr_samochodu=v_sam_upd;

if v_cena<100000 then
v_ilosc_tmp:=4;
elsif v_cena>=100000 and v_cena<250000 then
v_ilosc_tmp:=3;
elsif v_cena>=250000 then v_ilosc_tmp:=1;
end if;

update bd_samochod
set ilosc=ilosc+v_ilosc_tmp
where nr_samochodu=v_sam_upd;

if v_cena>=20000 and v_cena<30000 then
v_cena:=v_cena-2000;
elsif v_cena>=30000 and v_cena<50000 then
v_cena:=v_cena-5000;
    elsif v_cena>=50000 and v_cena<150000 then
v_cena:=v_cena-10000;
elsif v_cena>=150000 then
v_cena:=v_cena-20000;
end if;

v_razem:=v_cena*v_ilosc_tmp;

insert into bd_wydatki(cena,ilosc,data_zakupu,nr_samochodu,razem)
values(

v_cena,v_ilosc_tmp,:OLD.data_sprzedazy,v_sam_upd,v_razem
);

```

```
end if;
```

```
end loop;
```

```
end if;
```

```
END;
```

```
create or replace TRIGGER "TR_INS_KLIENT" BEFORE INSERT ON BD_KLIENT  
FOR EACH ROW
```

```
declare v_tmp number;
```

```
v_nr number;
```

```
BEGIN
```

```
v_tmp:=SEQ_NR_KLIENTA.NEXTVAL;
```

```
:NEW.nr_klienta:=v_tmp;
```

```
END;
```

```
create or replace TRIGGER TR_INS_SAMOCOD
```

```
BEFORE INSERT ON BD_SAMOCOD
```

```
FOR EACH ROW
```

```
DECLARE
```

```
v_tmp number;
```

```
v_nr_silnika number;
```

```
v_data date;
```

```
v_czy_jest_faktura number;
```

```
BEGIN
```

```
v_tmp:=SEQ_SAMOCOD.NEXTVAL;
```

```
select * into v_nr_silnika from
```

```
(select nr_silnika
```

```
from bd_silniki
```

```
order by dbms_random.value)
```

```
where rownum=1;
```

```
:NEW.nr_samochodu:=v_tmp;
```

```
:NEW.nr_silnika:=v_nr_silnika;
```

```
END;
```

```
create or replace TRIGGER TR_INS_UPD_DEL_POZYCJA_FAKTURY
```

```

BEFORE DELETE OR INSERT OR UPDATE ON BD_POZYCJA_FAKTURY
FOR EACH ROW
declare
v_czy_mozna_dodac number;
v_czy_jest_taki_sam number;
v_ile_wszystkich number;
v_znajdz_cene_pojazdu bd_samochod.cena%type;
v_znajdz_numer_takiego_samego number;
v_tmp_ins number;

v_update_znajdz_zmieniana_pozycje number;
v_update_tmp number;--zmienna ktora powie, czy mozna zupdatowac dany
rekord

v_tmp number;
v_dokad number;
v_status varchar2(20);

BEGIN

    if inserting then

        select ilosc into v_czy_mozna_dodac--sprawdzam, czy mozna wogole dodac
taka ilosc jakos pozycje faktury
        from bd_samochod
        where nr_samochodu=:NEW.nr_samochodu;

        if v_czy_mozna_dodac>=:NEW.ilosc then

            select count(*) into v_czy_jest_taki_sam--sprawdzam czy bede dodawal
nowa pozycje czy jednak moze tylko zmienie ilosc na pozycji
            from BD_POZYCJA_FAKTURY
            where NR_samochodu=:NEW.nr_samochodu and NR_FAKTURY=:NEW.nr_faktury;

            if v_czy_jest_taki_sam=0 then

                select count(*) into v_tmp_ins from bd_pozycja_faktury--sprawdzam ile
jest pozycji danej faktury
                where nr_faktury=:NEW.nr_faktury;

                if v_tmp_ins=0 then v_ile_wszystkich:=0;
                else

```



```

        select max(nr_pozycji) into v_ile_wszystkich--ustalam max, na tej
podstawie wylicze nowa pozycje dodawanej pozycji

        from bd_pozycja_faktury

        where NR_FAKTURY=:NEW.nr_faktury;

        end if;

        v_ile_wszystkich:=v_ile_wszystkich+1;

        select cena into v_znajdz_cene_pojazdu from bd_samochod

        where nr_samochodu=:NEW.nr_samochodu;

        :NEW.nr_pozycji:=v_ile_wszystkich;

        :NEW.cena:=v_znajdz_cene_pojazdu;

        :NEW.vat:=:NEW."%vat"*:NEW.ilosc*v_znajdz_cene_pojazdu/100;

        :NEW.brutto:=v_znajdz_cene_pojazdu*:NEW.ilosc+:NEW.vat;

        elsif v_czy_jest_taki_sam>0 then

            raise_application_error(-20001,'Nie mozna dodac do faktury pozycji o
takim samym samochodzie!

            Mozesz zupdatowac ta pozycje');

        end if;

        update bd_samochod

        set ilosc=ilosc-:NEW.ilosc

        where nr_samochodu=:NEW.nr_samochodu;

        elsif v_czy_mozna_dodac<:NEW.ilosc then

            raise_application_error(-20001,'Nie mozna dodac do faktury pozycji o
ilosci wiekszej niz jest dostepna!');

        end if;

        elsif updating ('ilosc') then--bedzie mozna nadpisac ilosc

            select ilosc into v_czy_mozna_dodac--sprawdzam, czy mozna wogole dodac
taka ilosc jakos pozycje faktury

            from bd_samochod

            where nr_samochodu=:NEW.nr_samochodu;

```

```

v_update_tmp:=(:OLD.ilosc+v_czy_mozna_dodac)-:NEW.ilosc;

--stad

if v_update_tmp>=0 then

    update bd_samochod--zmieniam ilosc starego samochodu

    set ilosc=ilosc+(:OLD.ilosc-:NEW.ilosc)

    where nr_samochodu=:OLD.nr_samochodu;--tu jest obojetne czy OLD czy NEW
bo to jest ten sam nr samochodu

    elsif v_update_tmp<0 then

        raise_application_error(-20001,'Nie mozna dodac do faktury pozycji o
ilosci wiekszej niz jest dostepna!');

    end if;

--dotad

elsif deleting then

    NULL;

end if;

END;

create or replace TRIGGER TR_INS_WYDATKI
BEFORE INSERT ON BD_WYDATKI
for each row
declare v_tmp number;
BEGIN

    v_tmp:=seq_wydatki.nextval;

    :NEW.nr_wydatku:=v_tmp;

END;

create or replace view klient as

select
k.klient_imie,k.klient_nazwisko,k.klient_adres,k.klient_miasto,k.kod_pocz
towy,w.nazwa_wojewodztwa,sum(brutto) as "Suma wydana"

from bd_klient k,bd_pozycja_faktury p,bd_faktura f,bd_wojewodztwa w

where p.nr_faktury=f.nr_faktury and f.NR_KLIENTA=k.nr_klienta and
w.NR_WOJEWODZTWA=k.NR_WOJEWODZTWA

group by
k.klient_imie,k.klient_nazwisko,k.klient_adres,k.klient_miasto,k.kod_pocz
towy,w.nazwa_wojewodztwa

order by sum(brutto);

```

```

create or replace view samochod_brutto as

select s.nr_samochodu as "n",s.marka as "m",s.model as
"mo",s.skrzynia_biegow as "b",s.rok_prod as "r",s.kolor as "k",s.cena as
"c",sil.pojemnosc as "p",sil.moc as "mc",sum(brutto) as "s"

from bd_pozycja_faktury p,bd_samochod s,bd_silniki sil

where p.nr_samochodu=s.nr_samochodu and s.nr_silnika=sil.nr_silnika

group by
s.nr_samochodu,s.marka,s.model,s.skrzynia_biegow,s.rok_prod,s.kolor,s.cen
a,sil.pojemnosc,sil.moc,s.ilosc

order by s.rok_prod,s.nr_samochodu;

```

```

create or replace view samochod_razem as

select s.nr_samochodu as "n",s.marka as "m",s.model as
"mo",s.skrzynia_biegow as "b",s.rok_prod as "r",s.kolor as "k",s.cena as
"c",sil.pojemnosc as "p",sil.moc as "mc",sum(razem) as "s"

from bd_wydatki w,bd_samochod s,bd_silniki sil

where w.nr_samochodu=s.nr_samochodu and s.nr_silnika=sil.nr_silnika

group by
s.nr_samochodu,s.marka,s.model,s.skrzynia_biegow,s.rok_prod,s.kolor,s.cen
a,sil.pojemnosc,sil.moc,s.ilosc

order by s.rok_prod,s.nr_samochodu;

```

```

create or replace view samochod_zysk as

select b."n" as "n",b."m" as "m",b."mo" as "mo",b."b" as "b",b."r" as
"r",b."k" as "k",b."c" as "c",b."p" as "p",b."mc" as "mc",b."s"-r."s" as
"s"

from samochod_razem r,SAMOCOD_BRUTTO b

where b."n"=r."n"

order by b."r",b."n";

```

```

create or replace view brutto as

select sum("Wartosc") as "Brutto",extract(year from "Data") as
"Rok", (case

to_char(extract(month from "Data"))

when '1' then 'styczen'

when '2' then 'luty'

when '3' then 'marzec'

when '4' then 'kwiecien'

when '5' then 'maj'

when '6' then 'czerwiec'

when '7' then 'lipiec'

when '8' then 'sierpien'

when '9' then 'wrzesien'

```

```

when '10' then 'pazdziernik'

when '11' then 'listopad'

when '12' then 'grudzien'

end) as "Miesiac"

from

(select sum(brutto) as "Wartosc",f.DATA_SPRZEDAZY as
>Data",poz.nr_faktury

from bd_pozycja_faktury poz,bd_faktura f

where f.nr_faktury=poz.nr_faktury

group by poz.nr_faktury,f.data_sprzedazy)

group by extract(month from "Data"),extract(year from "Data")

order by extract(year from "Data"),extract(month from "Data");

create or replace view razem as

select sum("Wartosc") as "Razem",extract(year from "Data") as "Rok", (case
to_char(extract(month from "Data")))

when '1' then 'styczen'

when '2' then 'luty'

when '3' then 'marzec'

when '4' then 'kwiecien'

when '5' then 'maj'

when '6' then 'czerwiec'

when '7' then 'lipiec'

when '8' then 'sierpien'

when '9' then 'wrzesien'

when '10' then 'pazdziernik'

when '11' then 'listopad'

when '12' then 'grudzien'

end) as "Miesiac"

from

(select sum(razem) as "Wartosc",data_zakupu as "Data"

from bd_wydatki

group by data_zakupu)

group by extract(month from "Data"),extract(year from "Data")

order by extract(year from "Data"),extract(month from "Data");

create or replace view zysk as

select "Brutto"-"Razem" as "Zysk",r."Rok" as "Rok",r."Miesiac" as
"Miesiac"

from razem r,brutto b

```

```
where r."Miesiac" = b."Miesiac" and r."Rok" = b."Rok";
```

Dokładne opisy podanego kodu zawarłem w punkcie poprzednim przy okazji opisu generatora danych oraz tego z czego on korzysta.

2) Skrypty deinstalujące projekt

```
DROP TABLE bd_faktura CASCADE CONSTRAINTS;
```

```
DROP TABLE bd_klient CASCADE CONSTRAINTS;
```

```
DROP TABLE bd_pozycja_faktury CASCADE CONSTRAINTS;
```

```
DROP TABLE bd_samochod CASCADE CONSTRAINTS;
```

```
DROP TABLE bd_silniki CASCADE CONSTRAINTS;
```

```
DROP TABLE bd_wojewodztwa CASCADE CONSTRAINTS;
```

```
DROP TABLE bd_wydatki CASCADE CONSTRAINTS;
```

```
DROP SEQUENCE SEQ_FAKTURA;
```

```
DROP SEQUENCE SEQ_NR_KLIENTA;
```

```
DROP SEQUENCE SEQ_SAMOCOD;
```

```
DROP SEQUENCE SEQ_SILNIKI;
```

```
DROP SEQUENCE SEQ_WYDATKI;
```

```
DROP TRIGGER TR_INS_BD_SILNIKI;
```

```
DROP TRIGGER TR_INS_FAKTURA;
```

```
DROP TRIGGER TR_INS_KLIENT;
```

```
DROP TRIGGER TR_INS_SAMOCOD;
```

```
DROP TRIGGER TR_INS_UPD_DEL_POZYCJA_FAKTURY;
```

```
DROP TRIGGER TR_INS_WYDATKI;
```

```
drop package GENEROWANIE_PROJEKT;
```

```
drop view razem;
```

```
drop view brutto;
```

```
drop view zysk;
```

```
drop view samochod_brutto;
```

```
drop view samochod_razem;  
drop view samochod_zysk;  
drop view klient;
```

5. Instrukcja instalacji projektu i sprawdzenie jej poprawności

- 1) Uruchomić skrypt instalacyjny projekt.
- 2) Projekt jest gotowy, za pomocą generatora danych, który znajduje się w
Wyżej wymienionej paczce można wygenerować pozycje w tabeli
bd_faktura, **bd_wydatki** oraz **bd_pozycje_faktury**. Liczba danych jest
losowa i uzależniona od losowanych dni, o które mają być przesuwane
nowe faktury w stosunku do ostatniej, o czym pisałem przy procedurze
generuj_dane. Początkowa data tworzenia faktur to '18/01/01', a końcowa
'19/12/31'.

6. Wyniki działania perspektyw analitycznych w postaci raportów pdf

- 1)

Raport zysków w poszczególnych miesiącach w latach 2018-2019

Miesiąc	Rok	Zysk
styczen	2018	1 240 100,00
luty	2018	1 833 510,00
marzec	2018	80 010,00
kwiecień	2018	1 487 805,00
maj	2018	188 860,00
czerwiec	2018	299 130,00
lipiec	2018	1 112 645,00
sierpień	2018	2 031 600,00
wrzesień	2018	1 896 130,00
październik	2018	871 490,00
listopad	2018	775 550,00
grudzien	2018	1 624 390,00
styczen	2019	599 615,00
luty	2019	1 394 580,00
marzec	2019	1 067 075,00
kwiecień	2019	682 315,00
maj	2019	1 623 495,00
czerwiec	2019	1 443 385,00
lipiec	2019	2 139 465,00

Miesiac	Rok	Zysk
sierpień	2019	1 023 925,00
wrzesień	2019	793 440,00
październik	2019	586 100,00
listopad	2019	987 640,00
grudzień	2019	1 525 650,00

Strona 2 z 2

2)

Raport wydanych sum przez poszczególnych klientów

Imię	Nazwisko	Adres	Miasto	Kod pocztowy	Nazwa województwa	Suma wydana
Lukasz	Stanisław	Początkowa 33	Krasnik	23-200	lubelskie	6 138 315,00
Barbara	Cichosz	Spokojna 1	Olsztyn	10-015	warmińsko-mazurskie	6 551 595,00
Dariusz	Michalski	Piękna 21	Łódź	90-019	łódzkie	6 555 285,00
Teofila	Kowalska	Pospolita 39	Gdańsk	80-022	pomorskie	8 256 990,00
Dagmara	Pisklak	Cicha 45	Częstochowa	42-215	śląskie	8 546 040,00
Marek	Mostowiak	Piłsudskiego 135	Opole	45-016	opolskie	11 221 290,00
Rafał	Piotrowski	Konwaliowa 5	Lublin	20-023	lubelskie	11 663 475,00
Stanisław	Szczygiel	Browarna 100	Toruń	85-825	kujawsko-pomorskie	14 601 945,00

Henryk	Gall	Nieznana 99	Katowice	40-014	slaskie	15 073 650,00
Krzysztof	Andersen	Waska 13	Rzeszow	35-032	podkarpackie	16 339 320,00

3)

Zyski z samochodów w czasie od 18/01/01 do 19/12/31

Numer samochodu	Marka	Model	Skrzynia biegów	Rok produkcji	Kolor	Cena[.]	Pojemno silnika[cm3]	Moc silnika[KM]	Zysk
2	Opel	Astra	MANUALNA	2018	CZARNY	56 000,00	1400	96	732 160,00
4	Opel	Insignia	MANUALNA	2018	CZERWONY	72 000,00	1800	140	637 440,00
5	Opel	Vectra	MANUALNA	2018	ZIELONY	46 000,00	2000	150	323 080,00
8	Polonez	Caro	MANUALNA	2018	BIALY	36 000,00	1900	136	371 840,00
10	Seat	Toledo	MANUALNA	2018	BIALY	76 000,00	1900	136	456 120,00
11	Seat	Cordoba	MANUALNA	2018	SZARY	52 000,00	1400	96	718 920,00
13	Seat	Exeo	MANUALNA	2018	GRANATOWY	89 000,00	2200	180	1 303 450,00
15	Volvo	S40	MANUALNA	2018	ROZOWY	55 000,00	2000	150	475 950,00
18	Volvo	S60	MANUALNA	2018	NIEBIESKI	61 000,00	2200	180	693 930,00
19	Volvo	S80	MANUALNA	2018	NIEBIESKI	84 000,00	1800	140	703 680,00
22	Volvo	S90	MANUALNA	2018	CZERWONY	69 000,00	1800	140	620 880,00
23	Porsche	Cayenne	MANUALNA	2018	ZIELONY	100 000,00	1600	102	372 000,00
26	Porsche	Panamera	MANUALNA	2018	POMARANCZOWY	230 000,00	2200	180	1 402 500,00

Numer samochodu	Marka	Model	Skrzynia biegów	Rok produkcji	Kolor	Cena[jj]	Pojemno silnika[cm3]	Moc silnika[KM]	Zysk
27	Porsche	911	MANUALNA	2018	POMARANCZOWY	650 000,00	1400	96	2 712 000,00
30	Fiat	500	MANUALNA	2018	CZARNY	20 500,00	1900	136	214 880,00
31	Nissan	Micra	MANUALNA	2018	CZARNY	25 500,00	2000	150	283 045,00
32	Nissan	350Z	MANUALNA	2018	CZARNY	125 500,00	2000	150	778 395,00
33	Seat	Alhambra	MANUALNA	2018	SIWY	65 500,00	2500	240	640 950,00
3	Opel	Insignia	MANUALNA	2019	CZERWONY	76 000,00	1900	136	769 440,00
6	Opel	Vectra	MANUALNA	2019	ZIELONY	49 000,00	2500	240	455 560,00
7	Polonez	Caro	MANUALNA	2019	BORDOWY	41 000,00	1900	136	360 900,00
9	Seat	Toledo	MANUALNA	2019	BIALY	81 000,00	1400	96	1 145 200,00
12	Seat	Cordoba	MANUALNA	2019	SZARY	57 000,00	1800	140	787 300,00
14	Seat	Exeo	MANUALNA	2019	ROZOWY	99 000,00	2000	150	917 560,00
16	Volvo	S40	MANUALNA	2019	CZARNY	60 000,00	1400	96	697 400,00
17	Volvo	S60	MANUALNA	2019	CZARNY	65 000,00	1800	160	639 050,00
20	Volvo	S80	MANUALNA	2019	ZOLTY	89 000,00	2000	150	731 280,00
21	Volvo	S90	MANUALNA	2019	CZERWONY	79 000,00	2500	240	481 740,00

Numer samochodu	Marka	Model	Skrzynia biegów	Rok produkcji	Kolor	Cena[jj]	Pojemno silnika[cm3]	Moc silnika[KM]	Zysk
24	Porsche	Cayenne	MANUALNA	2019	SREBRNY	140 000,00	1900	136	1 483 800,0
25	Porsche	Panamera	MANUALNA	2019	BIALY	250 000,00	1600	102	1 627 500,0
28	Porsche	911	MANUALNA	2019	ZLOTY	750 000,00	1800	140	3 657 500,0
29	Fiat	500	MANUALNA	2019	CZARNY	21 500,00	2500	240	112 455,00

